MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 1: Introduction to Machine Learning

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

About Me

Vassilis Vassiliades, PhD

Research Team Leader at CYENS Centre of Excellence

Research Interests:

Machine Learning, Evolutionary Computation, Robotics

Google Scholar:

https://scholar.google.com/citations?user=_Hzp2B0AAAAJ&hl=en

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Course Website

piazza.com/ucy.ac.cy/fall2022/mai612

Your main entry point. All materials linked from there.

Main communication channel.

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Logistics

- **Lectures**
  - Fridays 15:00-18:00 (Room FST01 - 147)

- **Labs**
  - Wednesdays 16:30-18:00 (Room FST01 - 201)
  - Instructor: Giorgos Demosthenous

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Assessment

- Lab assignments: 36%
- Midterm Exam: 24%
- Final Exam: 40%

To qualify one must:

- Hand in all assignments
- Achieve at least 50% weighted average in the midterm and final exam
- Achieve at least 50% overall

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Learning Outcomes

- Understand how to structure ML projects and their lifecycle: data preparation, development, evaluation, deployment.
- Gain practical experience with various supervised learning models for regression and classification problems.
- Know how to implement unsupervised learning models for clustering, dimensionality reduction, anomaly detection and recommendation systems.
- Understand what reinforcement learning is, how it can be used for sequential decision making problems and acquire hands-on experience with it.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Syllabus

- Part 1: Introduction
  - Introduction, Data preparation
- Part 2: Supervised Learning
  - Regression, Classification, Model Evaluation and Improvement, Trees and Forests, Kernel-based methods, Neural Networks
- Part 3: Unsupervised Learning
  - Clustering, Dimensionality Reduction, Anomaly Detection, Recommender Systems
- Part 4: Reinforcement Learning
  - Introduction, Markov Decision Processes and Dynamic Programming, Model-free Prediction and Control, Continuous state and action spaces, Model-based RL

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lectures Schedule

Date	Lectures	
	Number	Title
09/09/2022	1	Introduction to Machine Learning
	2	Data preparation
16/09/2022	3	Regression
	4	Classification
23/09/2022	5	Model Evaluation and Improvement
	6	Trees and Forests
30/09/2022	7	Kernel-based methods 1
	8	Kernel-based methods 2
07/10/2022	9	Neural Networks 1: Modelling
	10	Neural Networks 2: Training
14/10/2022	11	Neural Networks 3: Intro to Deep Learning
	12	Clustering
21/10/2022	MIDTERM EXAM	

Date	Lectures	
	Number	Title
28/10/2022	PUBLIC HOLIDAY	
04/11/2022		Midterm exam solutions
	13	Dimensionality Reduction
11/11/2022	14	Anomaly Detection
	15	Recommendation Systems
18/11/2022	16	Introduction to Reinforcement Learning
	17	MDPs and Dynamic Programming
25/11/2022	18	Model-free Prediction and Control
	19	Continuous State and Action Spaces
02/12/2022	20	Model-based RL
	21	Overview of advanced ML and Revision

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Programming assignments

- Python 3.x

- Libraries:
  - numpy
  - scikit-learn
  - pandas
  - seaborn

- More information in lab session

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 1: Introduction to Machine Learning

Learning Outcomes

You will gain a basic understanding of:

1. how Machine Learning (ML) relates to Artificial Intelligence (AI)
2. the difference between traditional programming and ML
3. what ML is, some terminology and applications
4. the three main types of ML
5. the lifecycle of an ML project

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

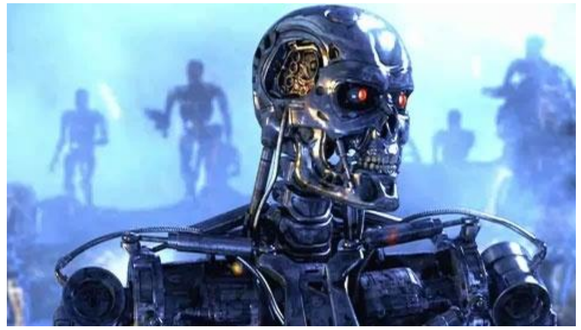
Master programmes in Artificial Intelligence 4 Careers in Europe

Artificial Intelligence

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Sci-fi Quiz



Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Intelligence

"Capacity for: logic, understanding, self-awareness, learning, emotional knowledge, reasoning, planning, creativity, critical thinking, and problem solving."

"Ability to perceive or infer information, and to retain it as knowledge to be applied towards adaptive behaviors within an environment or context."

Source: https://en.wikipedia.org/wiki/Intelligence

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

What is Artificial Intelligence?

Can it see?
(and identify things)

Can it hear?
(and respond in a useful and sensible way)

Can it read?
(and analyse patterns in text)

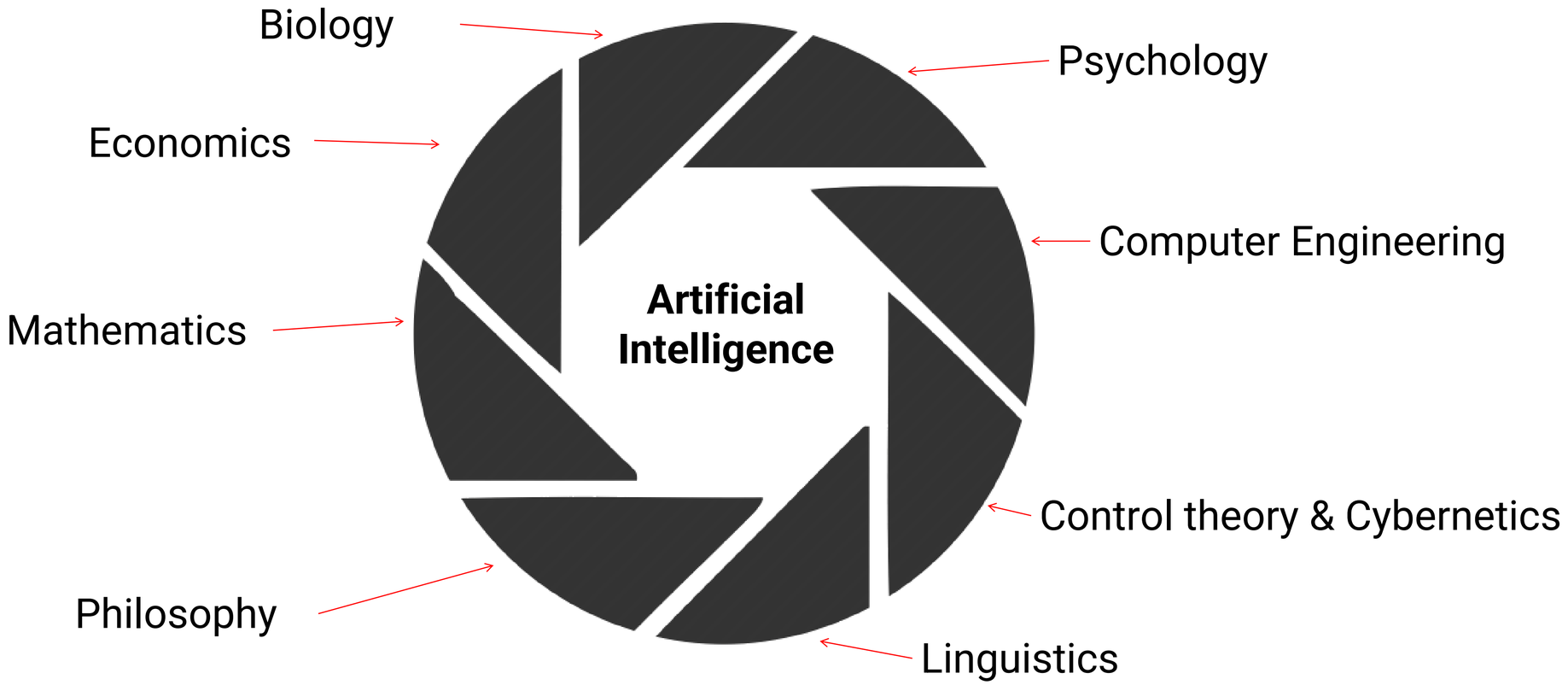
Can it reason?
(e.g. draw logical conclusions)

Can it move?
(by itself, not on a programmed path)

Can it learn?
(modify its behavior to become more efficient)

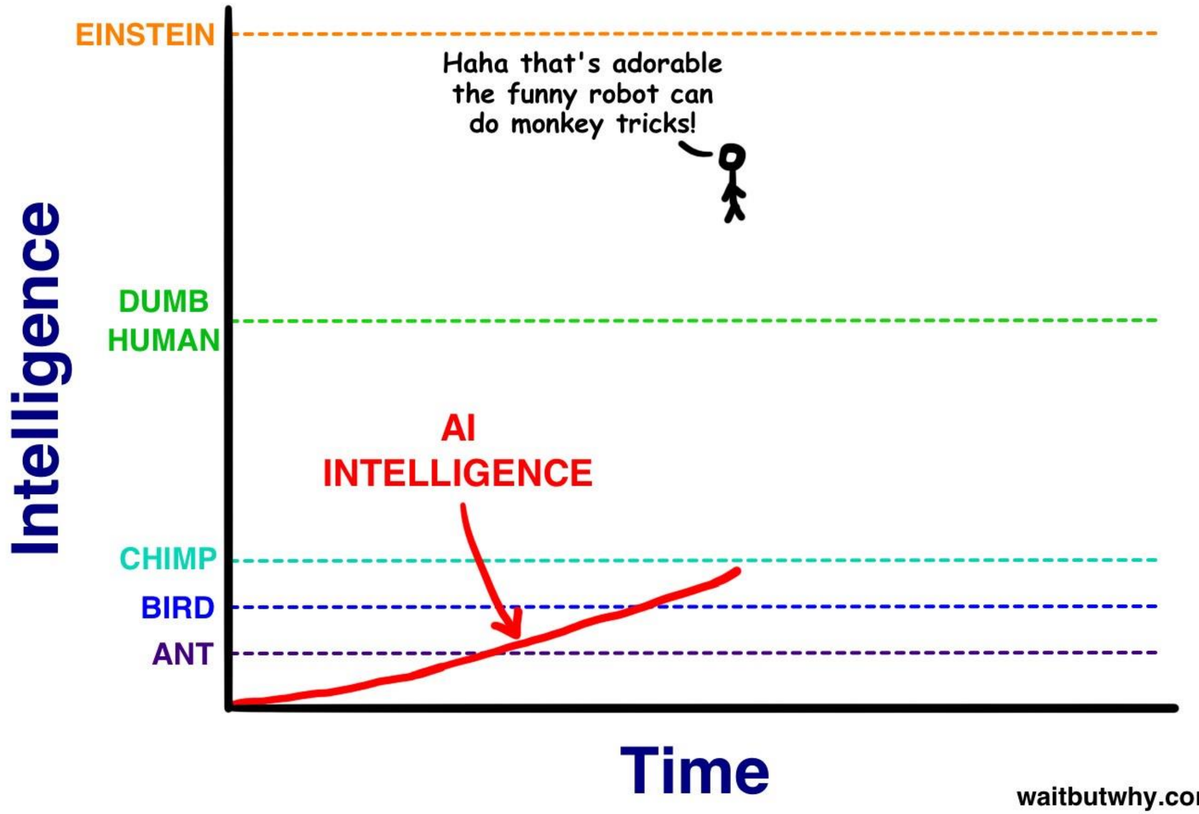
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Foundations of AI

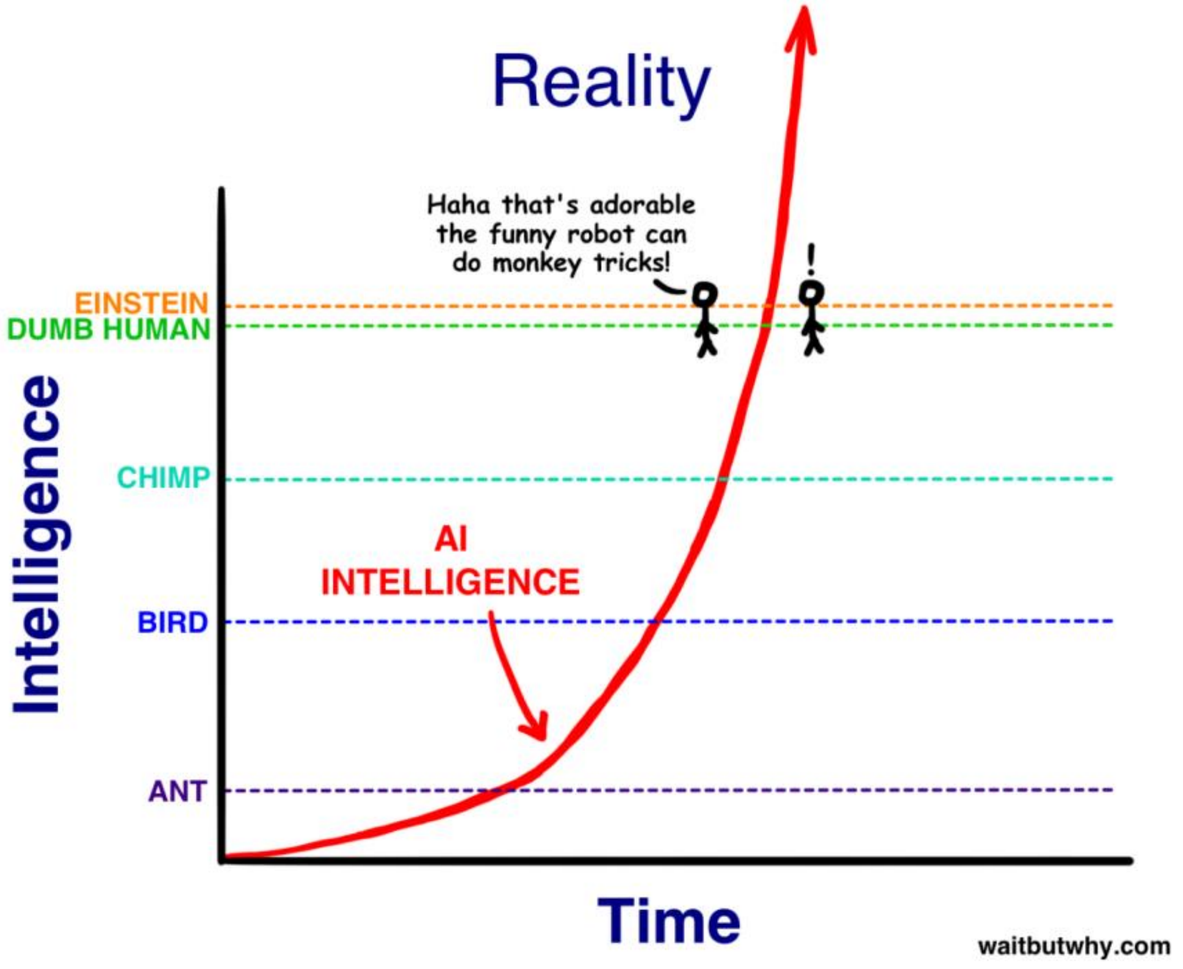


This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Our Distorted View of Intelligence



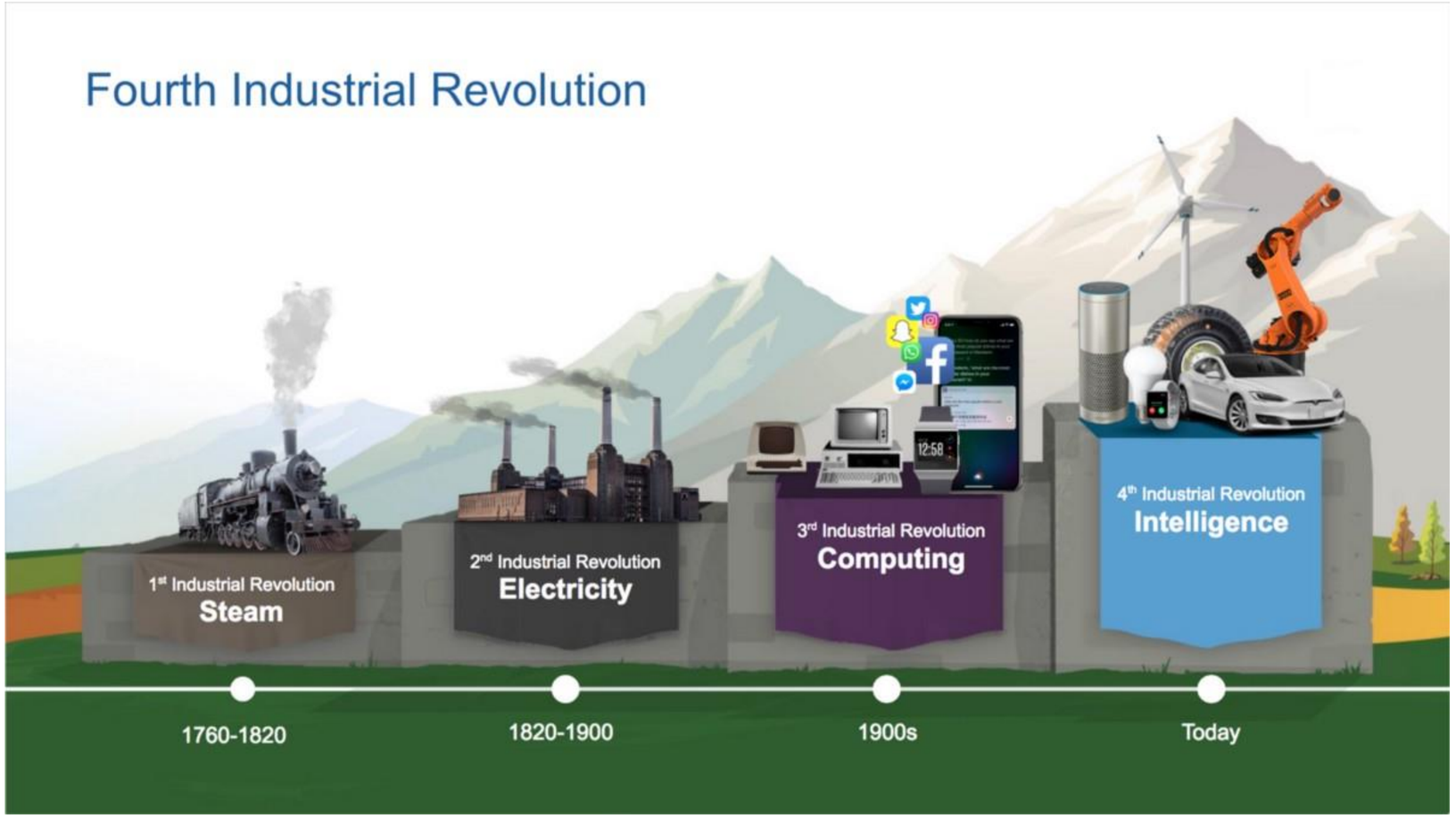
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423



Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

The fourth industrial revolution

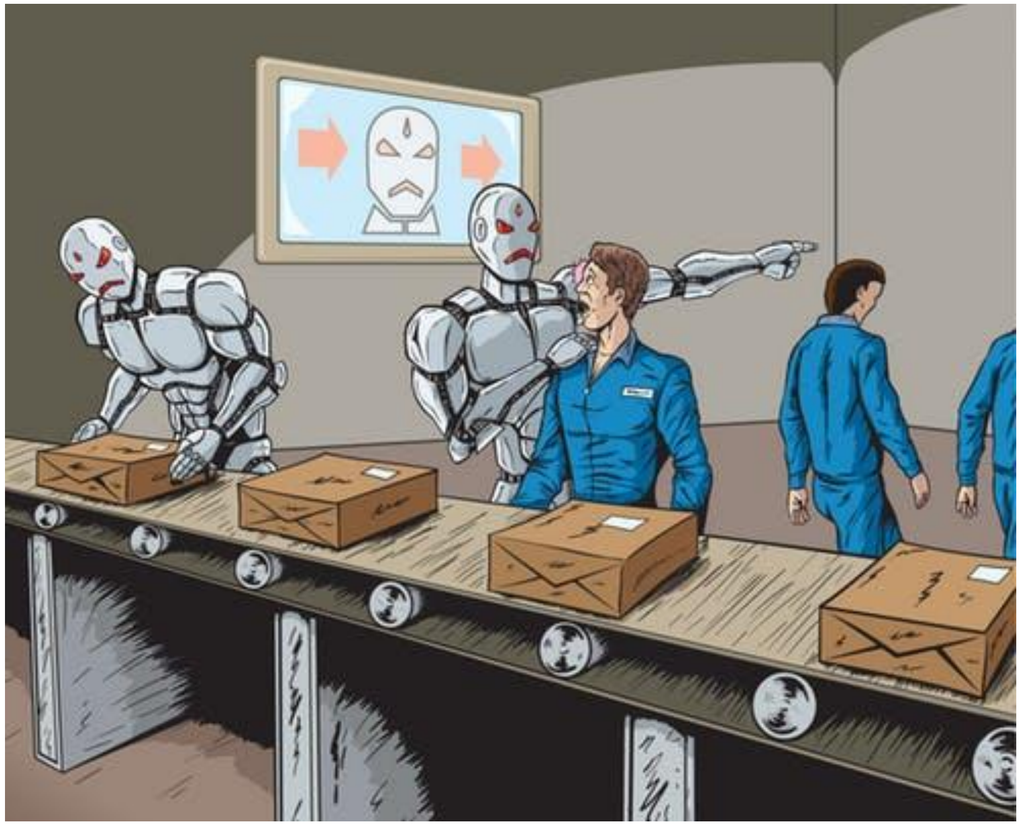


source

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Will robots take over our jobs?



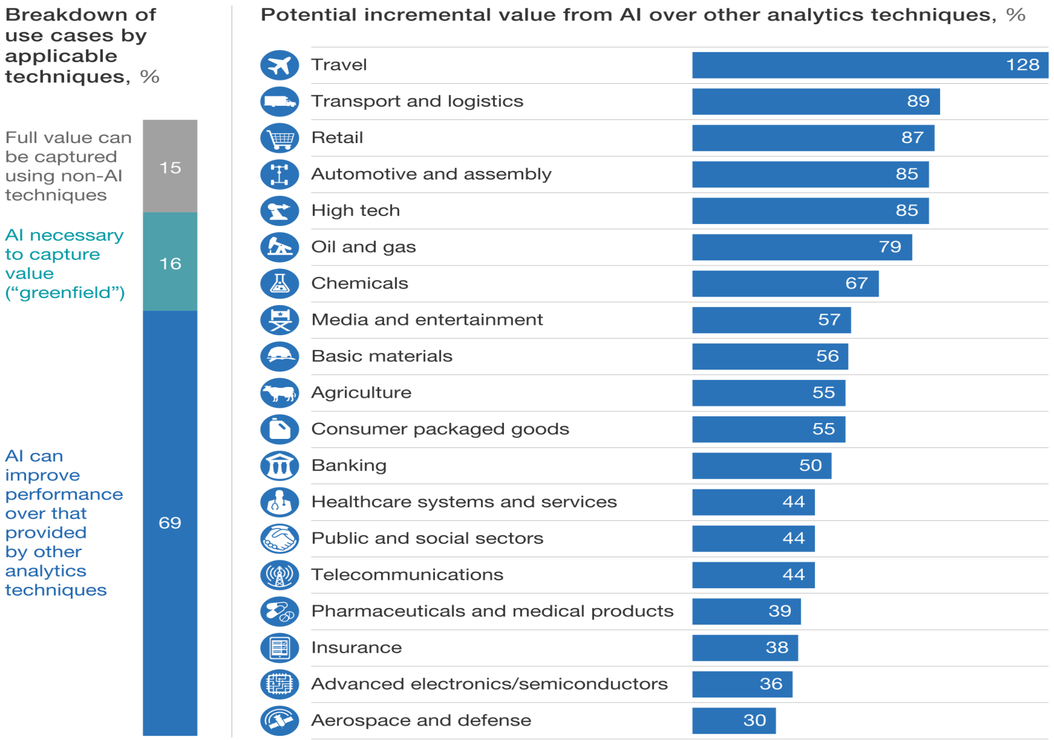
Which jobs will be replaced by AI?

Taxi drivers?
Lawyers?
Artists?

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

AI impact in all industries

In more than two-thirds of our use cases, artificial intelligence (AI) can improve performance beyond that provided by other analytics techniques.

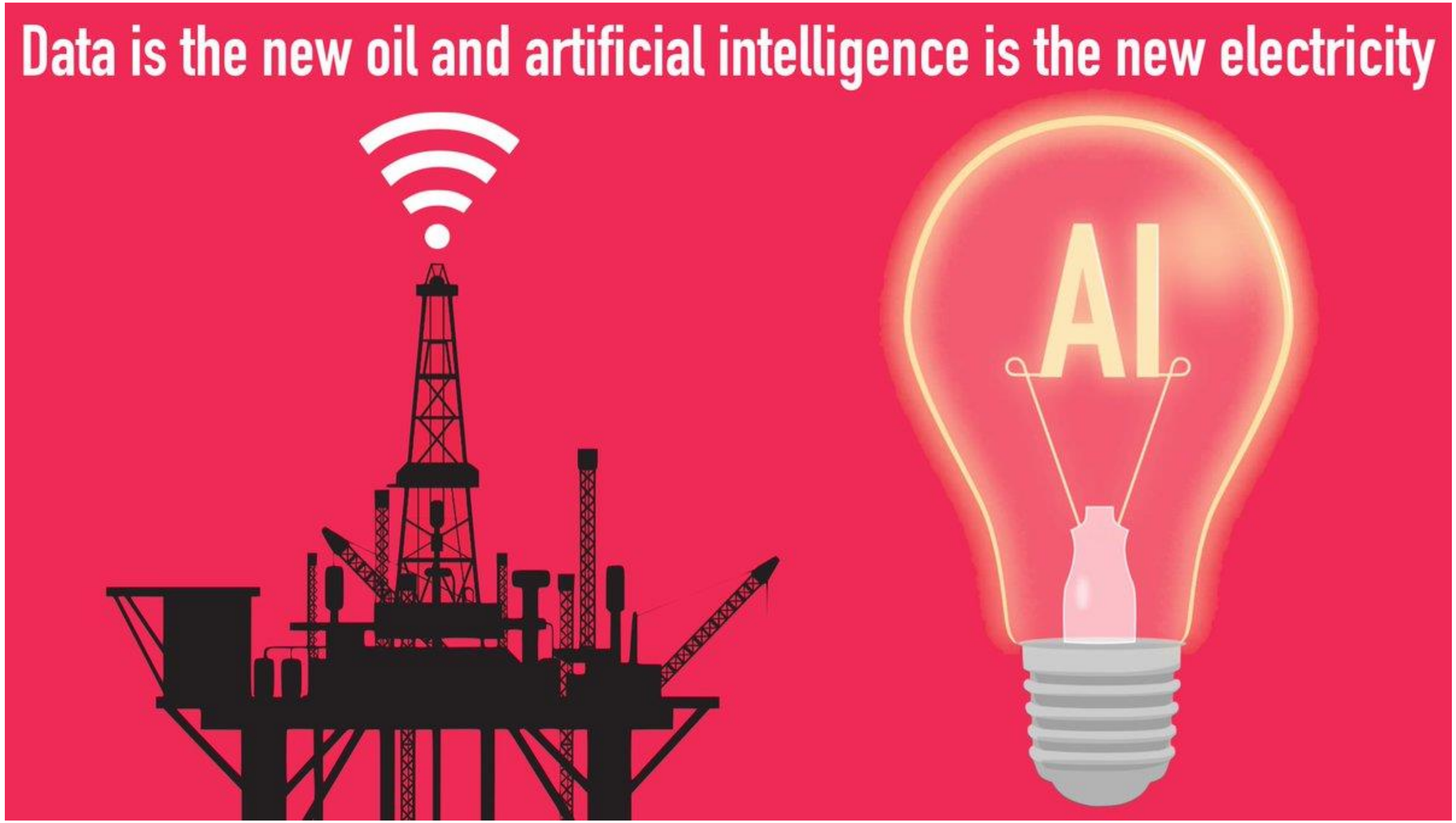


McKinsey&Company | **Source:** McKinsey Global Institute analysis

- Technology has created more jobs than it has destroyed (source)
- UK government estimated that AI would add £630 billion to UK economy by 2035!

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423



source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Machine Learning

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Programming vs Learning: Let's program a fruit image classifier

if most pixels are yellow: banana
else: strawberry

if most pixels are yellow: banana
else if most pixels are red: strawberry
else: lime

if most pixels are yellow:
  if shape is oval-like: lemon
  else: banana
else most pixels are red: strawberry
else: lime

Traditional programming is tedious and prone to human error!

Can we automate the discovery of the "correct" program?

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Programming vs Learning

Software 1.0

Programmer:

- identifies desirable behavior of every case
- writes explicit instructions to the computer

Source code:

- written in languages such as Python, C++, ...
- compiled into a binary which becomes a specific point in program space

Software 2.0

Programmer:

- specifies the **goal** of the desirable program (e.g., “correctly classify a dataset of input-output pairs of examples”, or “win a game of chess”)
- writes a **rough skeleton** of the code that identifies a subset of program space to search
- provides an algorithm to **search** this space.

Source code:

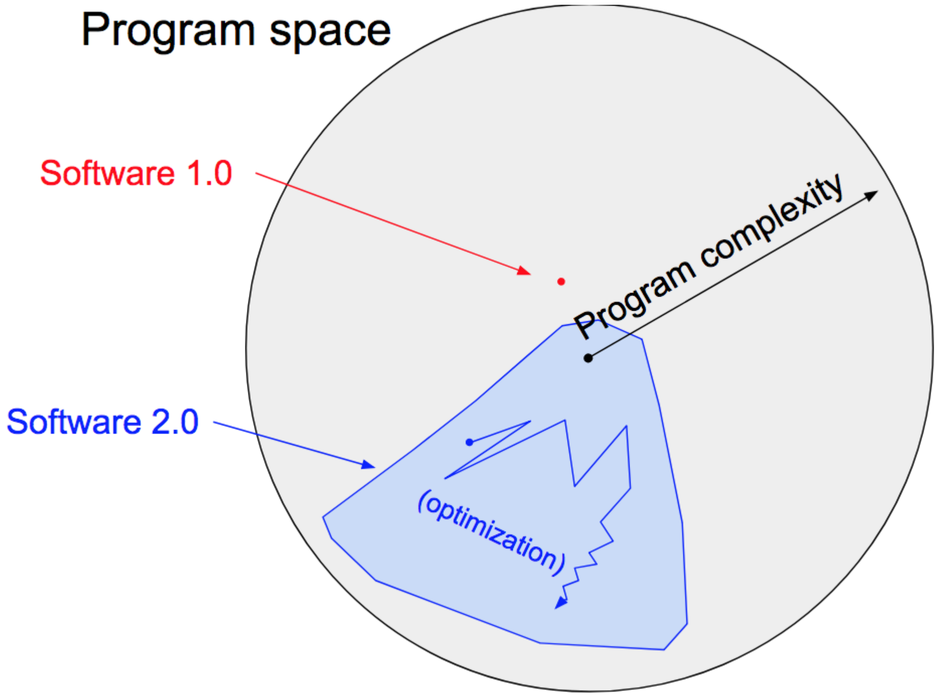
- dataset and/or functions defining desirable behavior
- “initial model” that gives the rough skeleton of the code but with details to be filled in
- “compilation” is the process of searching through program space for the “final model”

Adapted from source

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

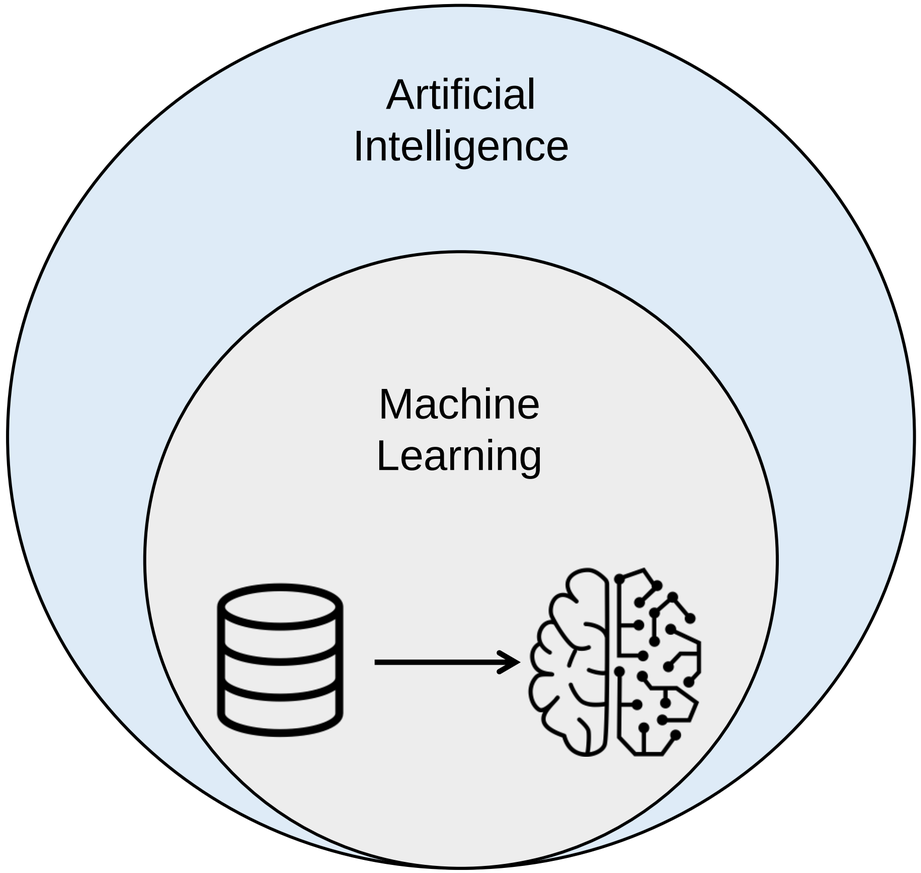
Programming vs Learning



Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Machine Learning (ML)



- Subfield of AI that teaches computers how to learn and act without explicitly programming them.
- Discover the hidden structure and regularities in the data.
- Use these to make good predictions about the future (unseen data).
- Program Discovery from Data.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

ML Definition and Terminology

"A **computer program** → **Model**

is said to

learn → **Learning Algorithm**

from

experience E → **Data**

with respect to some

task T → e.g., **object recognition, fraud detection, forecasting, customer segmentation**

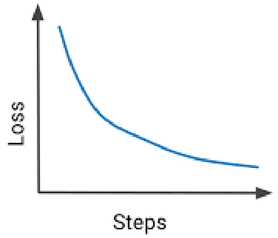
and some

performance measure P, → **Error Function**

if its performance on T, as measured by P,

improves → **decreasing error**

with experience E."



- Tom Mitchell. Machine Learning (1997)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

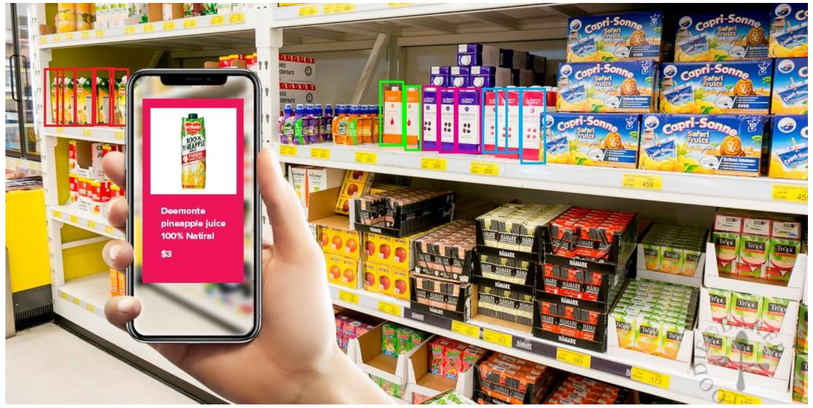
Master programmes in Artificial Intelligence 4 Careers in Europe

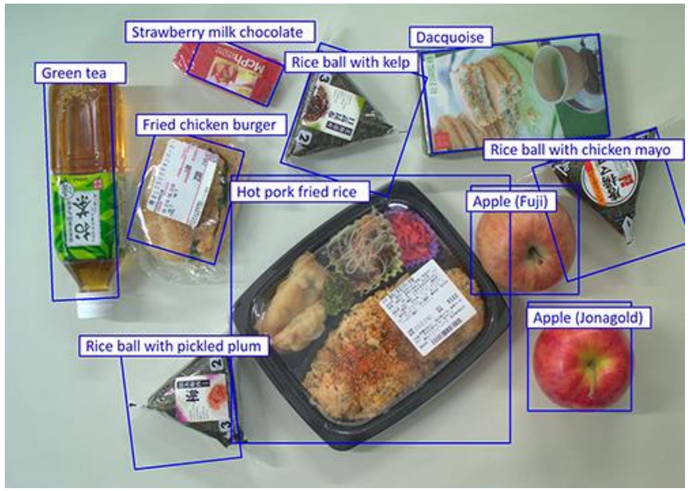
Applications

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Image Recognition



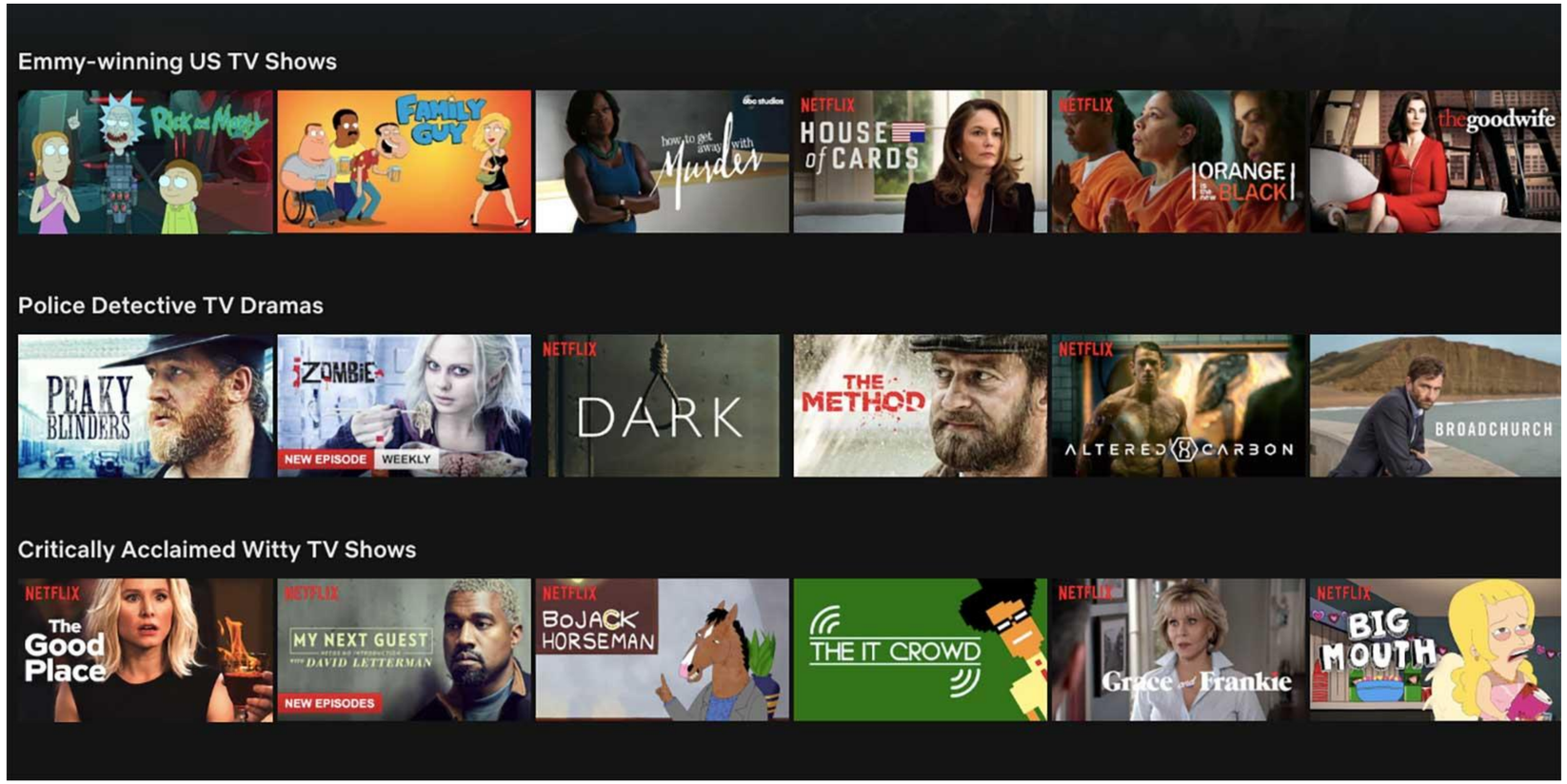


This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Video Games

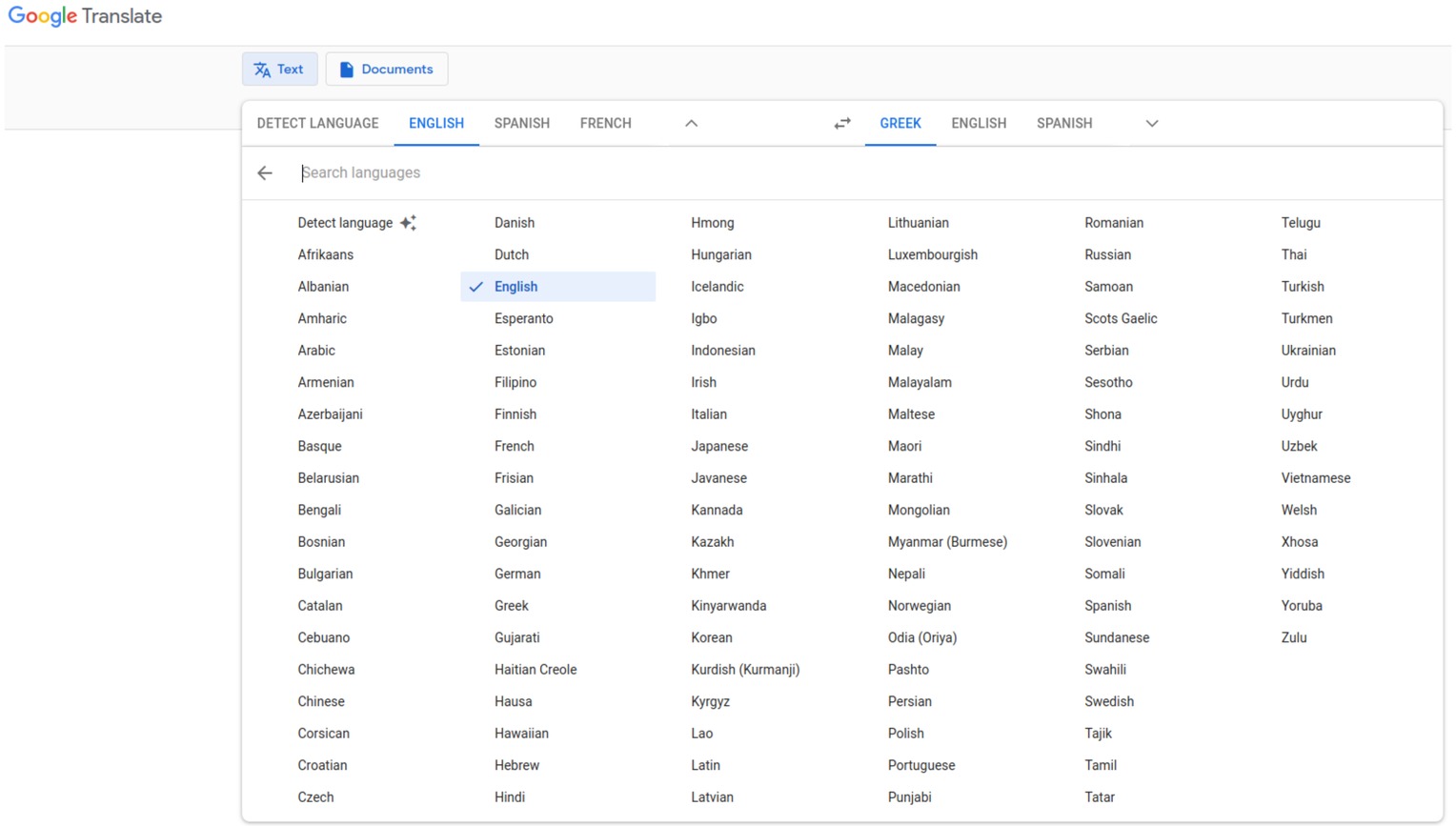
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Recommendation Systems



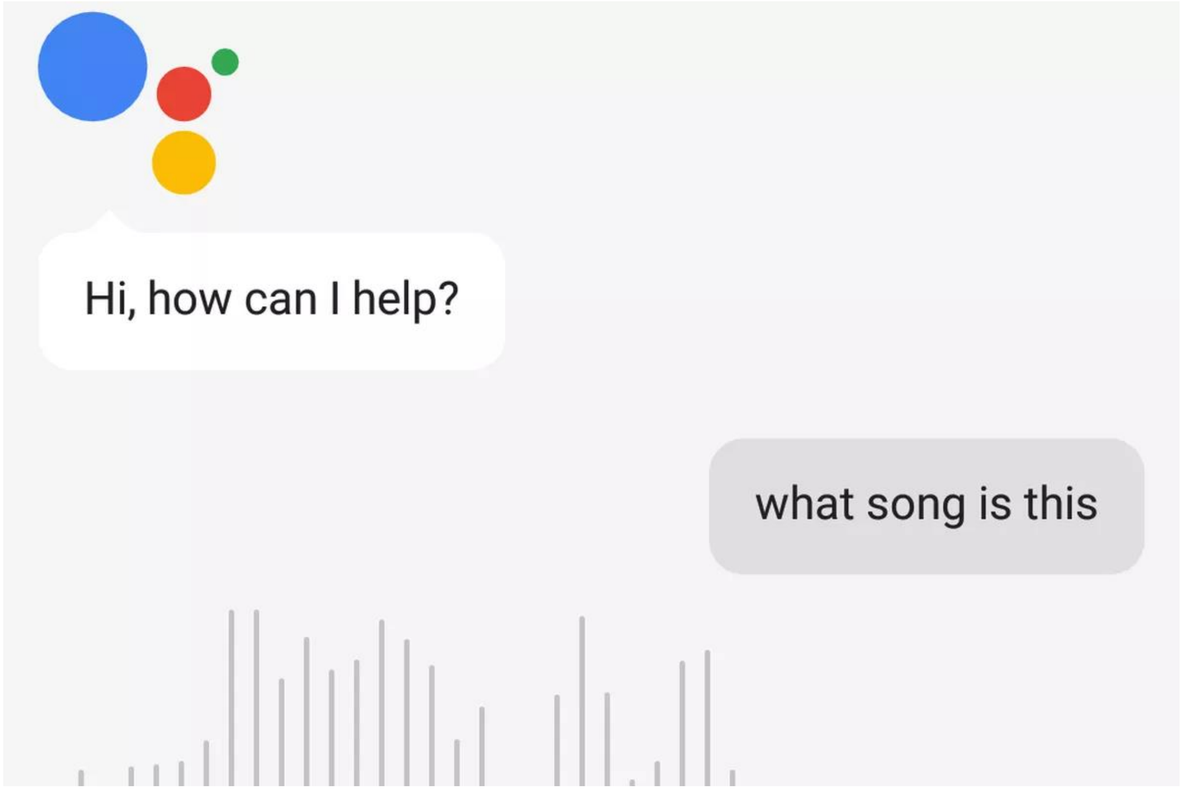
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Natural Language Translation



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Personal Assistants



This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Image Generation from Prompt

"Super Mario in the style of a Greek god statue, realistic photo in Athens"
- Generated using Dalle2

“Oriental painting of tigers wearing VR headsets during the Song dynasty”
- Generated using Imagen

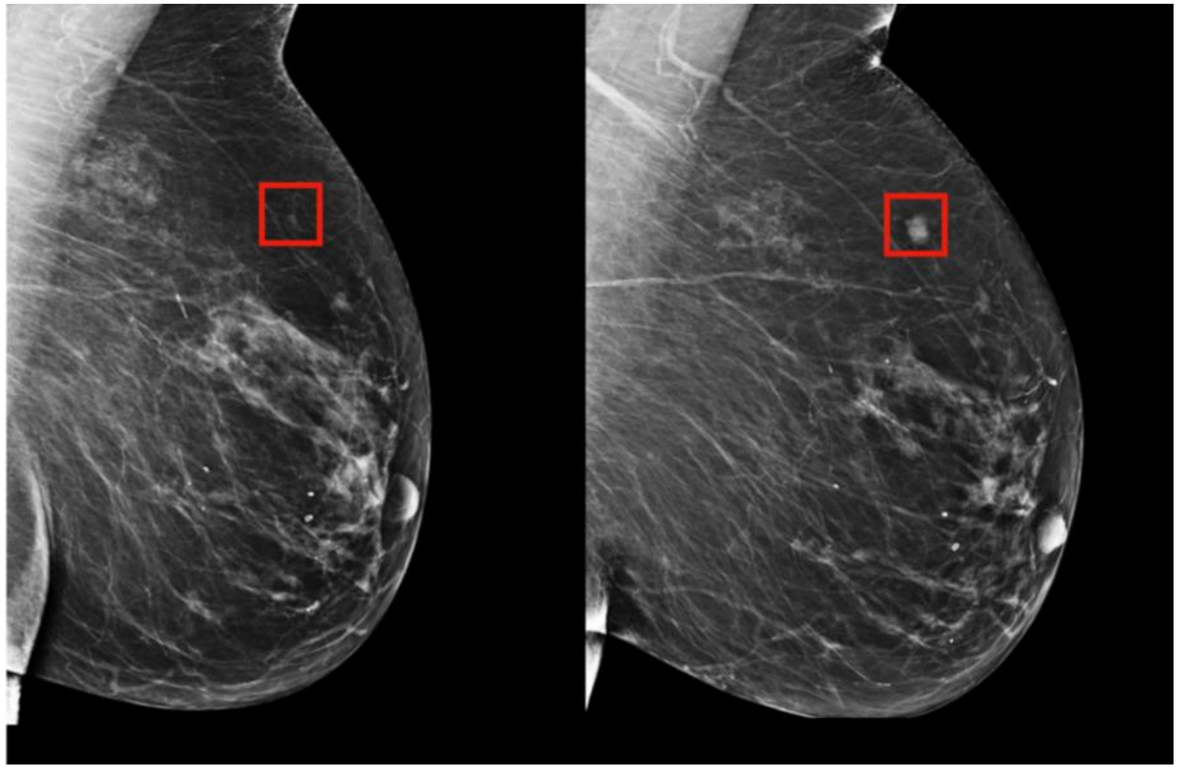
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Self-driving cars

Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Breast cancer prediction



Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Quiz (T / F)

1. Artificial intelligence is about learning from data

   False

2. A program uses machine learning if it improves at problem solving with experience

   True

3. The goal of machine learning is to create predictive models that generalize to unseen data

   True

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

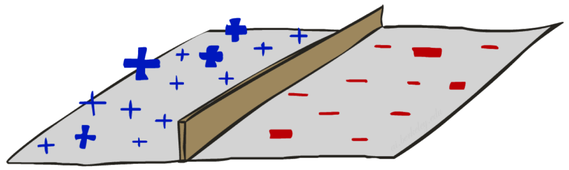
Types of ML

Co-financed by the European Union
Connecting Europe Facility

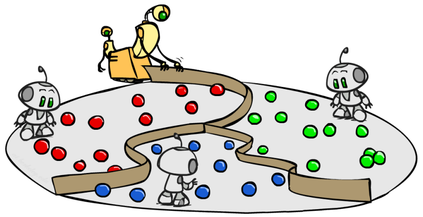
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Types of ML approaches

Supervised learning
learning with a teacher



Unsupervised learning
learning without any external feedback



Reinforcement learning
trial-and-error learning

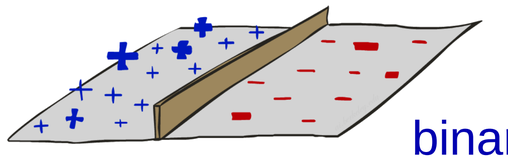
Images Source: ***UC Berkeley CS188 – Intro to AI :*** http://ai.berkeley.edu/course_schedule.html

Co-financed by the European Union
Connecting Europe Facility

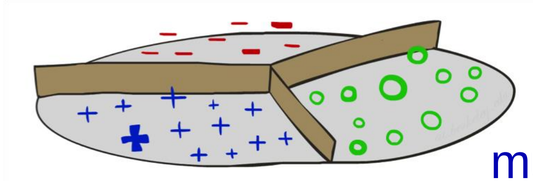
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Supervised Learning

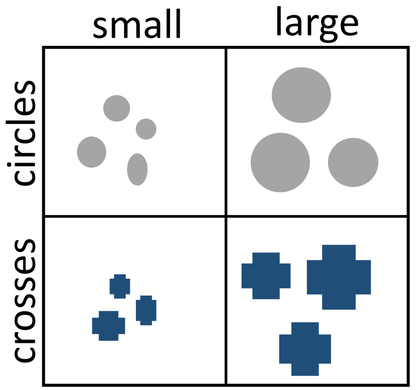
Classification (discrete variables)



binary

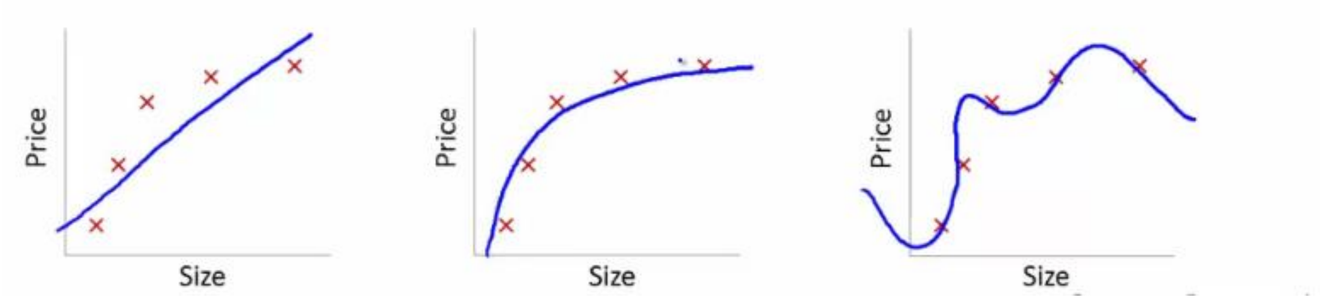


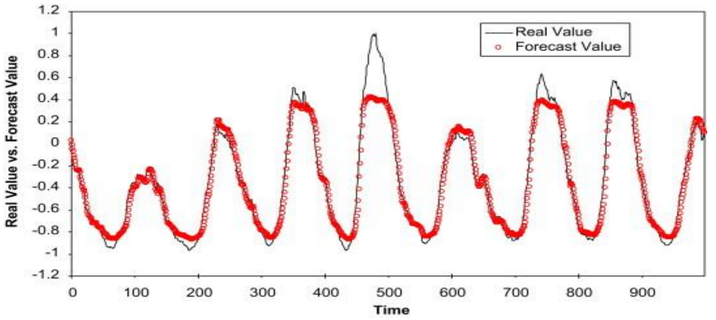
multi-class



multi-label

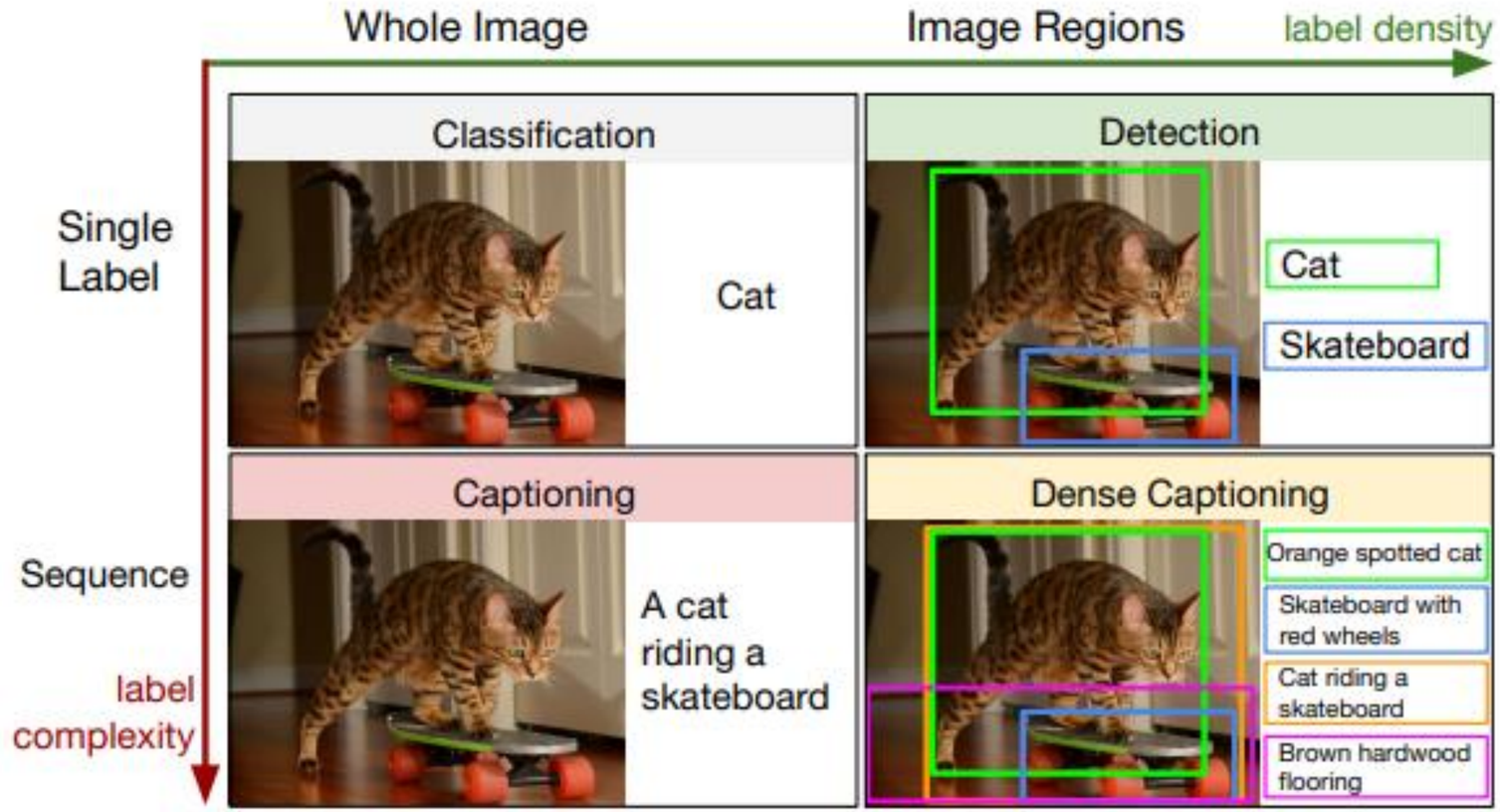
Regression (continuous variables)





This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

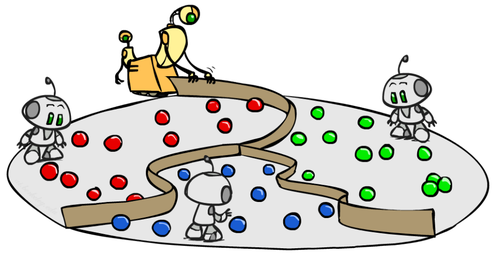
Image captioning



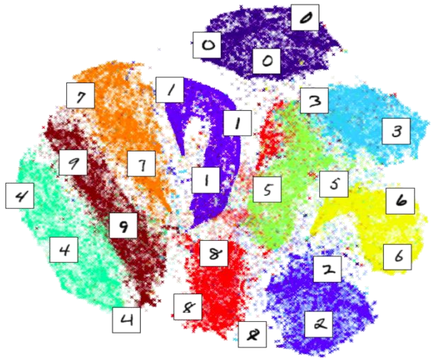
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Unsupervised Learning

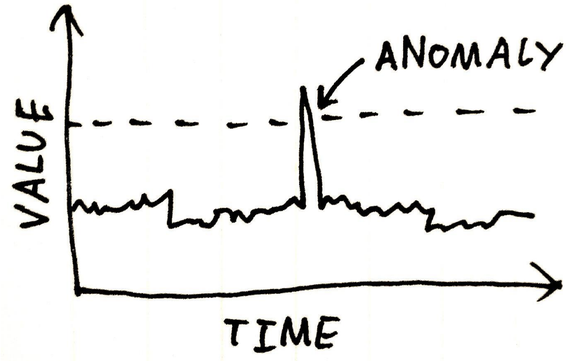
Clustering



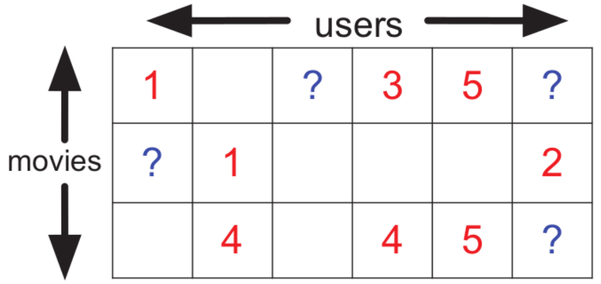
Dimensionality Reduction



Anomaly Detection



Matrix Completion



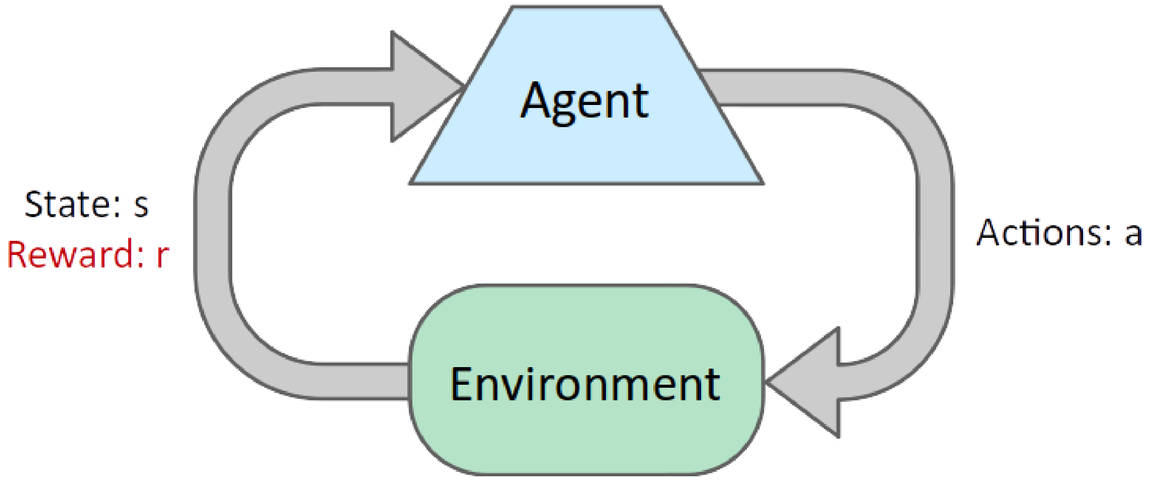
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Smart paintbrush

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Reinforcement Learning

- Learning by trial and error
- Not just prediction
- Feedback in the form of rewards
- Agent needs to learn to act so as to maximize cumulative reward



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

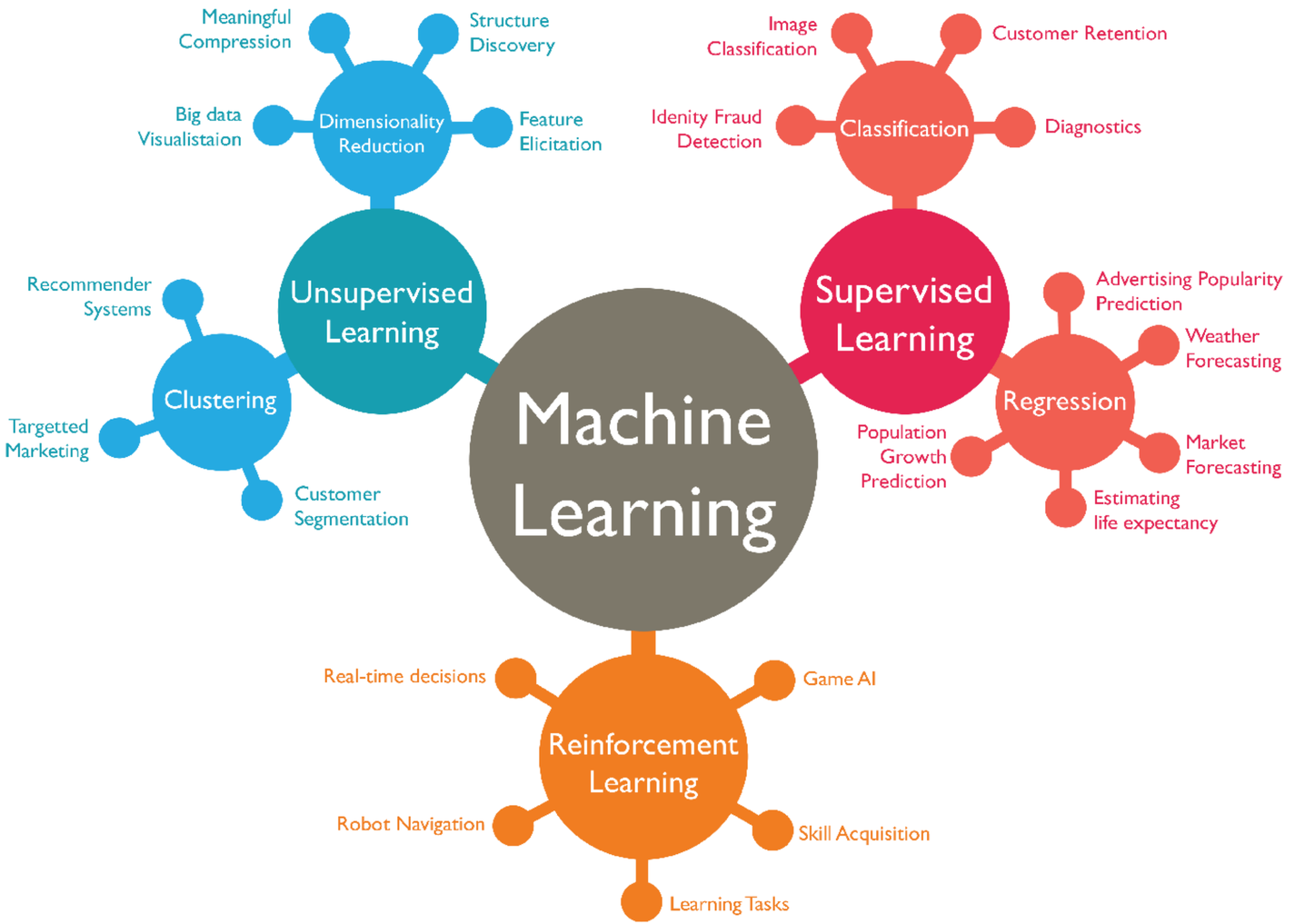
Robotic Quadrupedal Locomotion

YouTube Video

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Learning robust perceptive locomotion for quadrupedal robots in the wild

Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen,
Vladlen Koltun, Marco Hutter



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

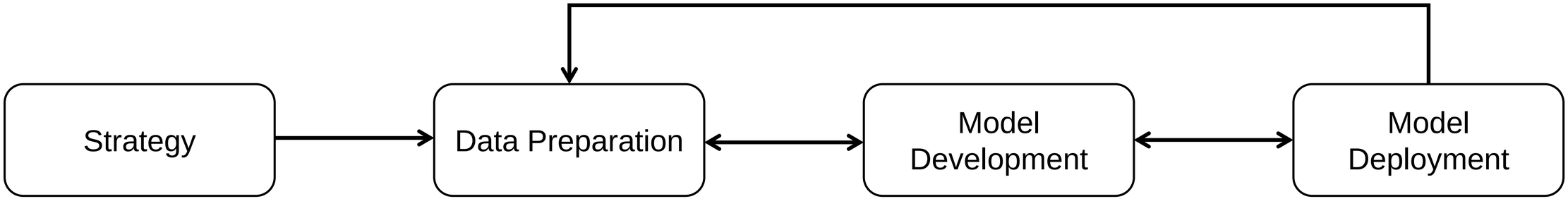
Quiz

In terms of type of machine learning approach (supervised/unsupervised/reinforcement) and problem (e.g., binary classification, clustering etc.) how would you characterize the following:

1. Spam filter — Supervised: Binary classification
2. Grouping customers to drive marketing actions — Unsupervised: Clustering
3. Animal sound recognition — Supervised: Multiclass classification
4. Music completion — Unsupervised
5. Identifying if an industrial machine is faulting, from its noise levels — Supervised/Unsupervised: Anomaly Detection
6. Teaching a robotic arm to pick and place various objects — Reinforcement Learning

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Machine Learning Project Lifecycle



- Project definition
- Objectives
- Solutions
- Requirements
- Planning

- Collection
- Preprocessing
- Transformation
- Visualization
- Dataset splitting

- Model training
- Model evaluation and testing
- Iterative improvement

- Model optimization
- Containers
- Web services
- Mobile devices
- Monitoring and maintainance

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 2: Data Preparation

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 2: Data Preparation

Learning Outcomes

You will understand:

1. the process of collecting data and preparing datasets for developing machine learning solutions
2. the importance of data preparation
3. the data preprocessing and data transformation steps
4. the importance of data visualization and exploratory data analysis
5. why we split a dataset into a training and a test set

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Fruit image classifier using ML

You are a ML engineer at company XYZ and you are tasked to create a mobile app that classifies different fruit images, to their respective classes (apple, orange, etc.).

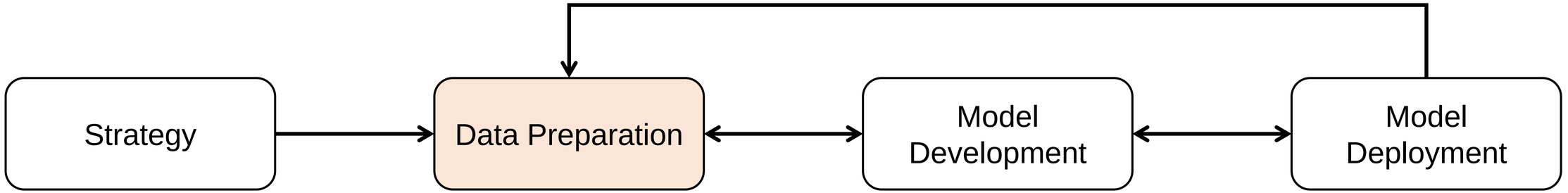
You want to use ML

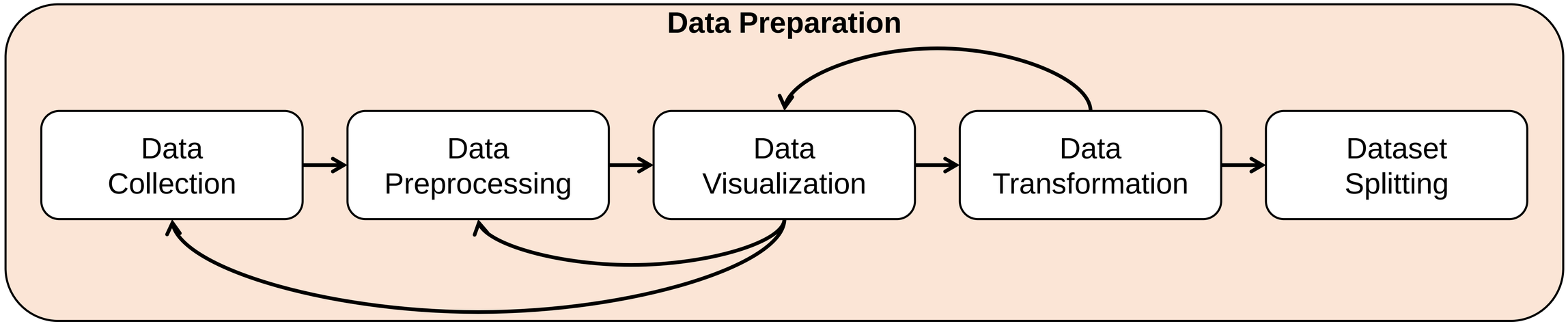
What is the first step of the process?
- ➢ Data collection

How?
- ➢ Online
- ➢ Take photos, and associate them with labels

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Preparation

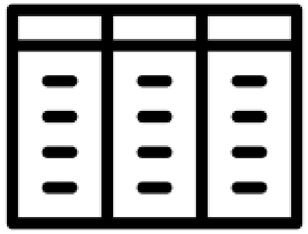




Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data

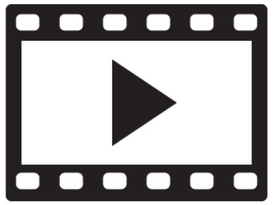


tabular

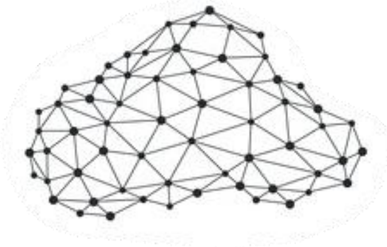
text

signals

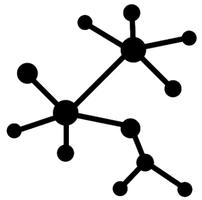
images



video



point clouds



graphs

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Data Collection

Co-financed by the European Union
Connecting Europe Facility

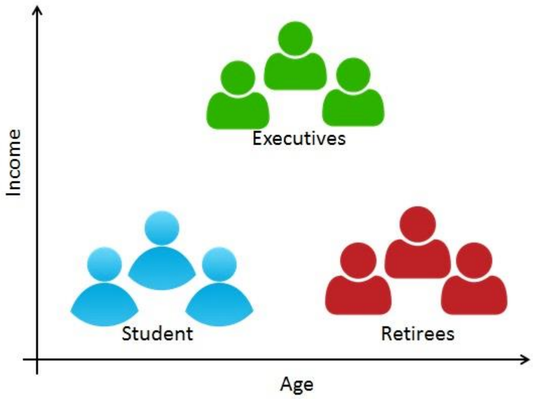
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Collection

Data may need to be acquired for the task at hand

- fruit images for a fruit image classifier

- questionnaire with demographics for customer segmentation



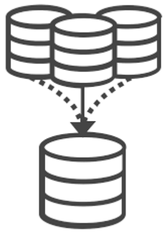
- a robotic arm learning to grasp different objects

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Collection

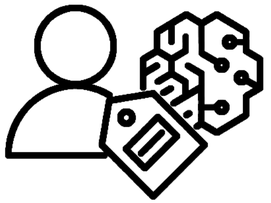
Data already exist:

- may need to be integrated from multiple sources



e.g. in fraud detection: timing between two consecutive transactions and distance between places where transactions happened

- may need to be labelled
  - all supervised learning settings
  - requires human effort (often domain experts)
  - can be outsourced (data labelling services, e.g., Amazon Mechanical Turk)
  - transfer learning: use an already trained model with its labels

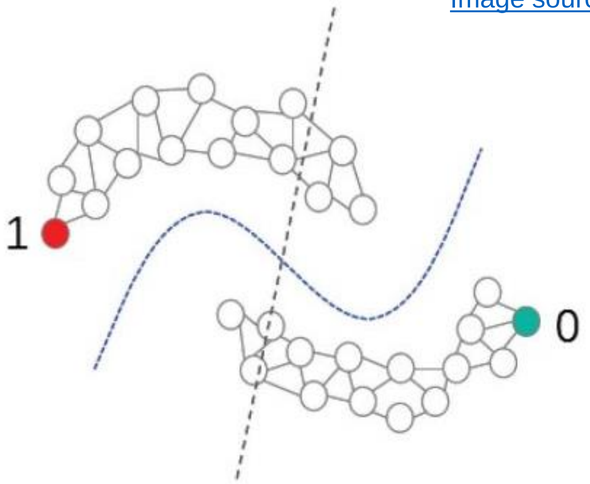


This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Collection

More advanced ML types related to data collection:

Image source



Semi-supervised learning

- Large amount of unsupervised data, small amount of labelled data
- Between unsupervised and supervised learning

Active Learning

- Used when labelling is costly
- Intelligently choose the next data point to label in a way that minimizes the effort of labelling

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Collection

Other issues:

Data Privacy

- Need to work with anonymized data
- The preprocessing step deals with sensitive data

Class Imbalance

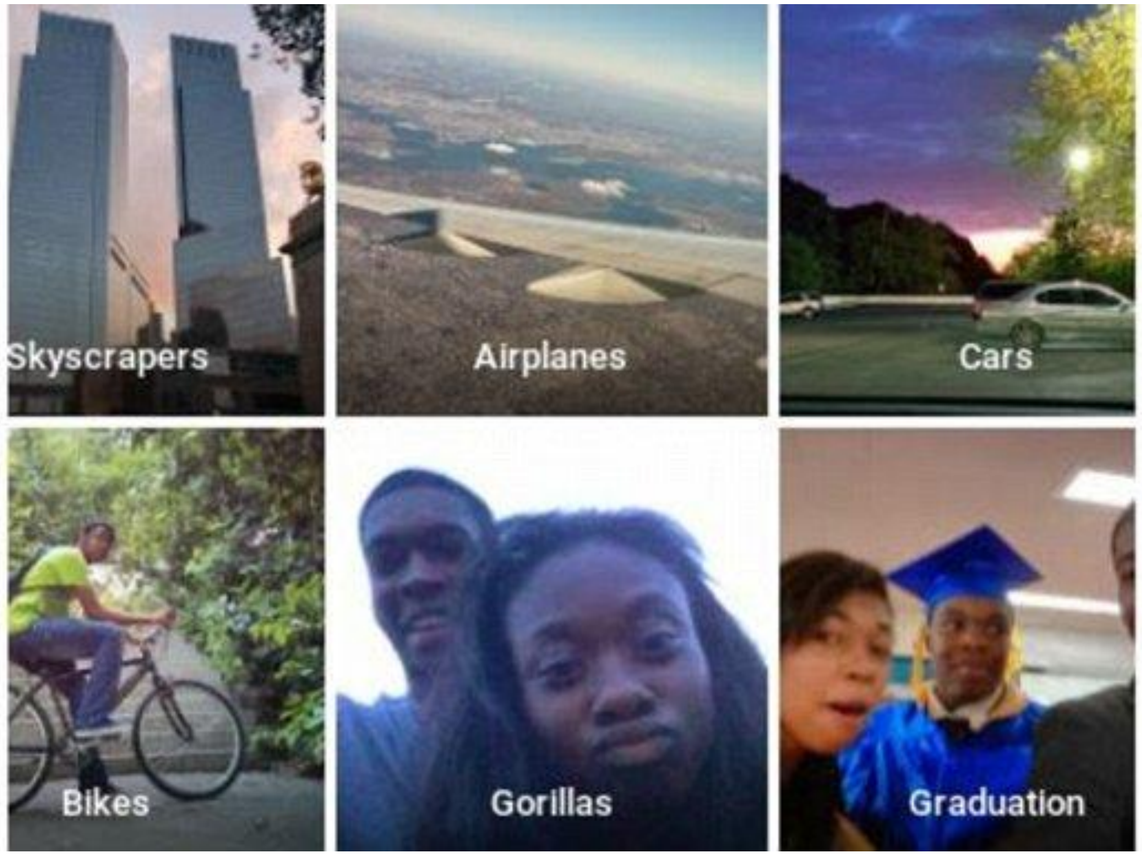
- Need to have approximately the same number of instances for all classes in classification tasks

Bias & Fairness

- It is very important to have diversity in our data
- See next slides

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Bias & Fairness: The ‘Gorillas’ Incident

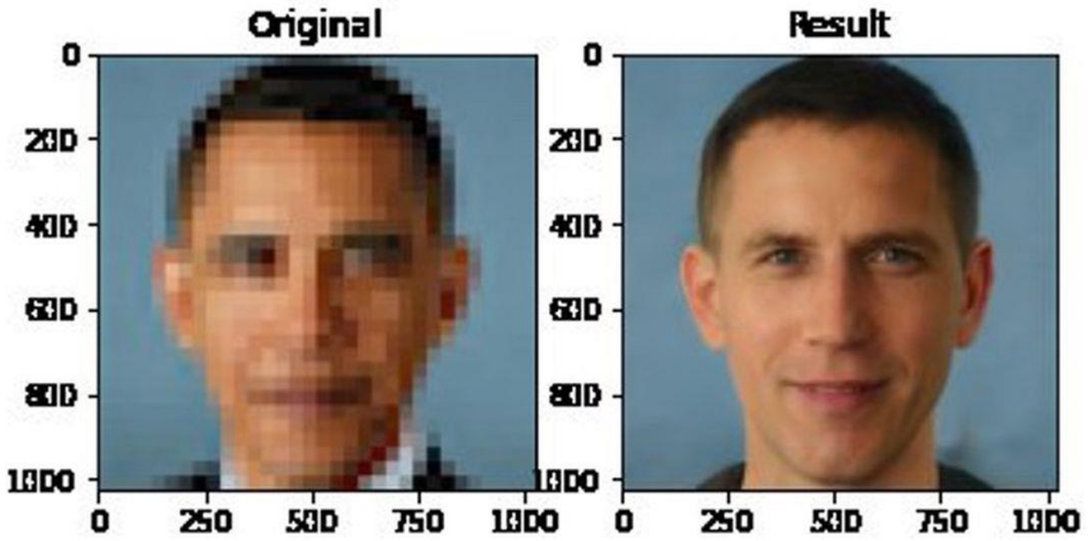


One of these predictions is very wrong!

Why do you think this happened?

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Bias & Fairness: Obama reconstructed as white male



Face upsampling system makes everyone look white because it was trained on a dataset which mainly contains pictures of white people.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Bias & Fairness: Error rate per gender and race

Gender Classifier	Darker Male	Darker Female	Lighter Male	Lighter Female	Largest Gap
Microsoft	94.0%	79.2%	100%	98.3%	20.8%
FACE++	99.3%	65.5%	99.2%	94.0%	33.8%
IBM	88.0%	65.3%	99.7%	92.9%	34.4%

All models have more trouble classifying correctly darker females than lighter males.

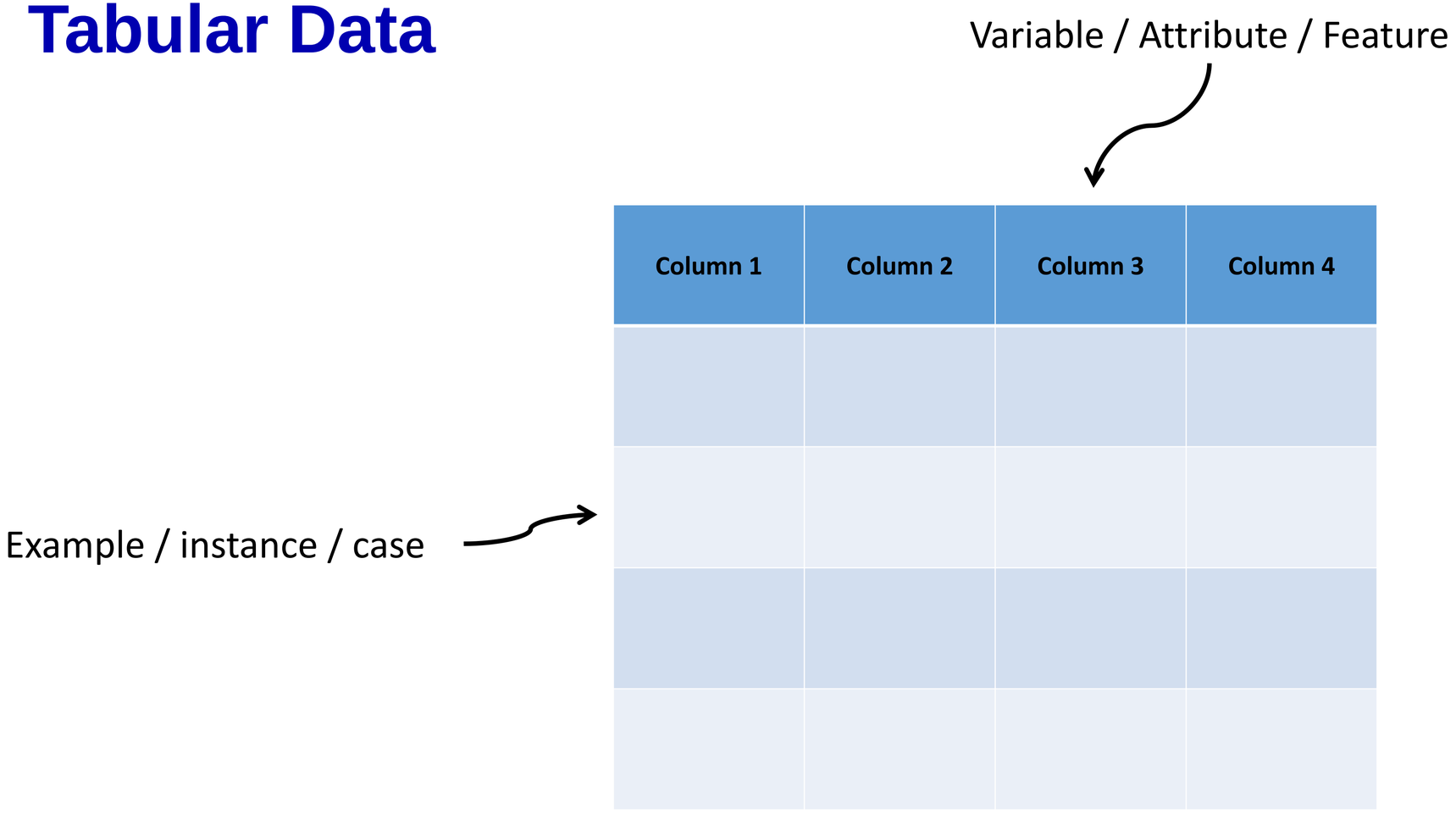
- Classifiers were not trained with sufficient diversity
- Importance of having balanced datasets

Your model is only as good as your data!

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Tabular Data



Input variables: columns provided to the model to make a prediction

Output variable: column to be predicted by the model

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Variable Types

Qualitative / Categorical

Made up of words

1. Nominal (named categories)
   - e.g. gender

2. Ordinal (order matters)
   - e.g. mood

Quantitative / Numerical

Made up of numbers

1. Discrete (integers)
   - e.g. shoe size

2. Continuous (any number)
   - e.g. weight

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Variable Types Quiz

Categorical (Nominal, Ordinal) or Numerical (Discrete, Continuous) ?

1. Ethnicity	Nominal
2. Age	Continuous or Discrete
3. Color	Nominal
4. Temperature	Continuous
5. Number of children	Discrete
6. Satisfaction rating (unhappy/neutral/happy)	Ordinal
7. Date?	String: Needs feature transformation

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Data Preprocessing

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Preprocessing

What is data preprocessing?

- Process of bringing **raw** data in a form suitable for modeling
- Typically involves: Data Cleaning, Data Encoding

Why is it important?

- ML algorithms expect numbers (e.g., some cannot deal with categorical attributes)
- ML algorithms have requirements (e.g., cannot deal with missing values)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Preprocessing

Data Cleaning

- Fix errors: typos, incorrect capitalization, inconsistencies etc. (e.g., “N/A” vs “Not Applicable”, “Male” vs “male”, …)
- Missing data (remove, replace with 0, or interpolate)
- Remove duplicate rows/columns
- Remove irrelevant data (e.g. when analyzing data for millennial customers, remove older generations)
- Remove outliers (might be due to a mistake in data collection)

Data Encoding

- Categorical variables
- Anonymizing data

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Encoding

Encode **categorical** variables using a numerical representation. Typically:

- Nominal variables: “one-hot” encoding: as many new binary variables as there are named categories

Gender = {male, female} -> GenderMale = {0,1}, GenderFemale = {0,1}
Color = {red, green, blue} -> ColorRed = {0,1}, ColorGreen = {0,1}, ColorBlue = {0,1}

- Ordinal variables: ordinal encoding: convert to integer values

SatisfactionRating = {sad,neutral,happy} -> {0,1,2}

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Encoding: One-hot encoding example

```
In [1]: # example of a one-hot encoding
   ...: from numpy import asarray
   ...: from sklearn.preprocessing import OneHotEncoder
   ...:
   ...: # define data
   ...: data = asarray([['red'], ['green'], ['blue']])
   ...: print(data)
   ...:
   ...: # define one hot encoding
   ...: encoder = OneHotEncoder(sparse=False)
   ...:
   ...: # transform data
   ...: onehot = encoder.fit_transform(data)
   ...: print(onehot)
[['red']
 ['green']
 ['blue']]
[[0. 0. 1.]
 [0. 1. 0.]
 [1. 0. 0.]]
```


This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Data Encoding: Ordinal encoding example

```
In [1]: # example of an ordinal encoding
   ...: from numpy import asarray
   ...: from sklearn.preprocessing import OrdinalEncoder
   ...:
   ...: # define data
   ...: satisfaction_rating = asarray([['sad'], ['neutral'], ['happy']])
   ...: print(satisfaction_rating)
   ...:
   ...: # define ordinal encoding
   ...: encoder = OrdinalEncoder()
   ...:
   ...: # transform to numbers
   ...: result = encoder.fit_transform(satisfaction_rating)
   ...: print(result)
[['sad']
 ['neutral']
 ['happy']]
[[2.]
 [1.]
 [0.]]
```


This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Data Visualization

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Visualization

Exploratory data analysis

- are there any obvious patterns (might give hints on what method to choose)?
- are there any obvious problems (label noise or outliers)?
- involves basic statistics (e.g., mean, variance, correlation coefficient) and plots

- Tabular data with small number of features: use a **pair plot**

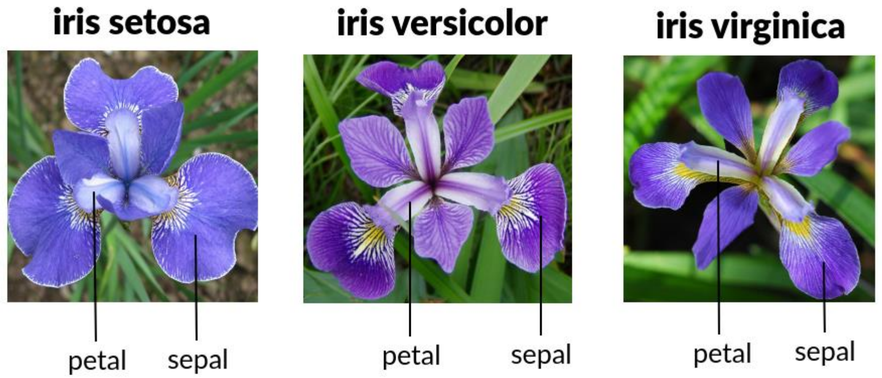
- High-dimensional data: use **dimensionality reduction**
  - e.g., Principal Components Analysis
  - will study in later lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

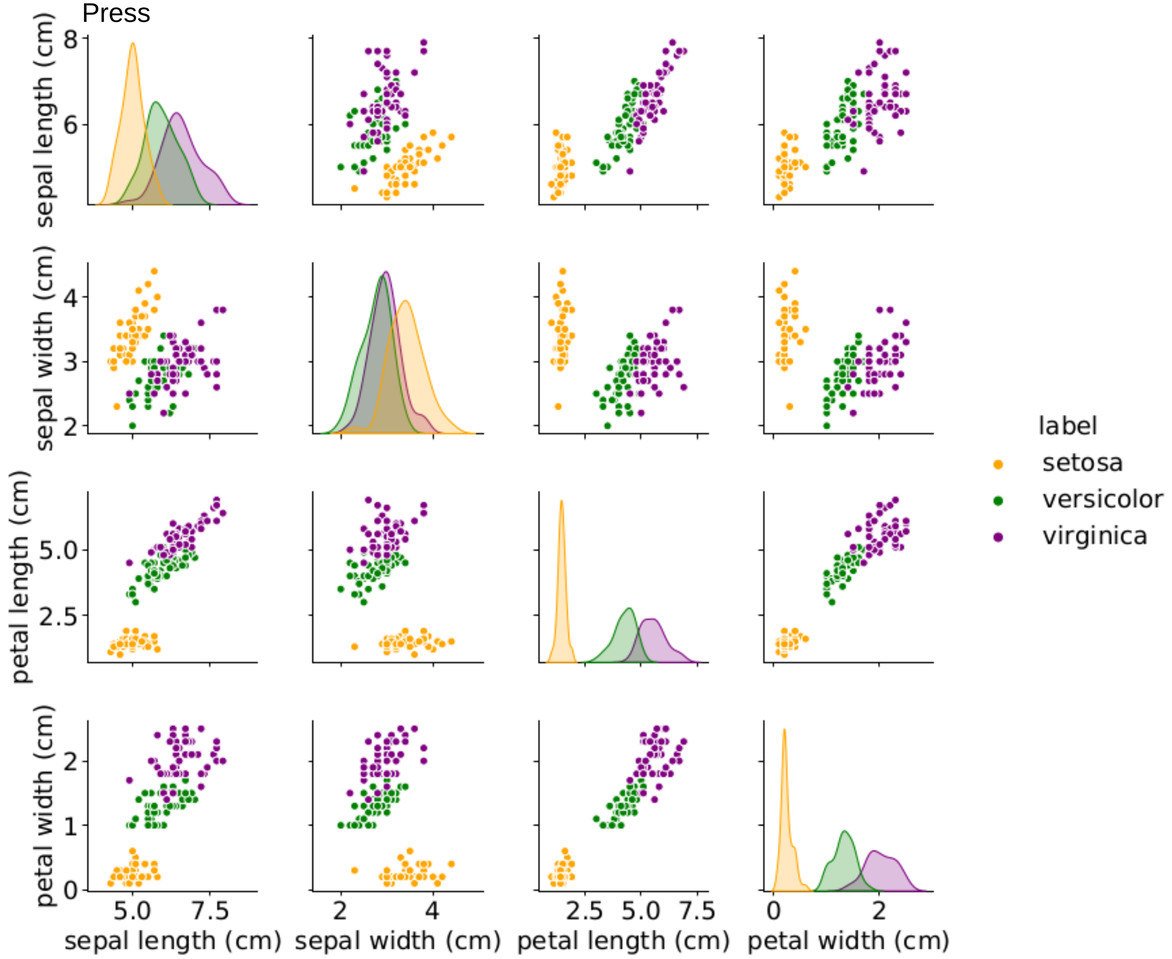
Iris dataset

- 3 classes of 50 instances each
- 4 features (petal/sepal length/width in cm)



Which class is linearly separable from the others?

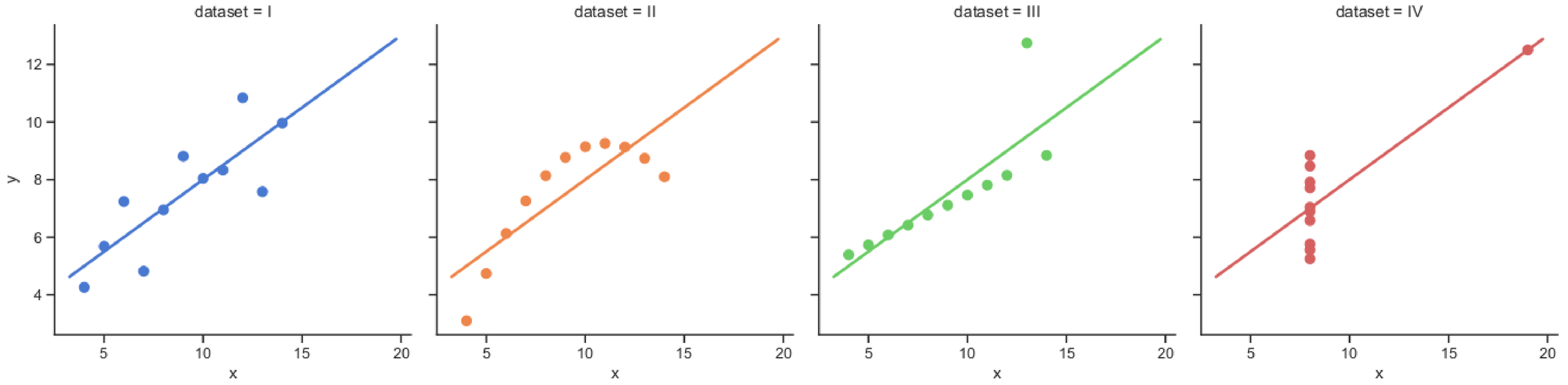
Source: Murphy, K. (2022). Probabilistic Machine Learning - An Introduction. MIT Press



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Importance of data visualization over summary statistics

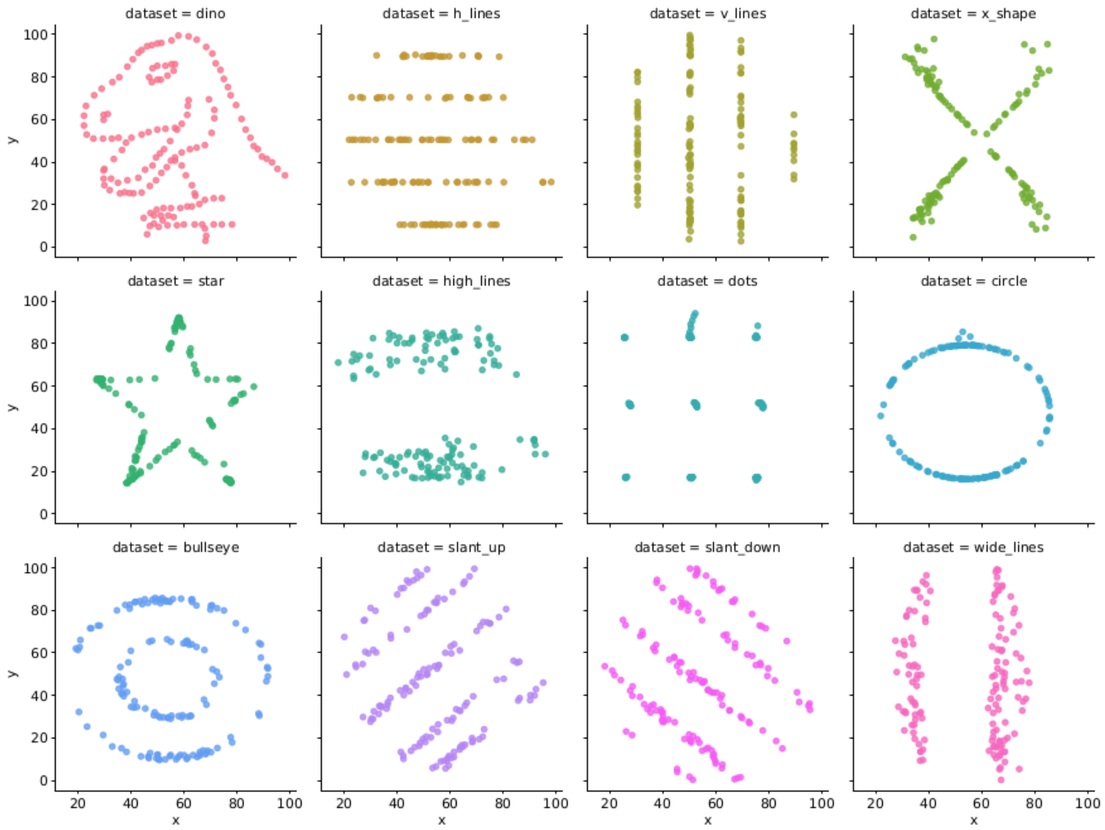
Source: Murphy, K. (2022). Probabilistic Machine Learning - An Introduction. MIT Press



Anscombe's quartet: All of these datasets have the same mean, variance and correlation coefficient.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Importance of data visualization over summary statistics



Source: Murphy, K. (2022). Probabilistic Machine Learning - An Introduction. MIT Press

Datasaurus dozen: All of these datasets have the same mean, variance and correlation coefficient.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Data Transformation

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Transformation

What is data transformation?

- Process of transforming data in a form **more** suitable for modeling
- Typically involves: Feature Scaling, Feature Selection, Feature Extraction, and Feature Construction (or Engineering)
- Sometimes: Data Augmentation, Data Sampling

Why is it important?

- ML algorithms have requirements (e.g., comp. complexity of some algorithms scales with number of data points)
- Model performance depends on data (e.g., different feature ranges affect many algorithms)

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Feature Scaling

Min-max normalization (or linear scaling):

- used when the feature is uniformly distributed in a fixed range
- typically scaled in the range [0,1]
- can be greatly affected by outliers

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Standardization (or z-score normalization):

- used when we don't know the feature range
- scaled feature distributions have mean=0 and sd=1
- less affected by outliers

$$x_{scaled} = \frac{x - mean}{sd}$$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Feature Selection

Process that chooses a subset of M features from the original set of N features (M < N), so that the feature space is optimally reduced according to a certain criterion.

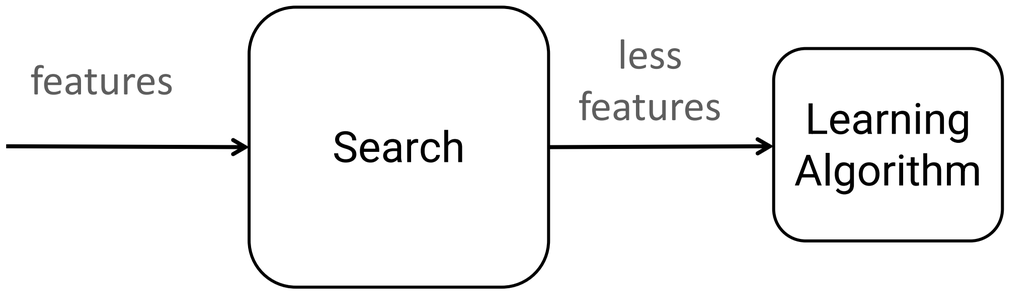
Role:

1. reduce dimensionality of feature space
2. speed up a learning algorithm
3. improve the predictive accuracy of a model
4. improve the comprehensibility of the results

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

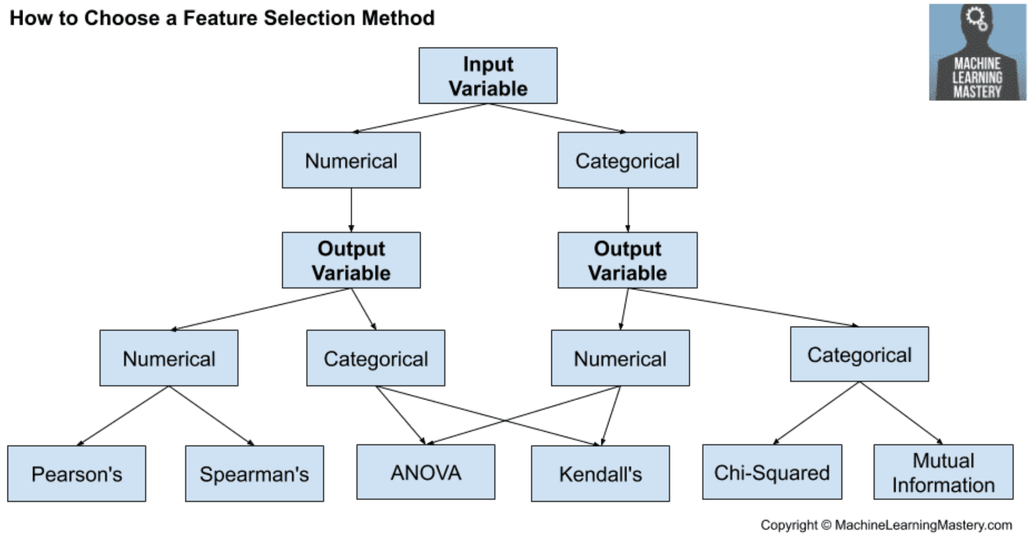
Feature Selection

Filter methods



Select subsets of features based on their **relationship with the target**

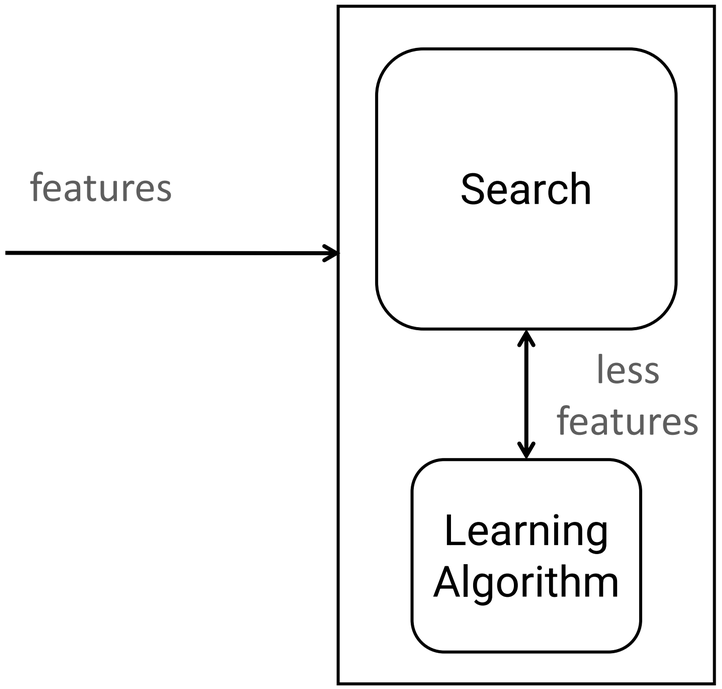
Typically by using statistical techniques



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Feature Selection

Wrapper methods



Search for **well-performing** subsets of features

Examples:

- Recursive Feature Elimination
- Heuristic search algorithms (e.g., genetic algorithms, particle swarm optimization)

Computational cost?

For each subset, a new model is trained and tested to obtain the accuracy

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Feature Selection

Intrinsic / implicit / embedded methods

Automatic feature selection during training

Examples:

- Rule-based models
- Tree-based models (will study in a later lecture)
- L1 regularization (will study in a later lecture)

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Feature Extraction (Dimensionality Reduction)

Process that extracts a set of M new features from the original N features (M < N) through some functional mapping.

Goal: search for a minimum set of new features via some transformation according to some performance measure.

Approaches:

- Principal Components Analysis (and other dimensionality reduction methods)
- Neural networks
- Will be studied in later lectures

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Feature Construction (Feature Engineering)

Process that discovers missing information about the relationships between features and augments the space of features by inferring or creating additional features.

Can be done using:

- Automated methods - examples:
  - Numerical features: polynomial expansion
  - Nominal features: conjunction, disjunction, negation
- Domain knowledge (e.g., SurfaceArea = Height x Width)

Goal: increase the expressive power of original features

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: Polynomial Features

```
In [1]: import numpy as np
   ...: from sklearn.preprocessing import PolynomialFeatures
   ...: X = np.arange(6).reshape(3, 2)
   ...: print(X)
   ...:
   ...: poly = PolynomialFeatures(2)
   ...: poly.fit_transform(X)
[[0 1]
 [2 3]
 [4 5]]
Out[1]:
array([[ 1.,  0.,  1.,  0.,  0.,  1.],
       [ 1.,  2.,  3.,  4.,  6.,  9.],
       [ 1.,  4.,  5., 16., 20., 25.]])
```

Degree-2 polynomial:
Input: [a,b]
Output: [1, a, b, a^2, ab, b^2].

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Quiz (T/F)

- Feature selection typically decreases the number of features

True

- Feature extraction typically increases the number of features

False

- Feature construction typically increases the number of features

True

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Augmentation (or Data Over-Sampling)

Technique that introduces additional data points (instances)

Used when we have:
- a **small dataset**
- an **imbalanced dataset** (e.g., 100 samples of class 0, 10 samples of class 1)

Approaches:
- naive random over-sampling (repeated data)
- add small Gaussian noise (for numerical features)
- SMOTE [1] and variants (for numerical features)

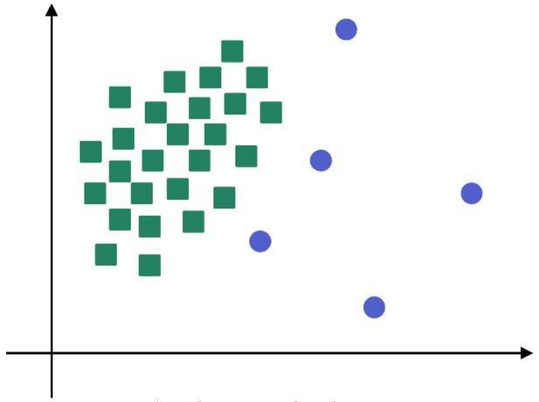
See imbalanced-learn library

[1] Nitesh et al. Smote: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16:321–357, 2002.

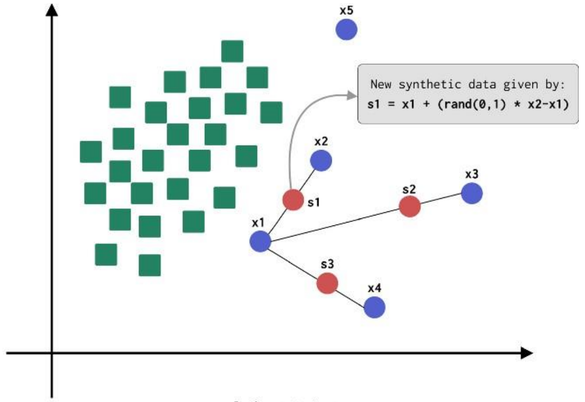
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

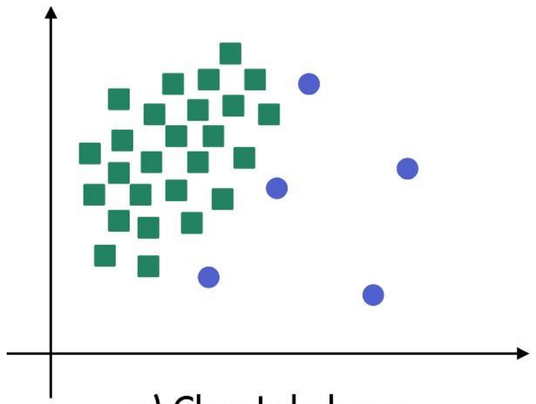
Data Augmentation (or Data Over-Sampling)



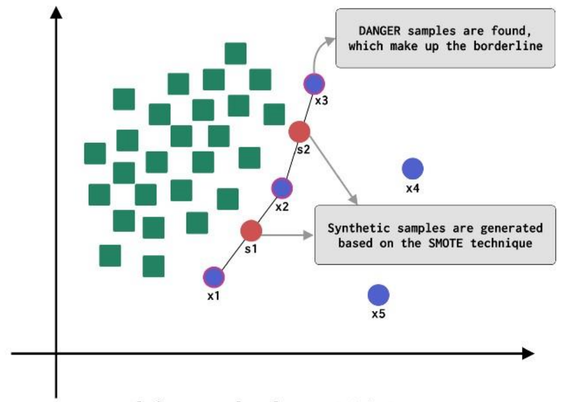
a) Class Imbalance



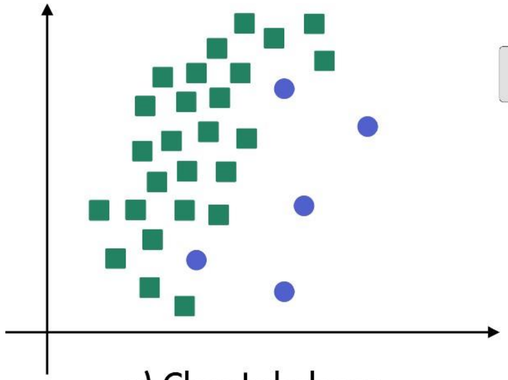
b) SMOTE



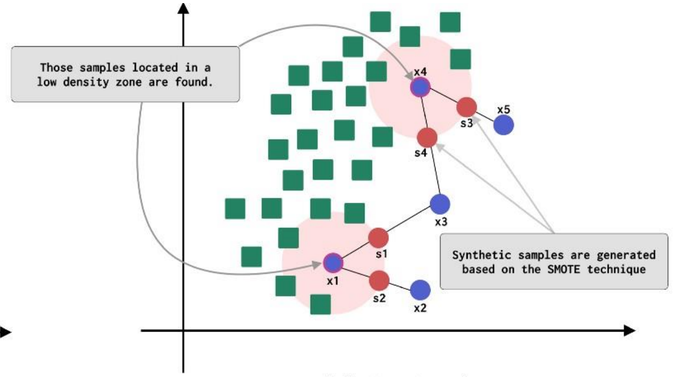
a) Class Imbalance



b) Borderline-SMOTE



a) Class Imbalance



b) ADASYN

Images source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Sampling (or Data Under-Sampling)

Technique that removes data points (instances)

Used when we have:

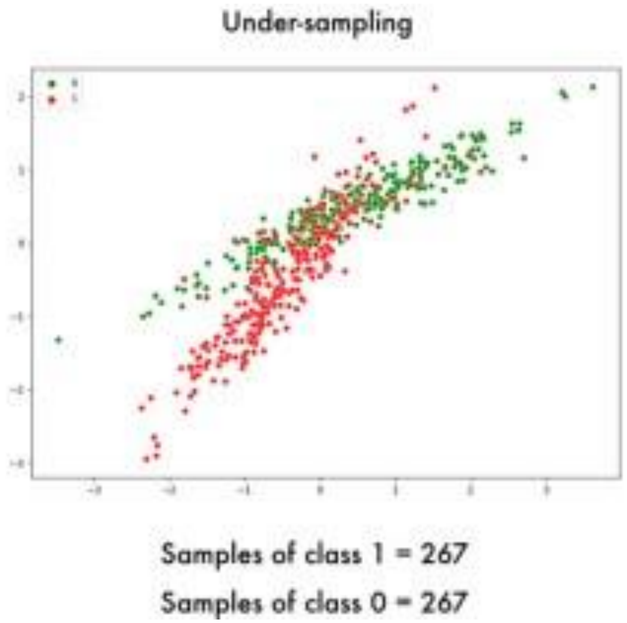
- a **huge dataset** (that cannot typically fit in memory)
- an **imbalanced dataset** (e.g., 1000 samples of class 0, 100 samples of class 1)

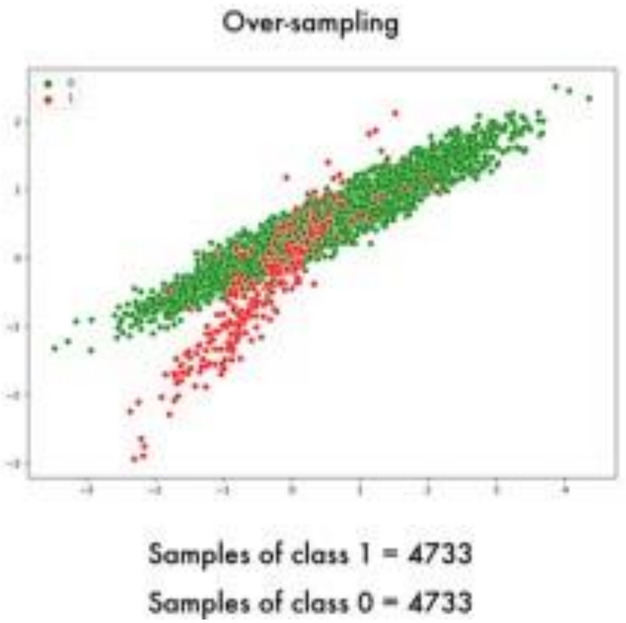
Approaches:

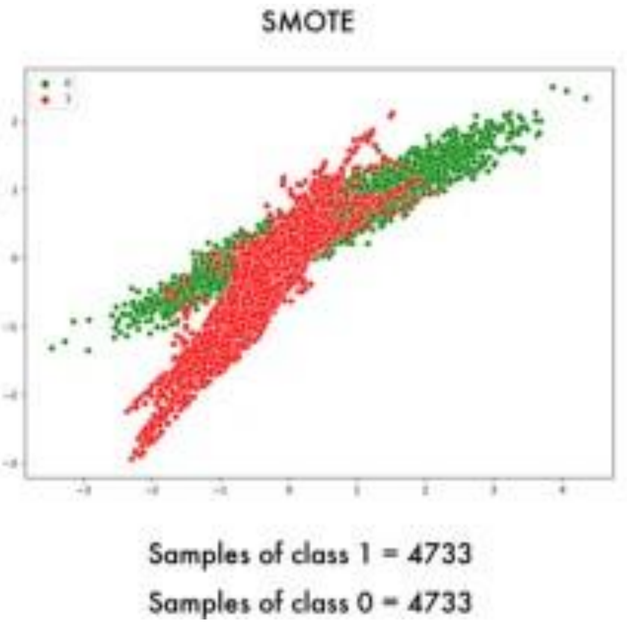
- simple random sampling
- cluster sampling

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Under-Sampling and Over-sampling







This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Dataset Splitting

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Dataset Splitting

70%

Used for fitting the model: improving its performance

30%

Used for checking the generalization performance of the model

Why do we need this?

Dataset is a sample from the underlying data distribution.

If we use the whole set we run into the risk of “overfitting” the model into the nuances of the sample.

More on this in later lectures...

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Common Pitfall: Data Leakage

Knowledge of the held-out (test) set is leaked when preparing the data

Typical example: feature scaling considers the whole dataset instead of only the training set

Can result in wrong model performance on new data

How to avoid it: fit the data transformation process on the training set, and only evaluate on the test set

The test set is not only used for evaluating the trained model, but also the data transformation process.

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Next Lecture

Part 2: Supervised Learning

- Regression
- Classification

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 3: Regression

Vassilis Vassiliades, PhD

Winter Semester 2022-2023

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Revision

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 3: Regression

Learning Outcomes

You will understand:

1. The problem of "regression" in supervised learning
2. The k-nearest neighbour regression model
3. The linear regression model
4. Gradient descent for finding the parameters of the linear regression model
5. The analytic solution to the linear regression model

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Supervised Learning

We present an input to the system: "teacher" provides the "target" output

System tries to match its predictions with the target ones

- Example: images of cats and dogs with their corresponding label


Cat


Dog

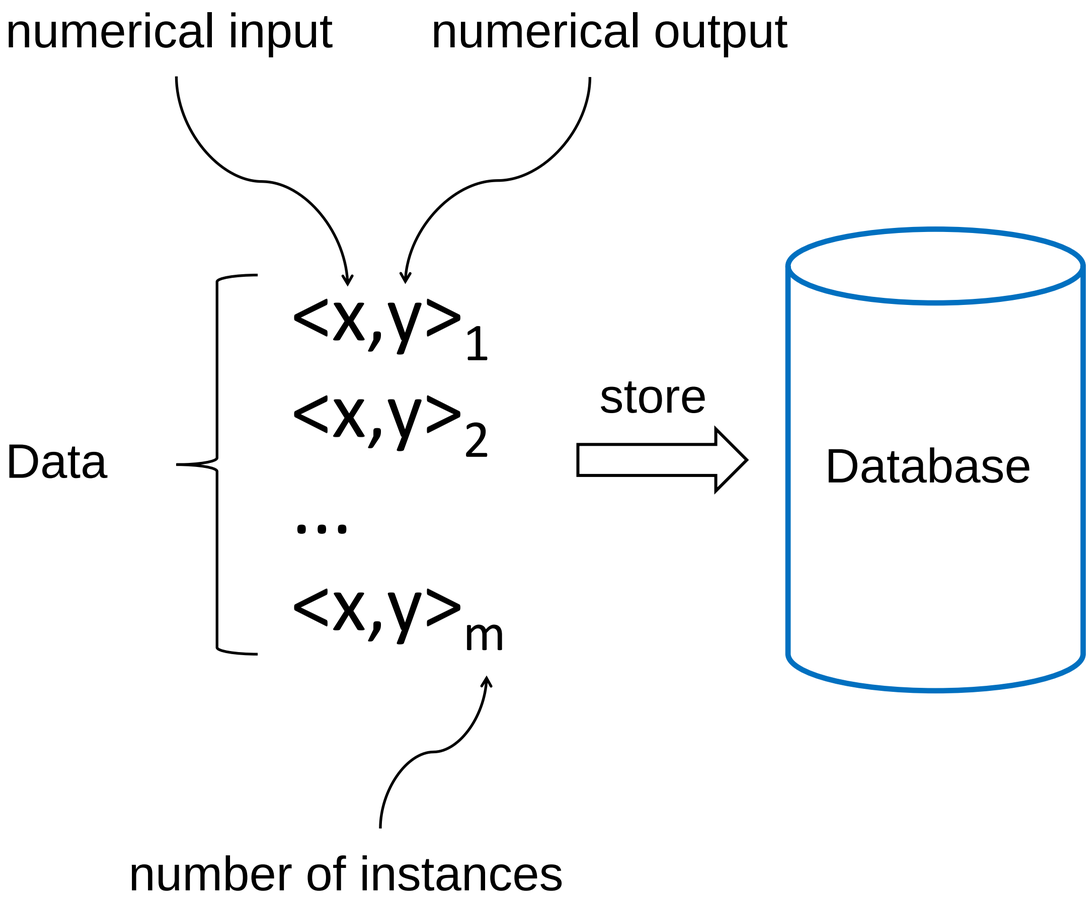
Goal: Use data to create a model that generalizes to unseen input-output pairs

Two common problems:

1. Regression - Output: continuous value
2. Classification - Output: discrete value

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression



What is the value of y given x?

y = l(x)

y = f [neighbors(x)]

Pros:
+ simple
+ fast
+ remembers

Cons:
- no generalization

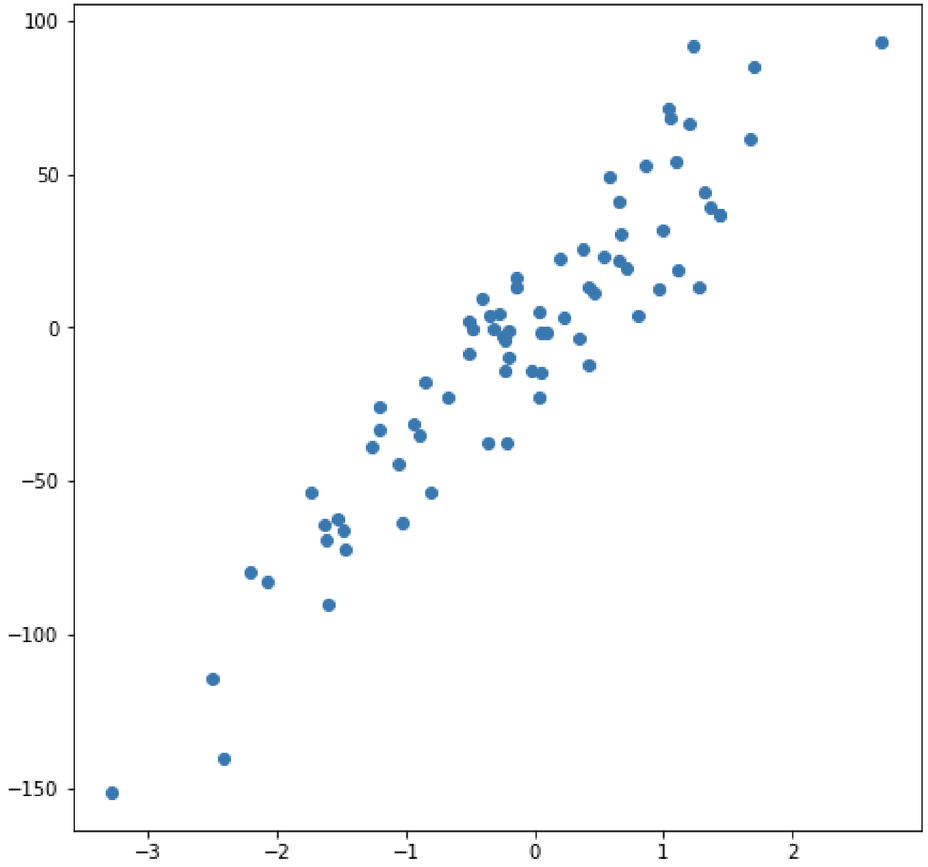
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

k-nearest neighbor regression

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

k-nearest neighbor regression

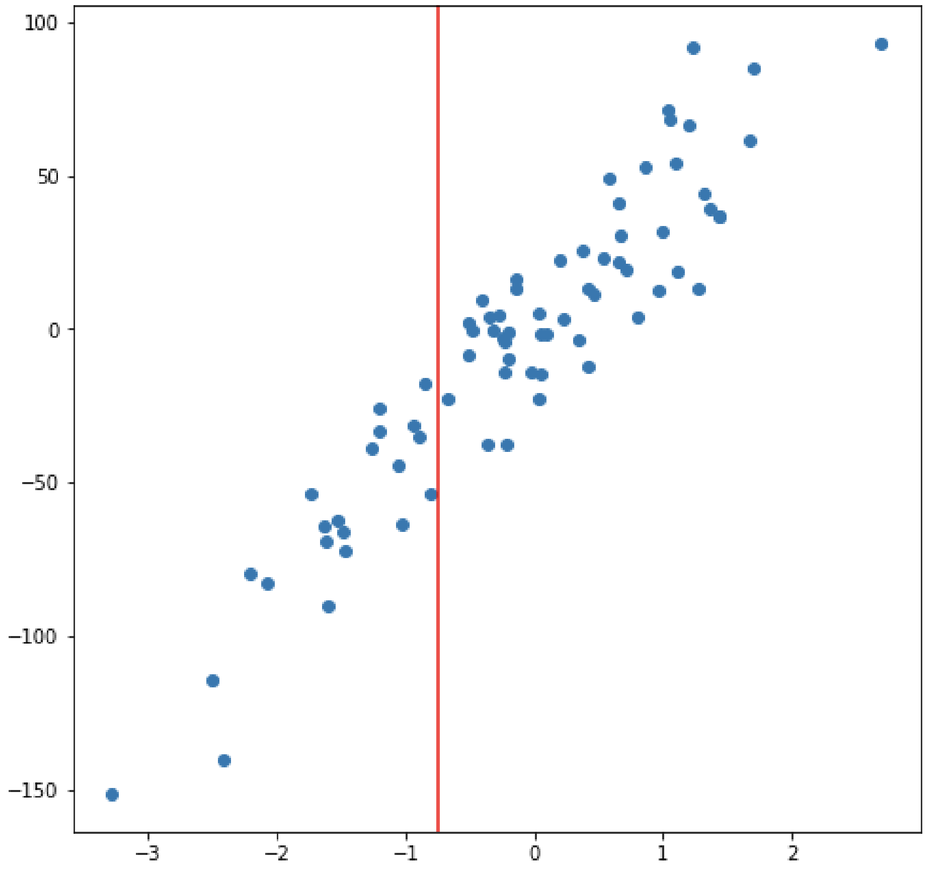
Source



This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

k-nearest neighbor regression

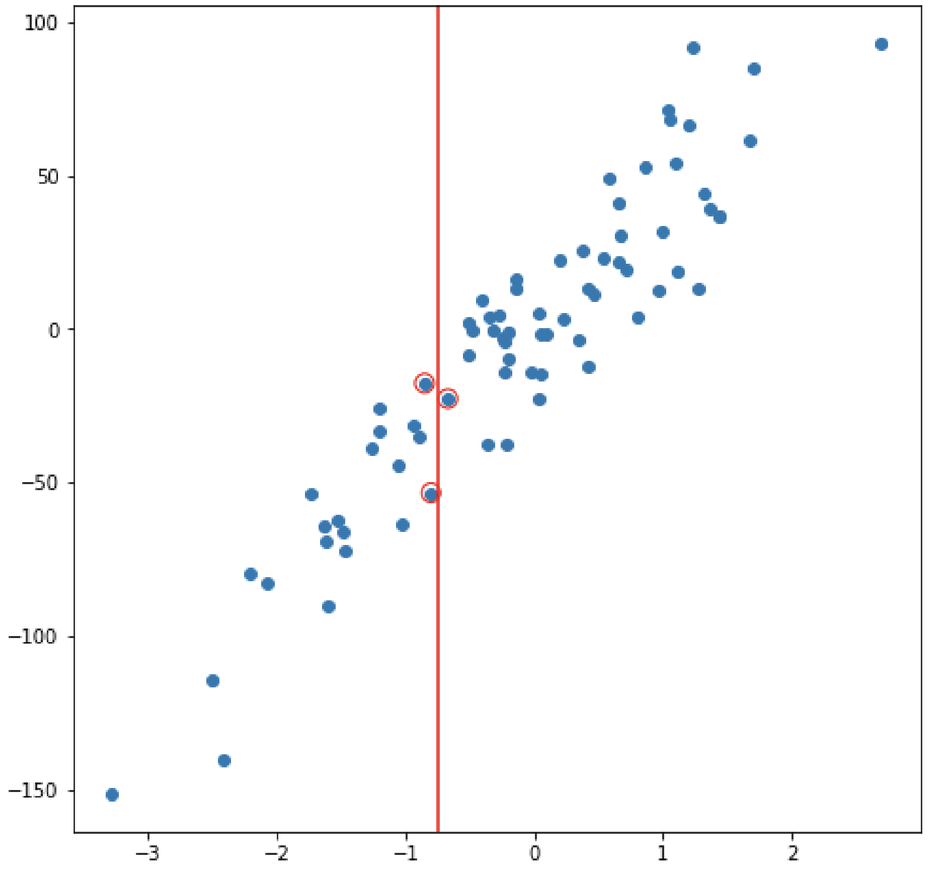
Source



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

k-nearest neighbor regression

Source

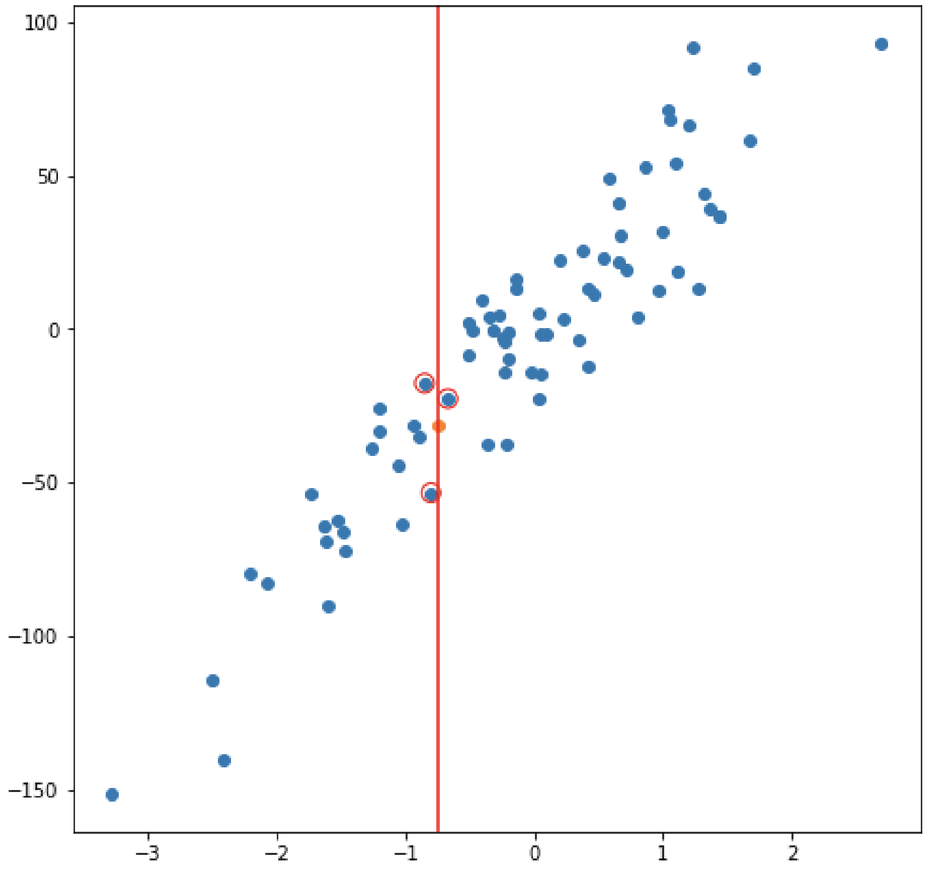


k = 3

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

k-nearest neighbor regression

Source

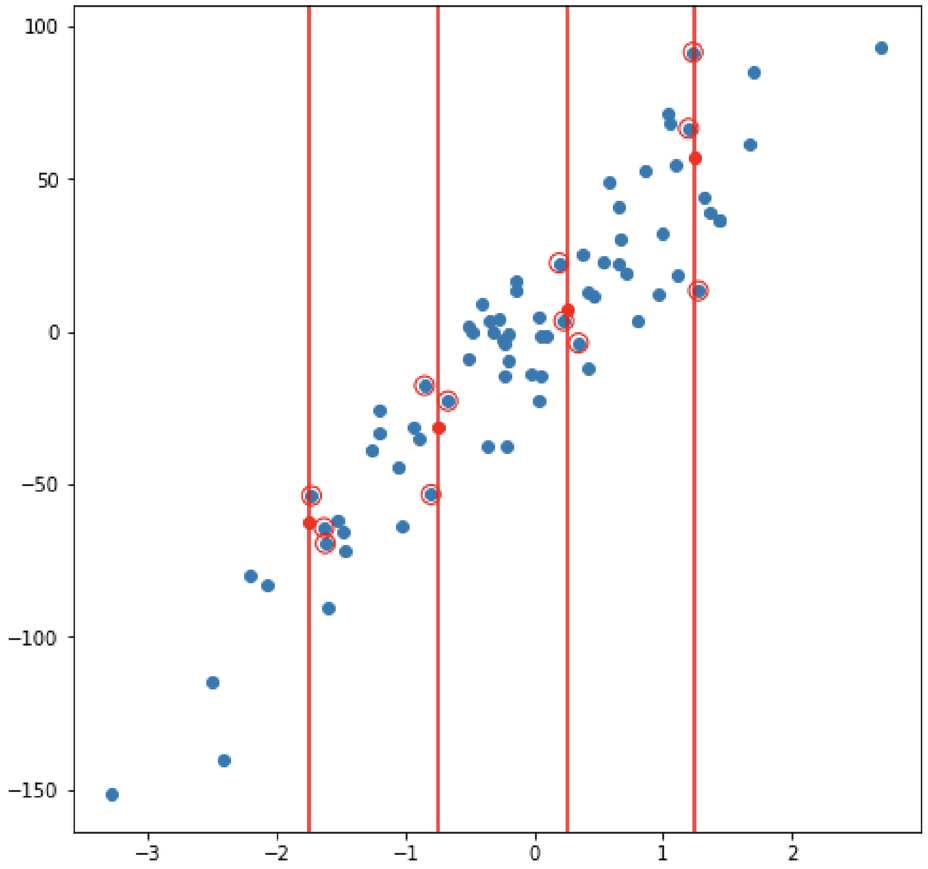


k = 3

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

k-nearest neighbor regression

Source

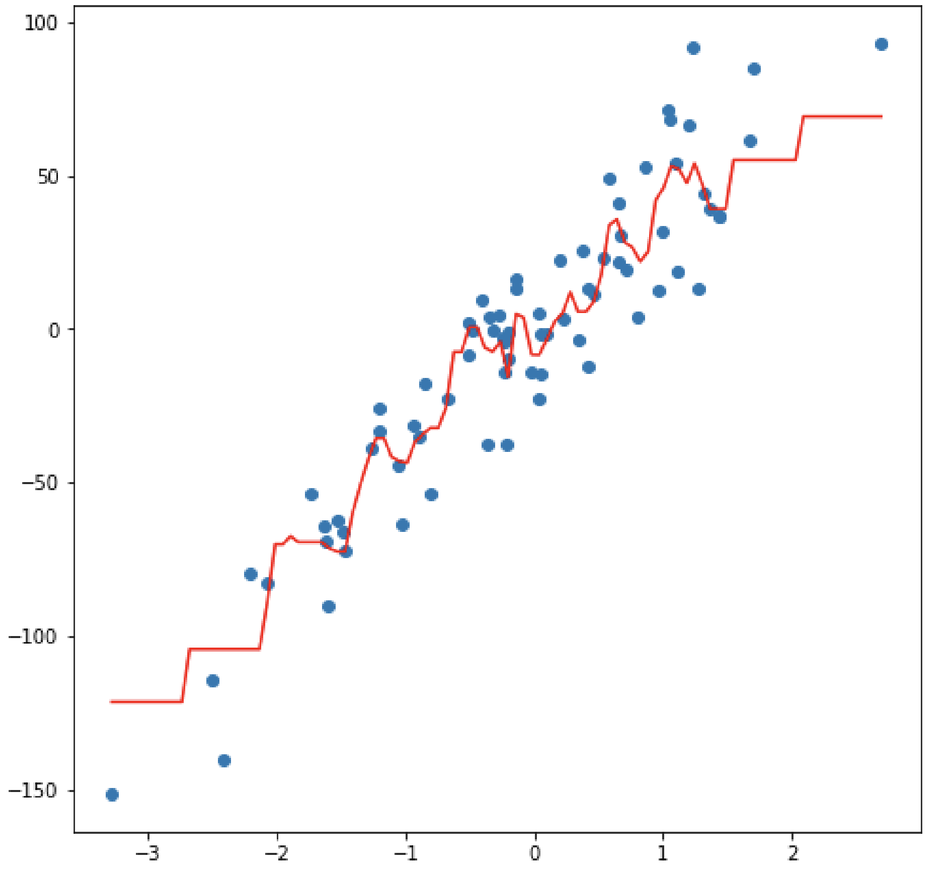


k = 3

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

k-nearest neighbor regression

Source

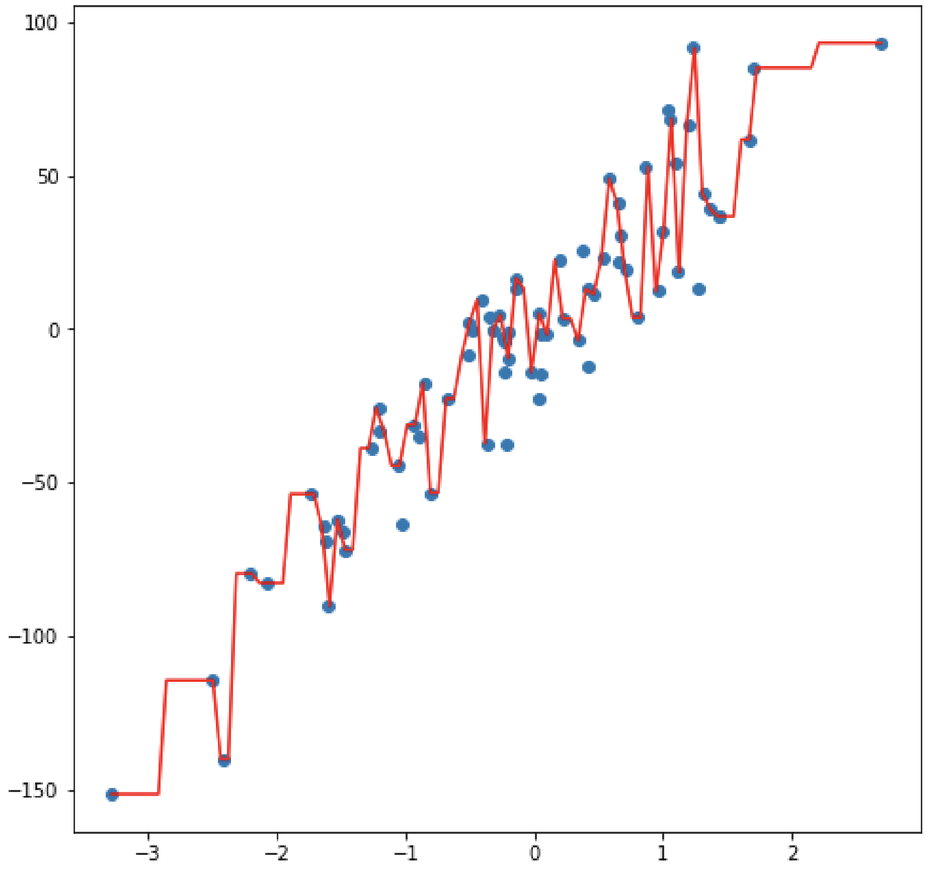


k = 3

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

k-nearest neighbor regression

Source



k = 1

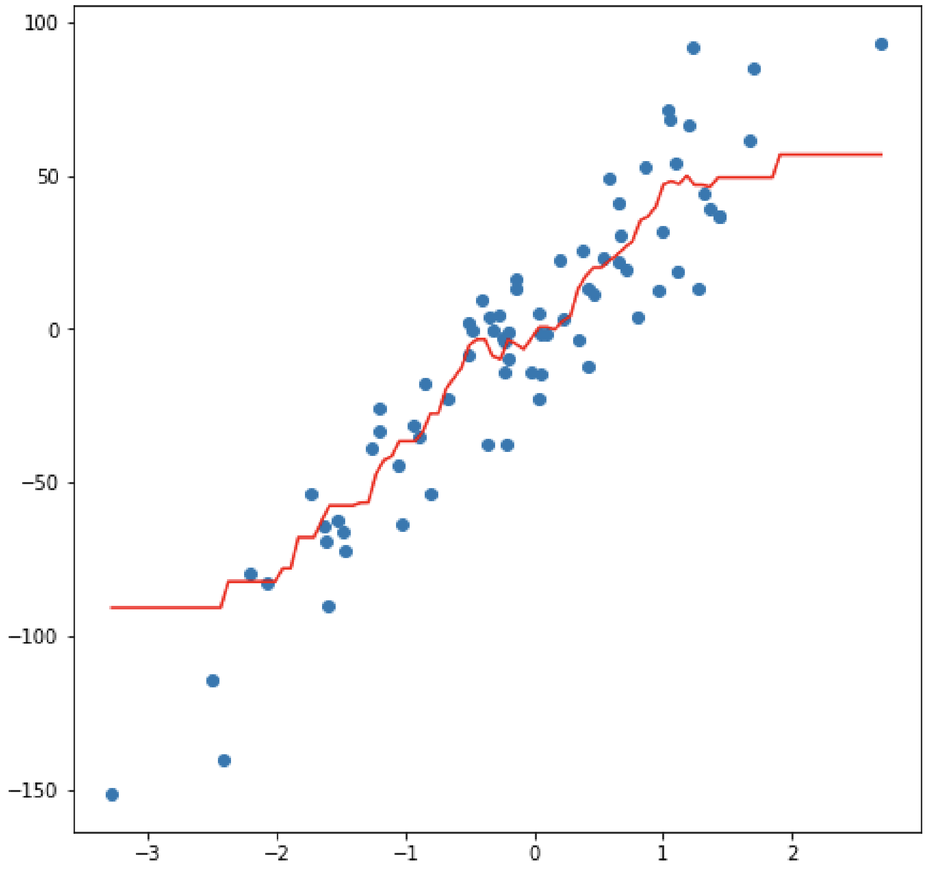
Model captures the nuances and noise instead of the trend

This is known as overfitting

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

k-nearest neighbor regression

Source



k = 10

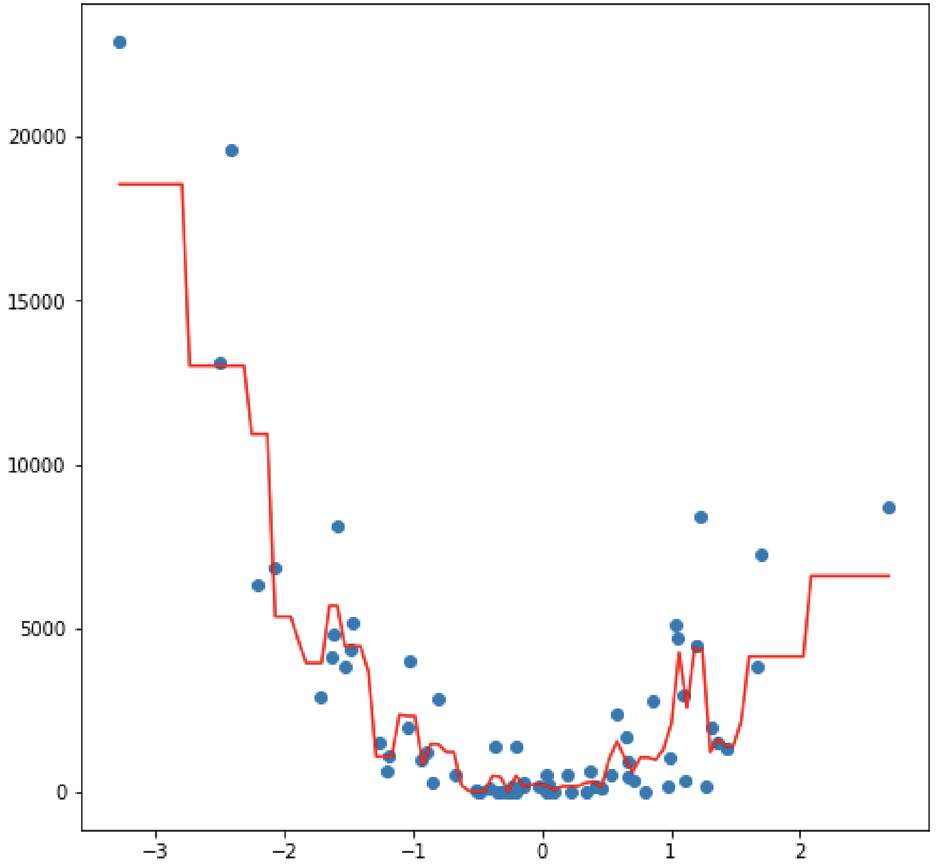
Smoother prediction, but problem at the edges

What if k=m?

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

k-nearest neighbor regression

Source



k = 3

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

k-nearest neighbor regression

Strengths

- easy to implement
- no training time
- handles non-linearities

Weaknesses

- prediction becomes slower as the dataset becomes bigger
- does not work well in high dimensions
- needs feature scaling
- sensitive to noisy data or outliers
- lacks interpretability

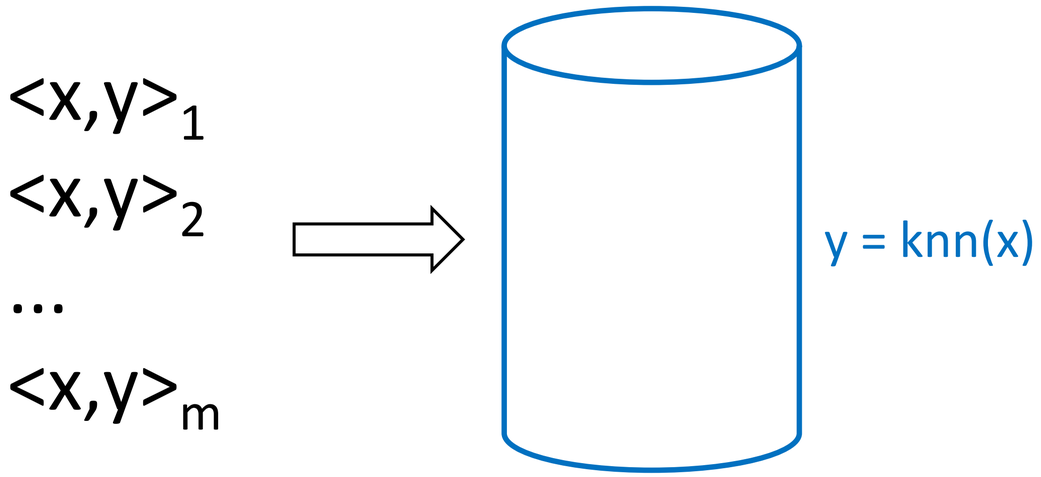
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example

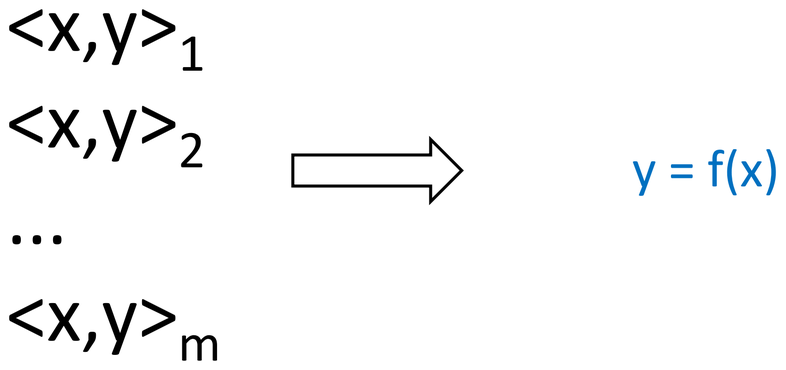
```
In [1]: X = [[0], [1], [2], [3]]
   ...: y = [0, 0, 1, 1]

In [2]: from sklearn.neighbors import KNeighborsRegressor
   ...: neigh = KNeighborsRegressor(n_neighbors=2)
   ...: neigh.fit(X, y)
Out[2]:
KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski',
          metric_params=None, n_jobs=None, n_neighbors=2, p=2,
          weights='uniform')

In [3]: print(neigh.predict([[1.5]]))
[0.5]
```


This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Regression



Alternative approach

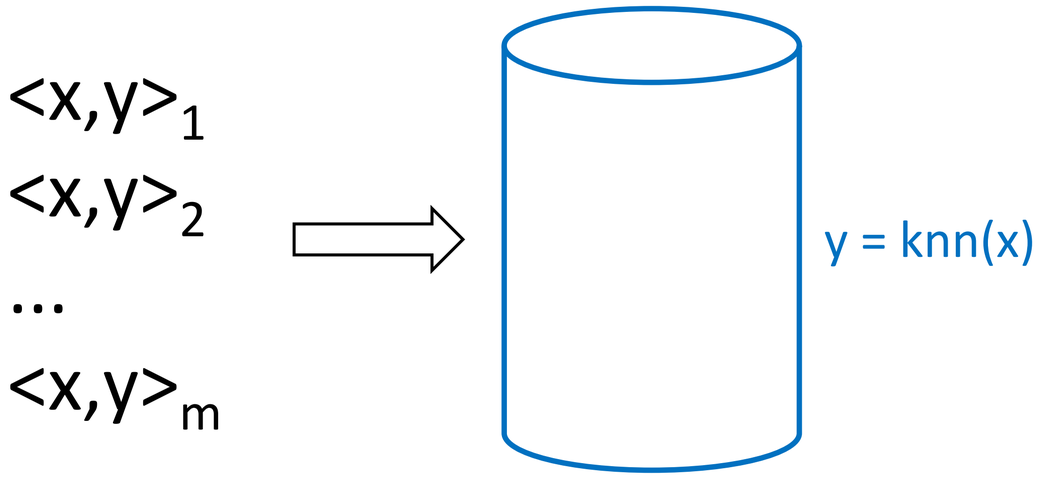


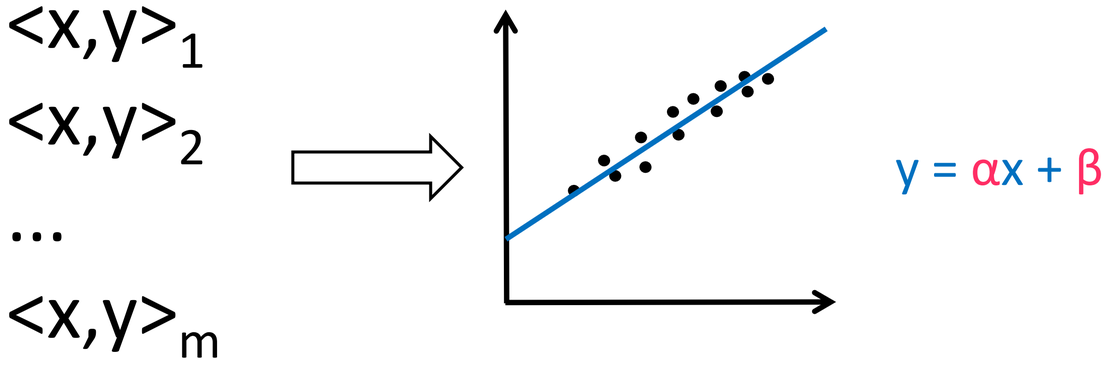
What is the simplest function we can use?

Instance-based learning / memory-based learning
or
Non-parametric models / Exemplar-based models

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression



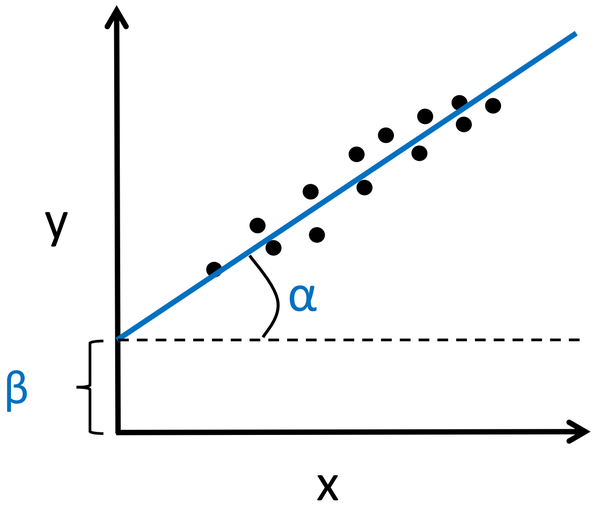


Instance-based learning / memory-based learning
or
Non-parametric models / Exemplar-based models

Model-based learning
or
Parametric models

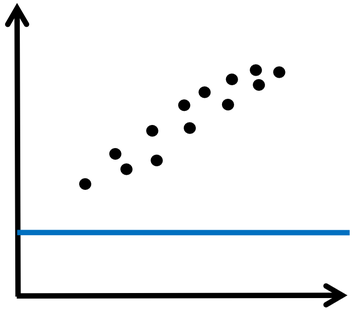
This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Linear Regression

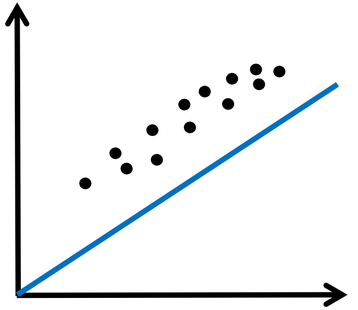


$y = \alpha x + \beta$

$\alpha=0$
$y=\beta$



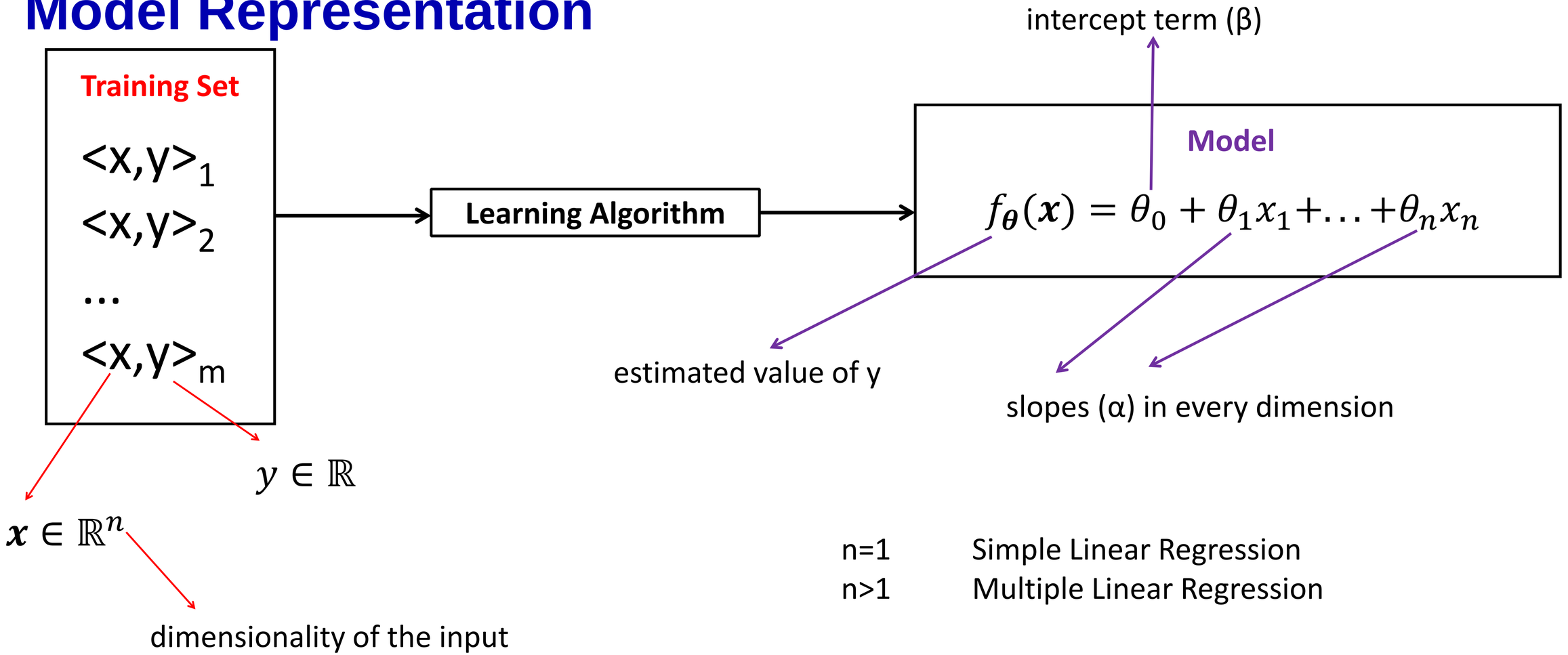
$\beta=0$
$y=\alpha x$



Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model Representation



n=1	Simple Linear Regression
n>1	Multiple Linear Regression

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model Representation

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 + \theta_1 x_1 + \ldots + \theta_n x_n$$

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 x_0 + \theta_1 x_1 + \ldots + \theta_n x_n \qquad \text{where } x_0 = 1, \text{ so now } \boldsymbol{x} \in \mathbb{R}^{n+1}$$

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^T \boldsymbol{x}$$

$$[\theta_0 \quad \theta_1 \quad \ldots \quad \theta_n] \cdot \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} = f_{\boldsymbol{\theta}}(\boldsymbol{x})$$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Objective

Choose $\theta_0, \ldots, \theta_n$ so that $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ is close to y for all our training examples $< \boldsymbol{x}, y >_{1:m}$

How?

A common approach is to use the method of **Least Squares**

- Minimize the sum of squares of residuals
- **Residual**: difference between the actual value (y) and the estimated ($f_{\boldsymbol{\theta}}(\boldsymbol{x})$ fitted by the model)

$$(f_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)})^2$$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Objective & Cost function

Choose $\theta_0, \ldots, \theta_n$ so that $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ is close to y for all our training examples $< \boldsymbol{x}, y >_{1:m}$

$$\underset{\boldsymbol{\theta}}{minimize}\ L(\boldsymbol{\theta})$$

$$L(\boldsymbol{\theta}) = \frac{1}{2m}\sum_{i=1}^{m}(f_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)})^2$$

Mean Squared Error function

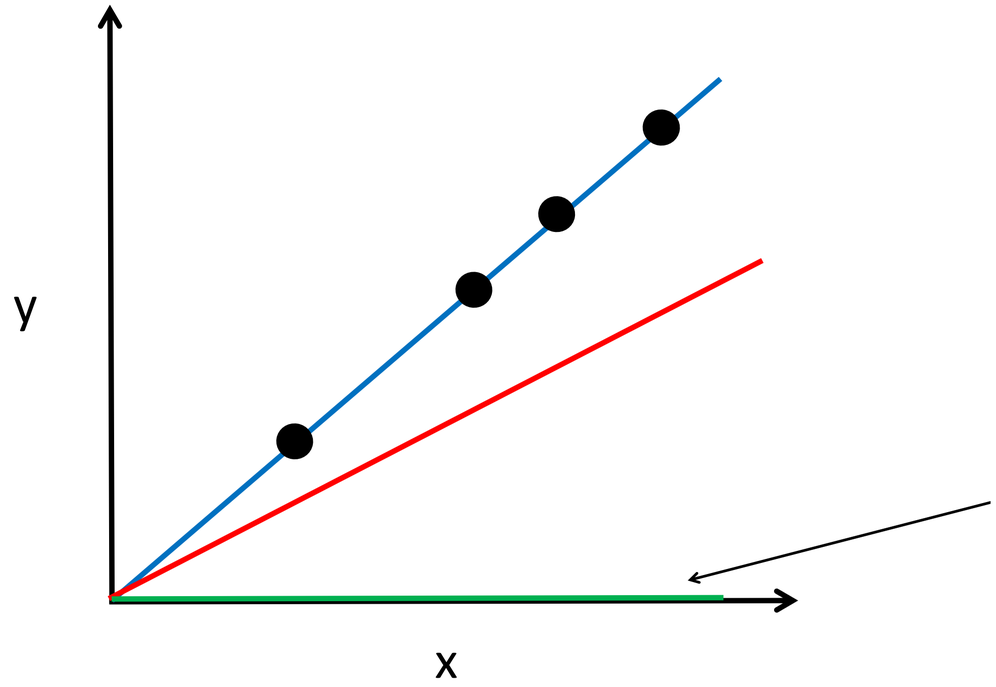
Can the error be < 0?

Co-financed by the European Union
Connecting Europe Facility

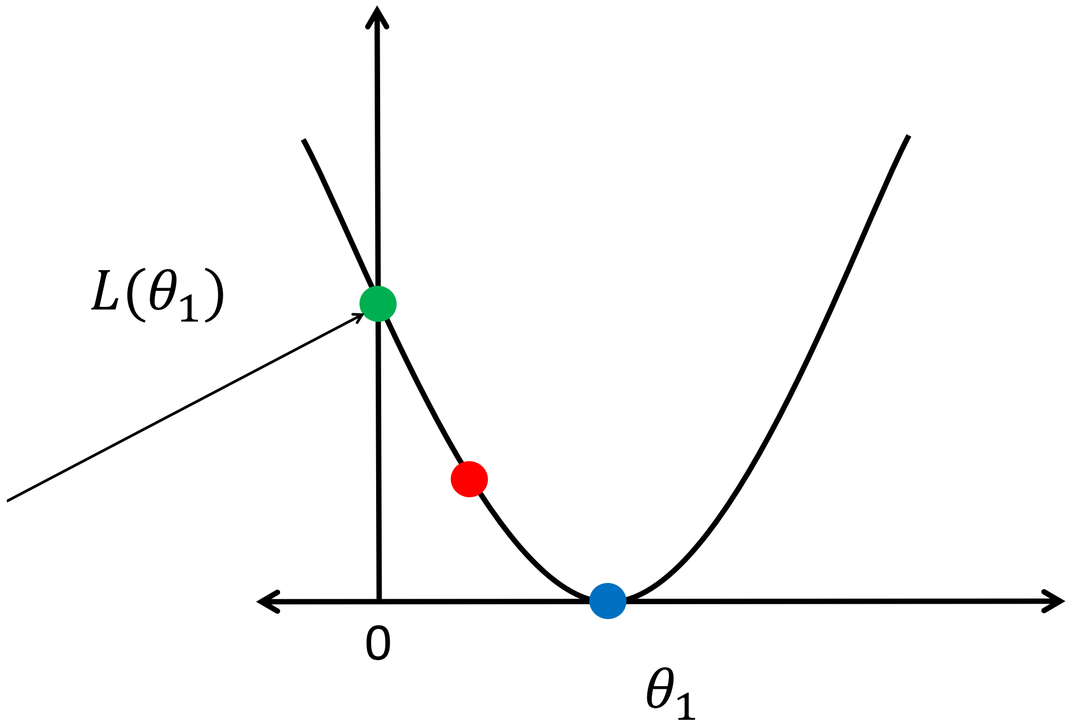
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Cost function

$f_{\theta_1}(x)$



$L(\theta_1)$



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

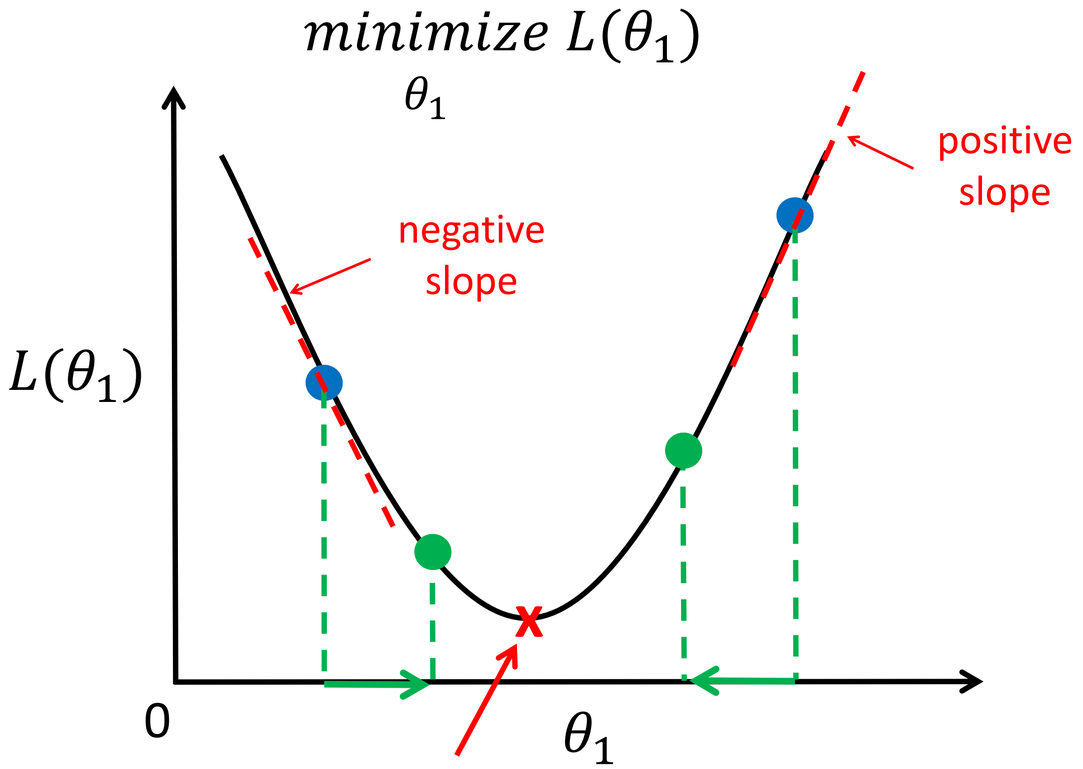
Master programmes in Artificial Intelligence 4 Careers in Europe

Gradient Descent

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Optimization: Gradient Descent



This is where we want to end up (local minimum)

1. Start from random θ_1
2. Repeat until convergence

change θ_1 so as to move downhill

$$\theta_1 = \theta_1 - \alpha \frac{dL(\theta_1)}{d\theta_1} \qquad \alpha \in [0,1]$$

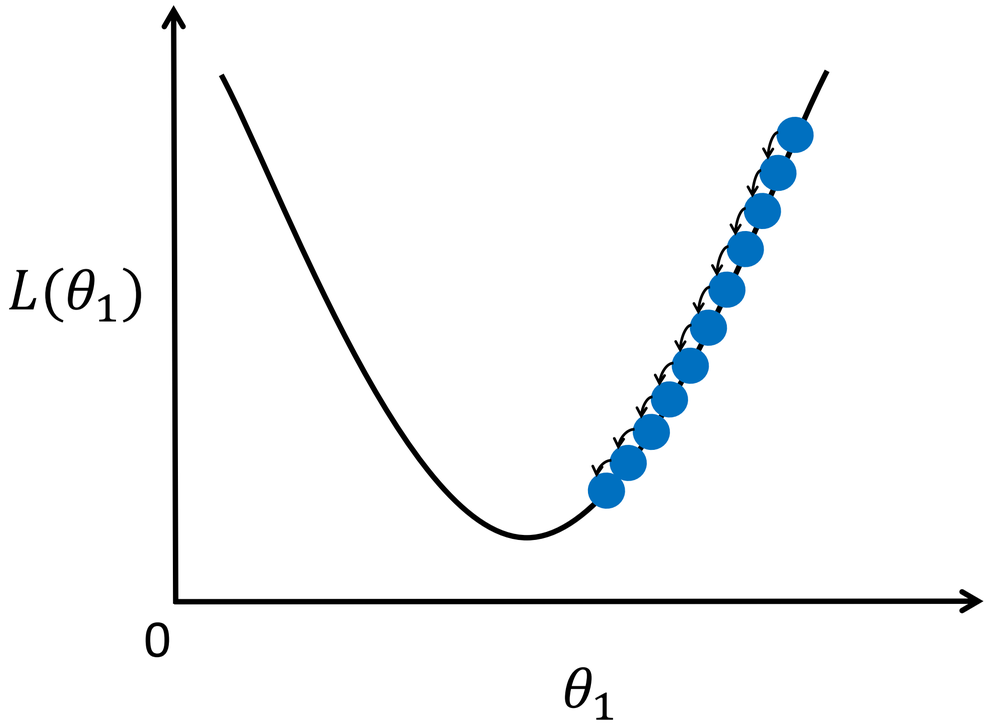
$\frac{dL(\theta_1)}{d\theta_1} > 0$ decreases θ_1

$\frac{dL(\theta_1)}{d\theta_1} < 0$ increases θ_1

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

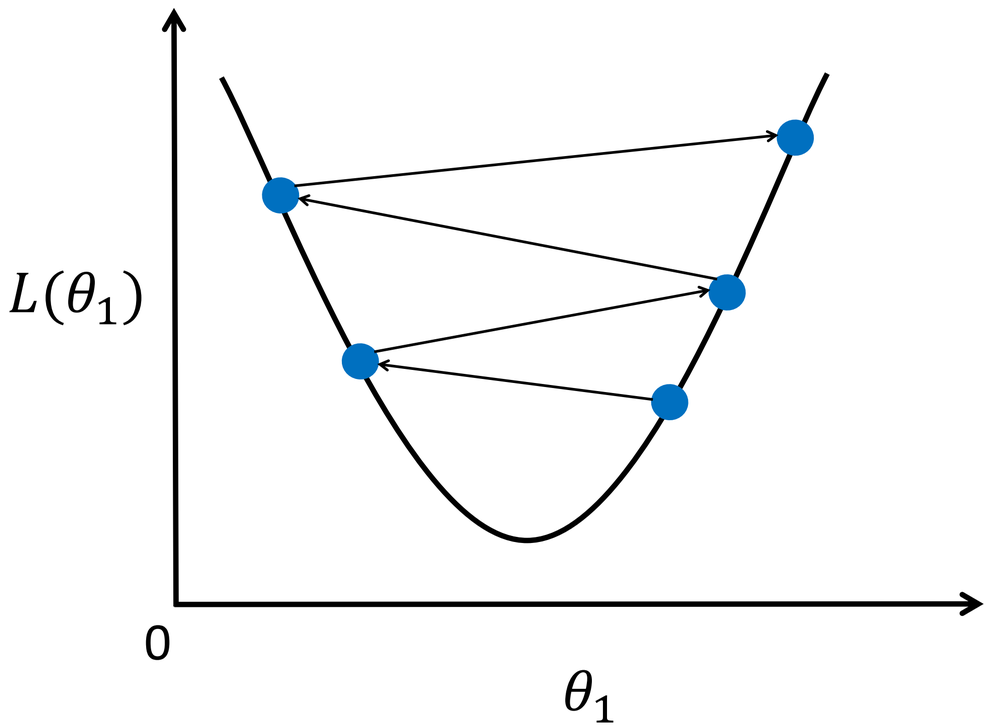
Gradient Descent: learning rate α

α too small



slow convergence

α too large



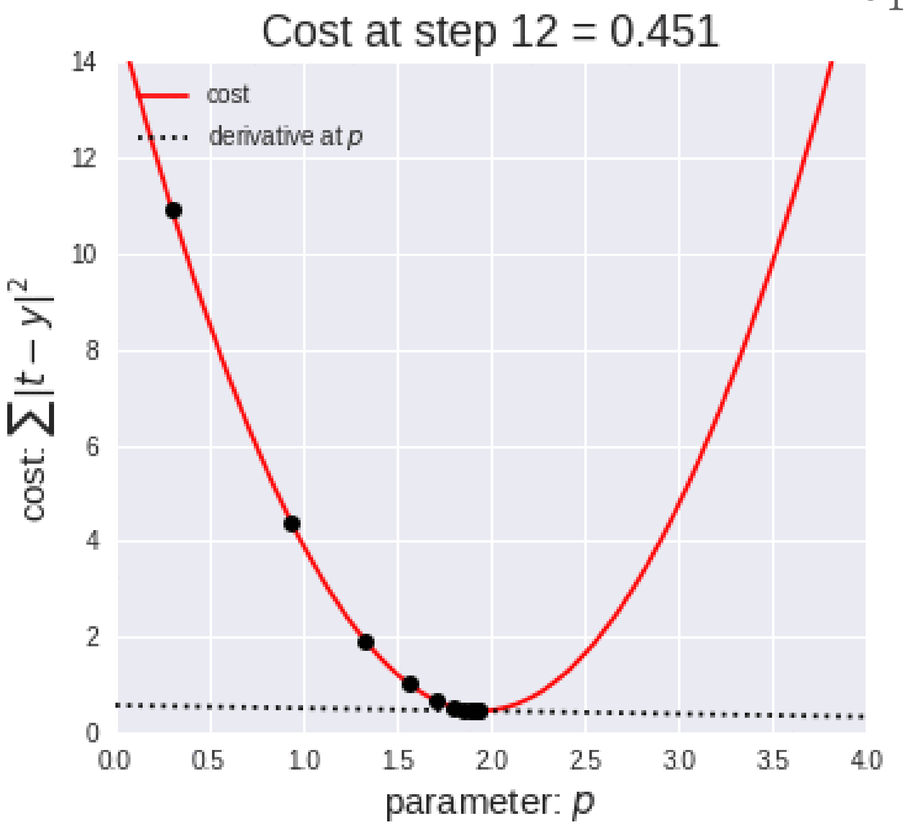
possible divergence

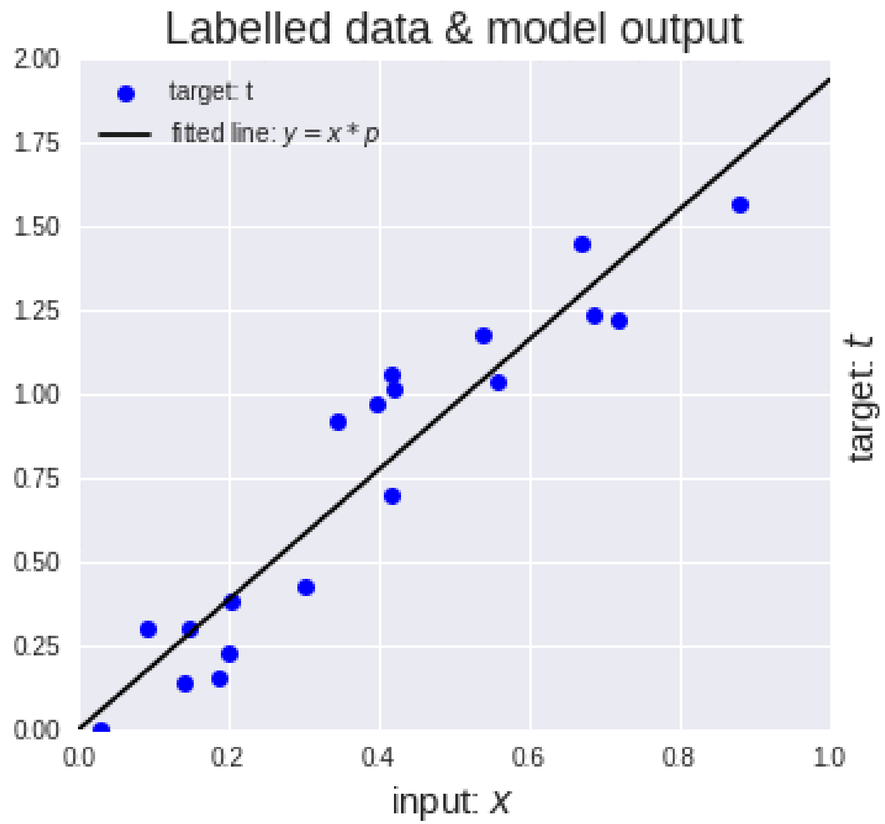
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gradient Descent

$$\underset{\theta_1}{minimize}\ L(\theta_1)$$



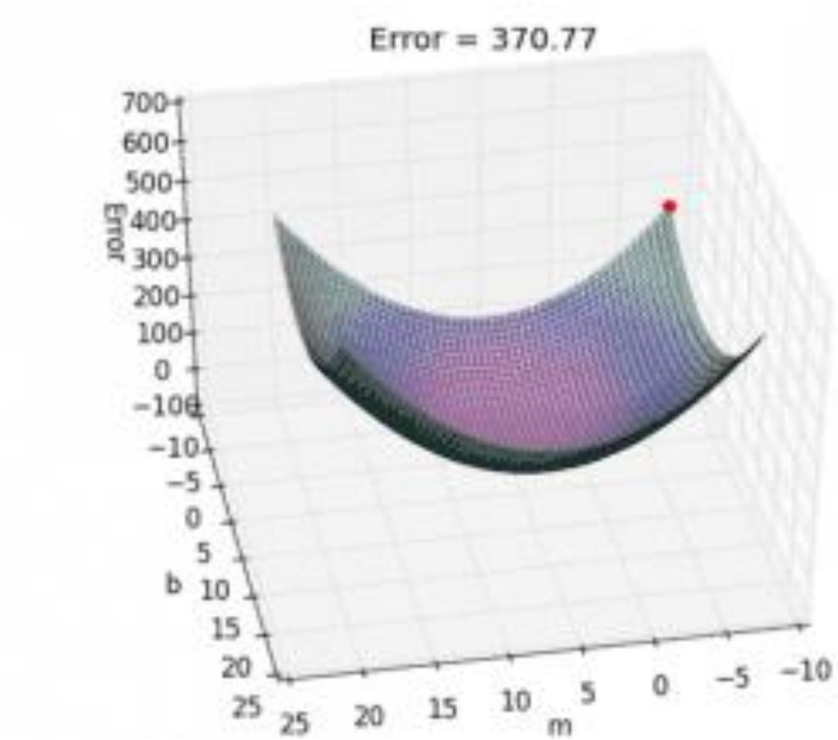


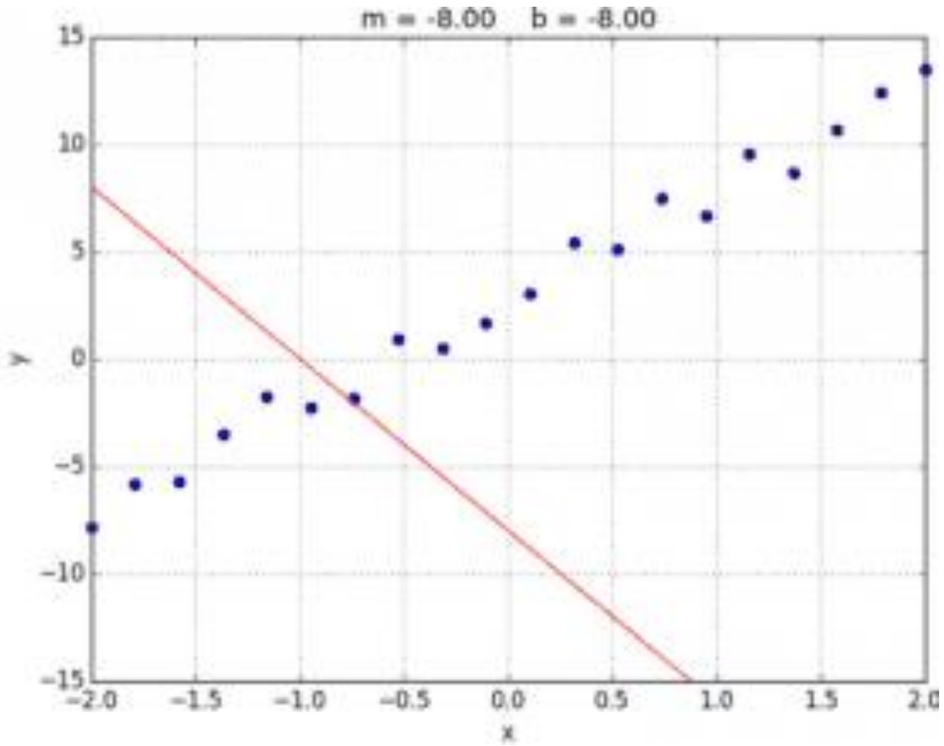
source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gradient Descent

$$\underset{\theta_0,\theta_1}{minimize}\ L(\theta_0,\theta_1)$$





source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gradient Descent in Practice

$$f_{\theta_0,\theta_1}(x) = \theta_0 + \theta_1 x$$

Implementation:

$||\nabla L|| < \varepsilon$

Repeat until convergence:

1. Calculate loss $L(\theta_0, \theta_1)$

$$L(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(f_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)})^2$$

2. Calculate gradient vector

$$\begin{bmatrix} \frac{\partial L(\theta_0,\theta_1)}{\partial \theta_0} \\ \frac{\partial L(\theta_0,\theta_1)}{\partial \theta_1} \end{bmatrix}$$

$$\frac{1}{m}\sum_{i=1}^{m}(f_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)})$$

$$\frac{1}{m}\sum_{i=1}^{m}(f_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)})x^{(i)}$$

3. Update parameter vector

$$\begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} = \begin{bmatrix} \theta_0 - \alpha \frac{\partial L(\theta_0,\theta_1)}{\partial \theta_0} \\ \theta_1 - \alpha \frac{\partial L(\theta_0,\theta_1)}{\partial \theta_1} \end{bmatrix}$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gradient Descent in Practice

"Batch" gradient descent:

- Uses all training examples (m)
- Used when the size of the training set is small (e.g., m = 10^1 – 10^3)

Other variants of gradient descent:

- Stochastic Gradient Descent
- Mini-batch Gradient Descent
- will look in later lectures

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gradient Descent in Practice

Use Feature Scaling

Make sure features are on a similar scale

Example:

x_1 = size (0 – 2000 feet^2)

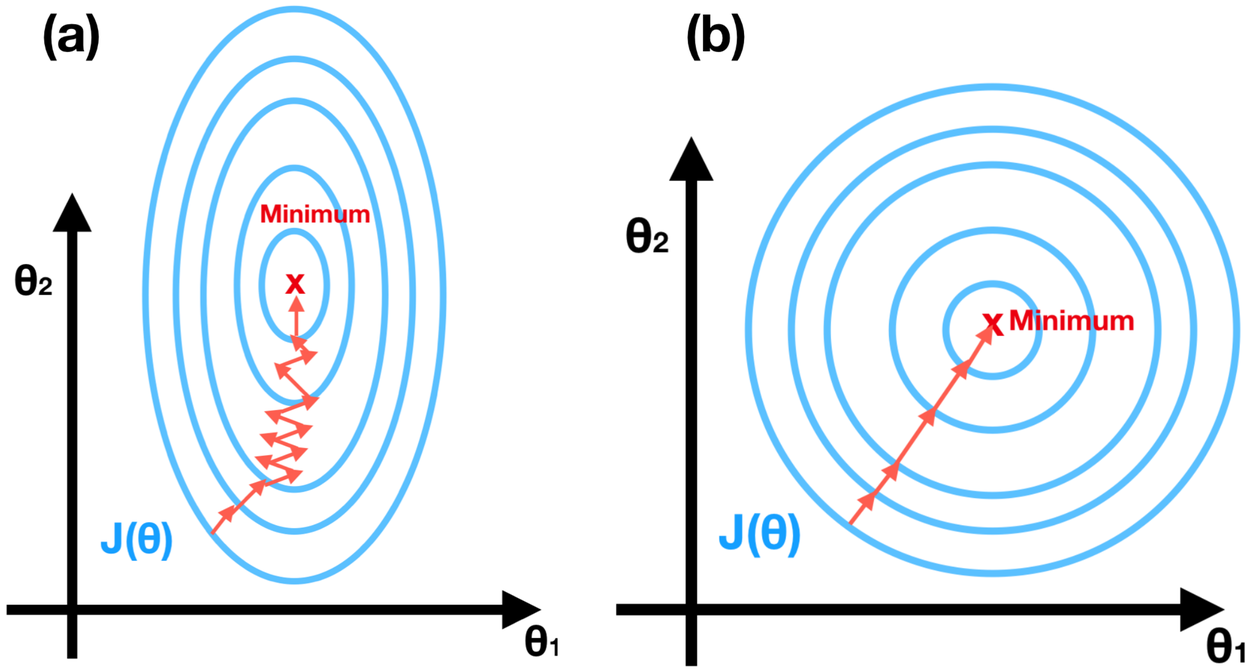
x_2 = number of bedrooms (1-5)

$new\_x_1 = x_1 / 2000$ $\quad$ $new\_x_1 \in [0,1]$

$new\_x_2 = x_2 / 5$ $\quad$ $new\_x_2 \in [0,1]$

Without feature scaling

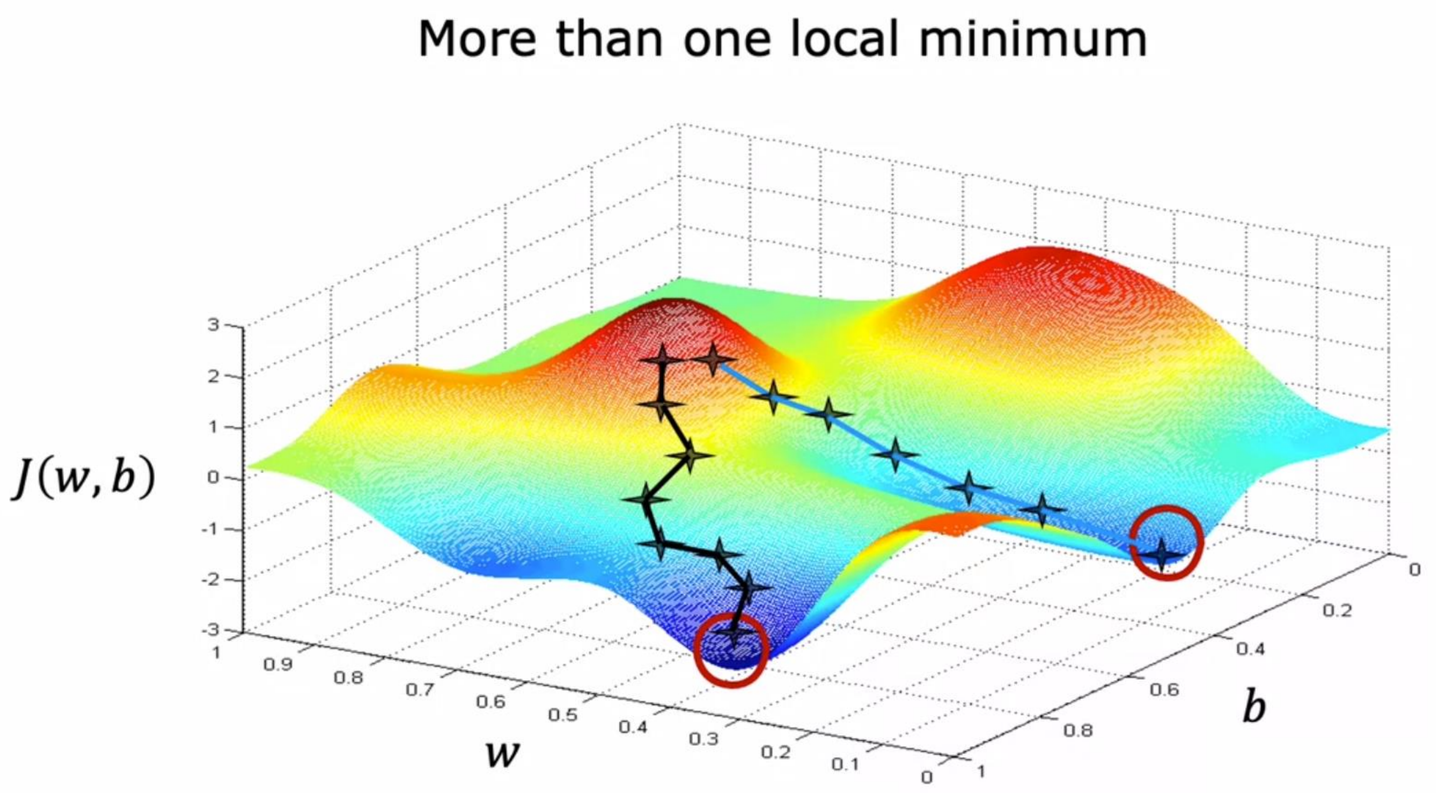
With feature scaling

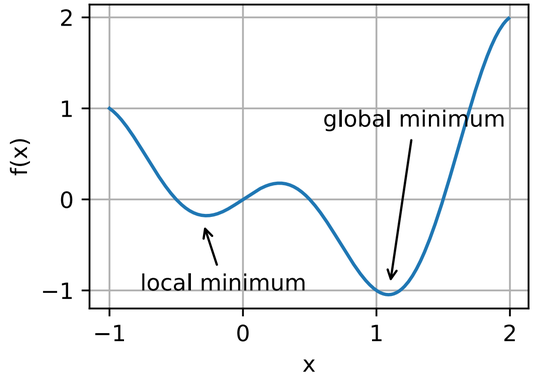


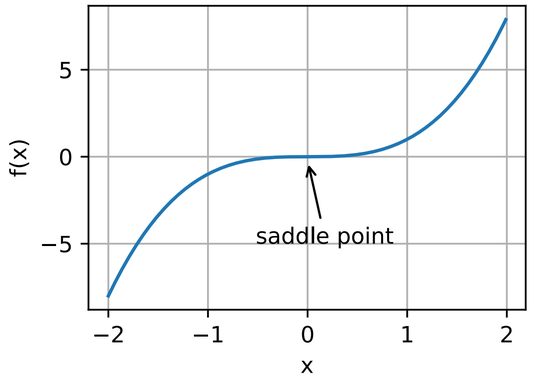
source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gradient Descent on a non-convex 2D function







Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gradient Descent on the Mean Squared Error

Does the **MSE** when using the **Linear Regression model** have more than one minima?

- No – a single global minimum

For this particular case, there is an analytic (closed-form / non-iterative) solution.

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Analytic Solution: Normal Equations

$$\theta_0 + \theta_1 x_1 + \ldots + \theta_n x_n \approx y$$

x	y
$x^{(1)}$	$y^{(1)}$
$x^{(2)}$	$y^{(2)}$
...	...
$x^{(m)}$	$y^{(m)}$

$$\underbrace{\begin{bmatrix} 1 & x_1^{(1)} & \ldots & x_n^{(1)} \\ 1 & x_1^{(2)} & \ldots & x_n^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(m)} & \ldots & x_n^{(m)} \end{bmatrix}}_{X} \underbrace{\begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}}_{\theta} \approx \underbrace{\begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}}_{y}$$

$$\theta = (X^T X)^{-1} X^T y$$

Important: $m > n$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example

```
In [1]: import numpy as np
   ...: from sklearn.linear_model import LinearRegression
   ...: X = np.array([[1, 1], [1, 2], [2, 2], [2, 3]])

In [2]: # y = 1 * x_0 + 2 * x_1 + 3
   ...: y = np.dot(X, np.array([1, 2])) + 3
   ...: reg = LinearRegression().fit(X, y)

In [3]: reg.score(X, y)
Out[3]: 1.0

In [4]: reg.coef_
Out[4]: array([1., 2.])

In [5]: reg.intercept_
Out[5]: 3.0000000000000018

In [6]: reg.predict(np.array([[3, 5]]))
Out[6]: array([16.])
```


This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Linear regression

Strengths

- easy to implement (analytic solution)
- small computation time for fitting (analytic solution)
- constant prediction time (does not save any data points)
- analytic solution does not need feature scaling
- less sensitive to noisy data
- more interpretable

Weaknesses

- does not handle nonlinear relationships in the data
- for large datasets it requires iterative methods, e.g., gradient descent
- gradient descent needs feature scaling

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Linear regression

Strengths

- easy to implement (analytic solution)
- small computation time for fitting (analytic solution)
- constant prediction time (does not save any data points)
- analytic solution does not need feature scaling
- less sensitive to noisy data
- more interpretable

Weaknesses

- does not handle nonlinear relationships in the data
- for large datasets it requires iterative methods, e.g., gradient descent
- gradient descent needs feature scaling

How can we fix this?
… without changing the algorithm?

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Polynomial regression

1. Use feature engineering to construct polynomial features

   e.g. for 2 variables
   and polynomial degree=2

   $$x_1, x_2, x_1^2, x_2^2, x_1 x_2$$

2. Find the solution to the new linear regression problem

$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2 \approx y$$

$$\boxed{\theta = (X^T X)^{-1} X^T y}$$

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 4: Classification

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 4: Classification

Learning Outcomes

You will understand:

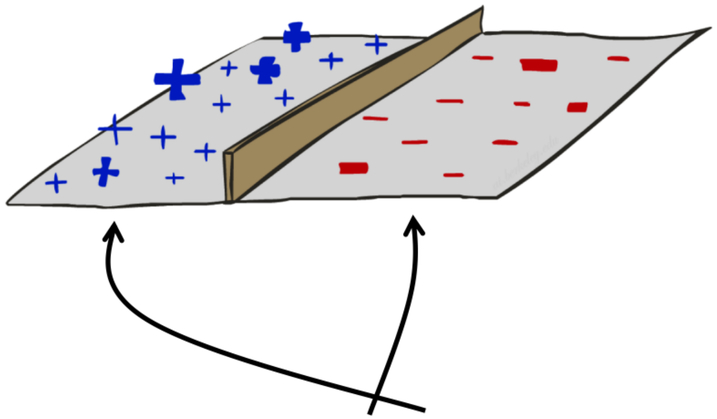
1. the k-nearest neighbor classification algorithm
2. how logistic regression works
3. the concept of the decision boundary
4. cross-entropy error function for binary and multi-class classification
5. error analysis: metrics, confusion matrix and ROC curves
6. multi-class classification using the OneVsRest and the softmax classifier.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

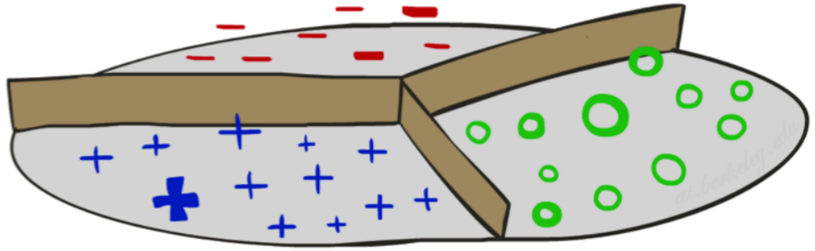
Classification

Binary classification



$y \in \{0,1\}$

Multiclass classification



$y \in \{0,1,2\}$

Images Source: ***UC Berkeley CS188 – Intro to AI course***

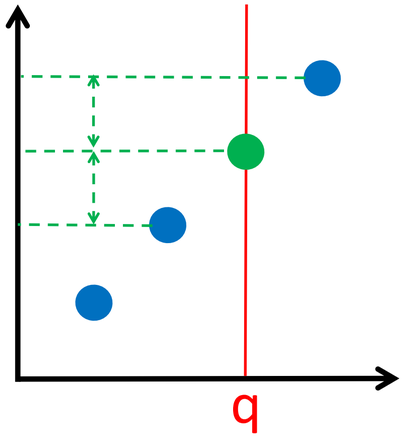
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

K-nearest neighbour classification

Given:

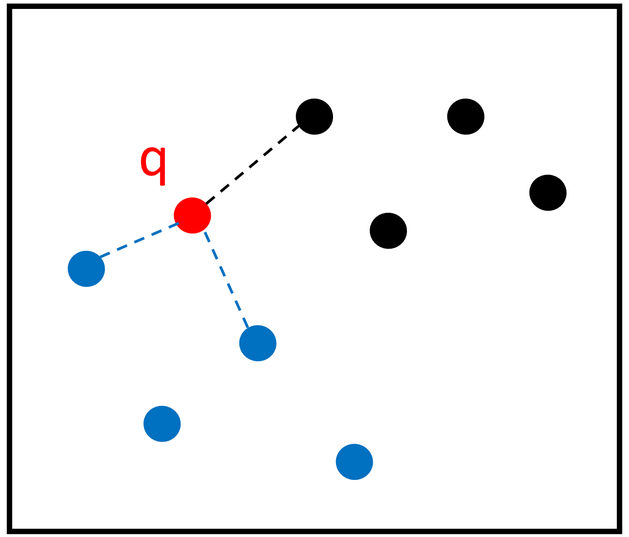
- Training Data $D = \{x^{(i)}, y^{(i)}\}_{i=1:m}$
- Query point q
- Distance Metric $d(q, x^{(i)})$
- Number of neighbours k



NN = { $i : d(q, x^{(i)})$ k smallest } (k closest to query point)

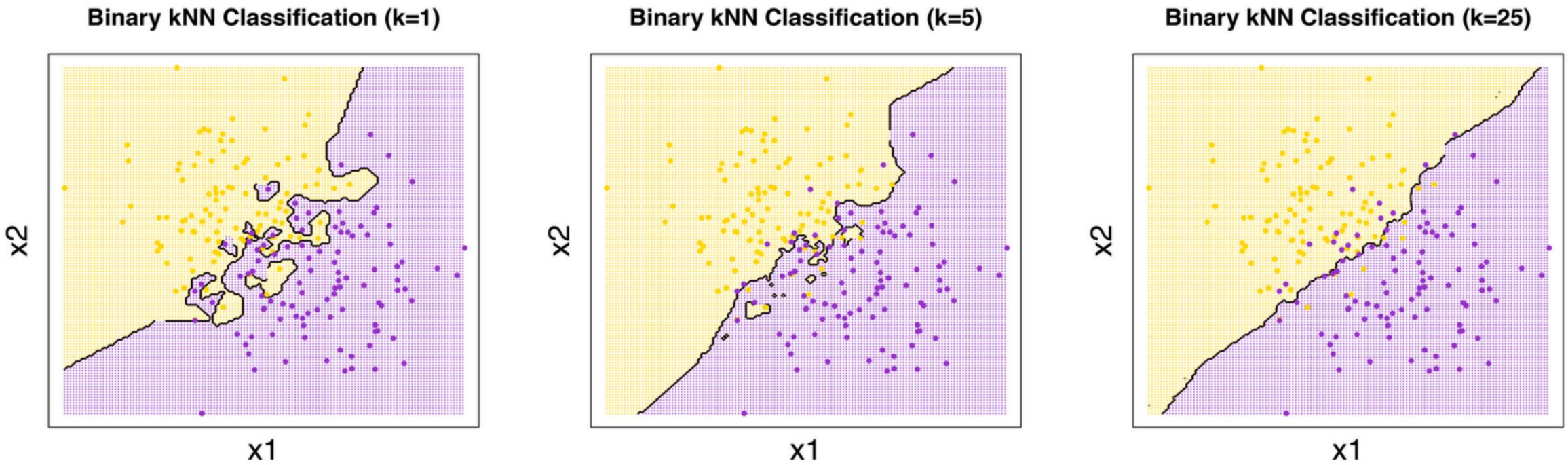
Return:

- Regression: mean of the $y^{(i)}$ in NN
- Classification: vote of the $y^{(i)}$ in NN



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

K-nearest neighbour classification



What if k = m? ⟶ **Always predict the majority class**

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example

```
In [1]: X = [[0], [1], [2], [3]]
   ...: y = [0, 0, 1, 1]

In [2]: from sklearn.neighbors import KNeighborsClassifier

In [3]: neigh = KNeighborsClassifier(n_neighbors=3)

In [4]: neigh.fit(X, y)
Out[4]:
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=None, n_neighbors=3, p=2,
           weights='uniform')

In [5]: print(neigh.predict([[1.1]]))
[0]

In [6]: print(neigh.predict_proba([[0.9]]))
[[0.66666667 0.33333333]]
```

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

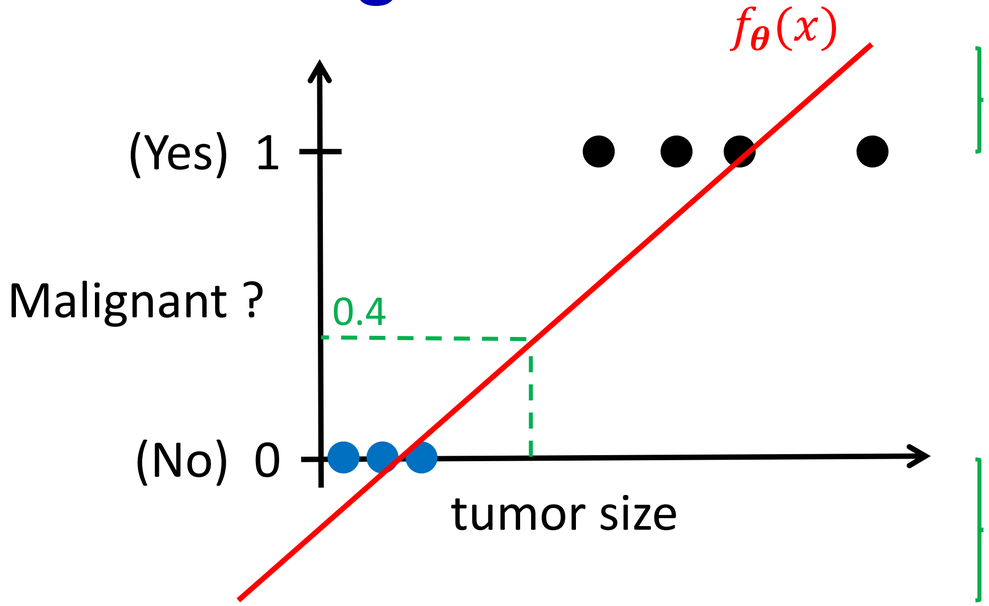
Master programmes in Artificial Intelligence 4 Careers in Europe

Logistic Regression

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Linear regression for classification



Logistic Regression

if $f_\theta(x) \geq 0.5$, predict "y=1"
if $f_\theta(x) < 0.5$, predict "y=0"

$0 \leq f_\theta(x) \leq 1$

Linear regression could work but:

- we need another parameter (threshold):

if $f_\theta(x) \geq 0.4$, predict "y=1"
if $f_\theta(x) < 0.4$, predict "y=0"

- it is unbounded $f_\theta(x)$ can be > 1 and < 0

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Logistic regression

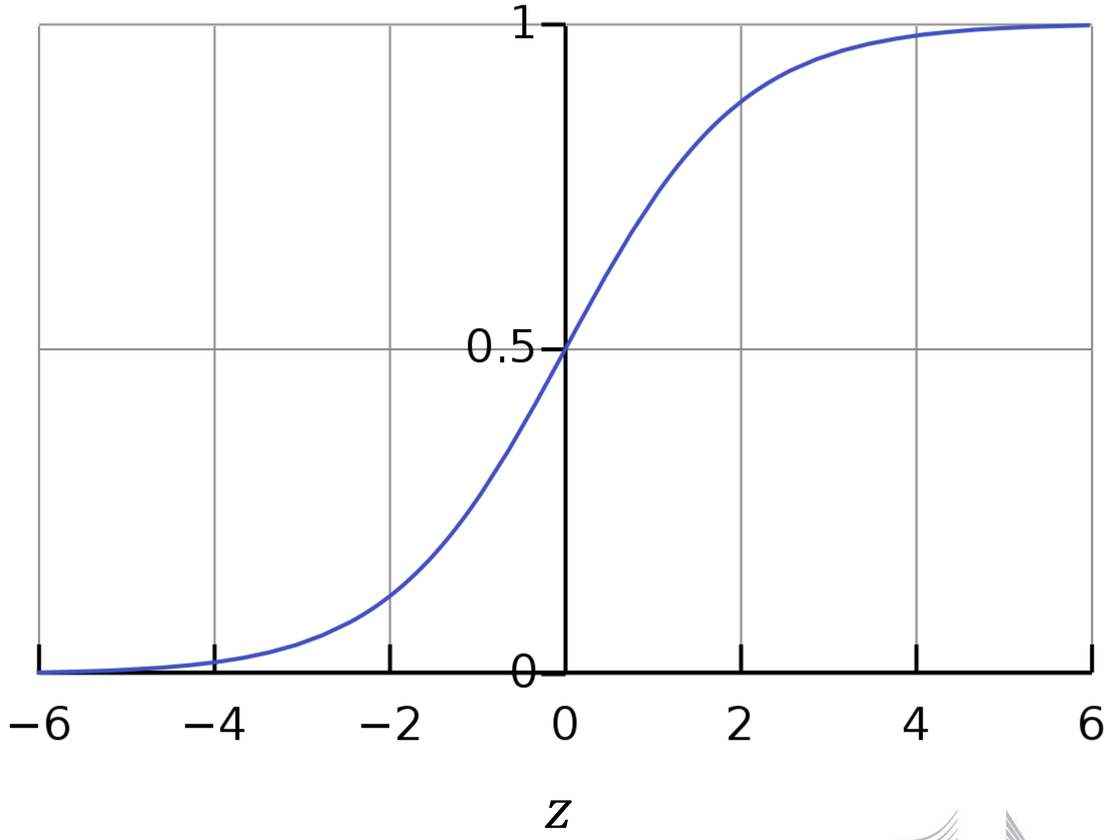
$$0 \leq f_{\boldsymbol{\theta}}(\boldsymbol{x}) \leq 1$$

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^T \boldsymbol{x}$$

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{(1 + e^{-\boldsymbol{\theta}^T \boldsymbol{x}})}$$

estimated probability that “y=1” on input x

Logistic function
or
Sigmoid function



This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Quiz

Suppose that we want to predict whether a tumor is malignant or not based on its size.

What is x (the input) and what do we mean by “y=1” and “y=0”?

x: size, “y=1”: tumor is malignant, “y=0”: tumor is not malignant

Suppose now that $f_{\boldsymbol{\theta}}(\boldsymbol{x}) = 0.8$

What does this mean?

The estimated probability that x is a tumor as predicted by our model is 0.8.

Similarly the estimated probability that x is NOT a tumor is 0.2 (= 1 - 0.8).

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Decision boundary

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = g(\boldsymbol{\theta}^T \boldsymbol{x})$$

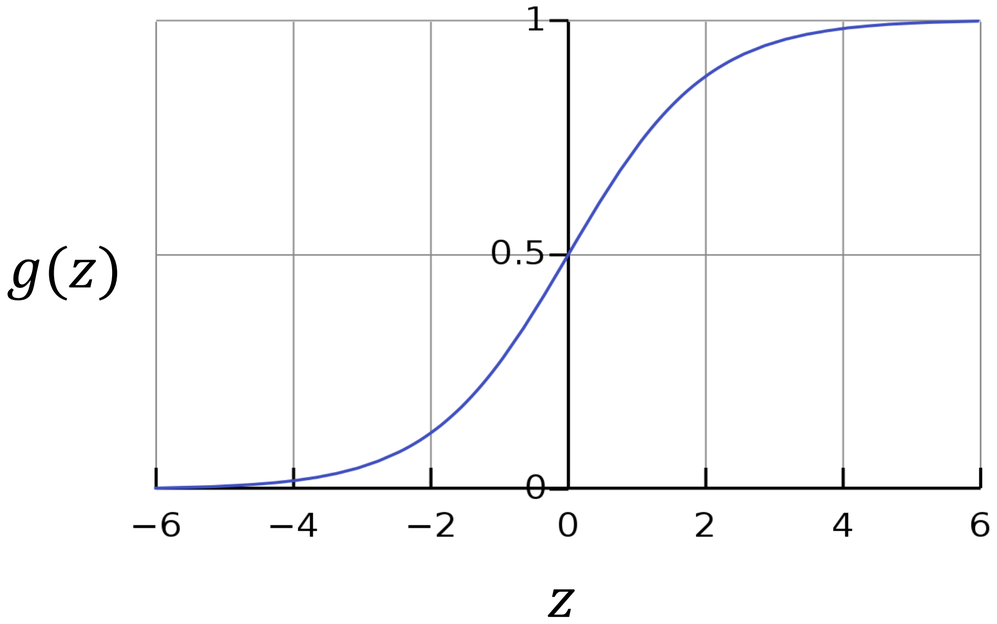
$$g(z) = \frac{1}{(1 + e^{-z})}$$

if $f_{\boldsymbol{\theta}}(\boldsymbol{x}) \geq 0.5$, predict “y=1”

$$\boldsymbol{\theta}^T \boldsymbol{x} \geq 0$$

if $f_{\boldsymbol{\theta}}(\boldsymbol{x}) < 0.5$, predict “y=0”

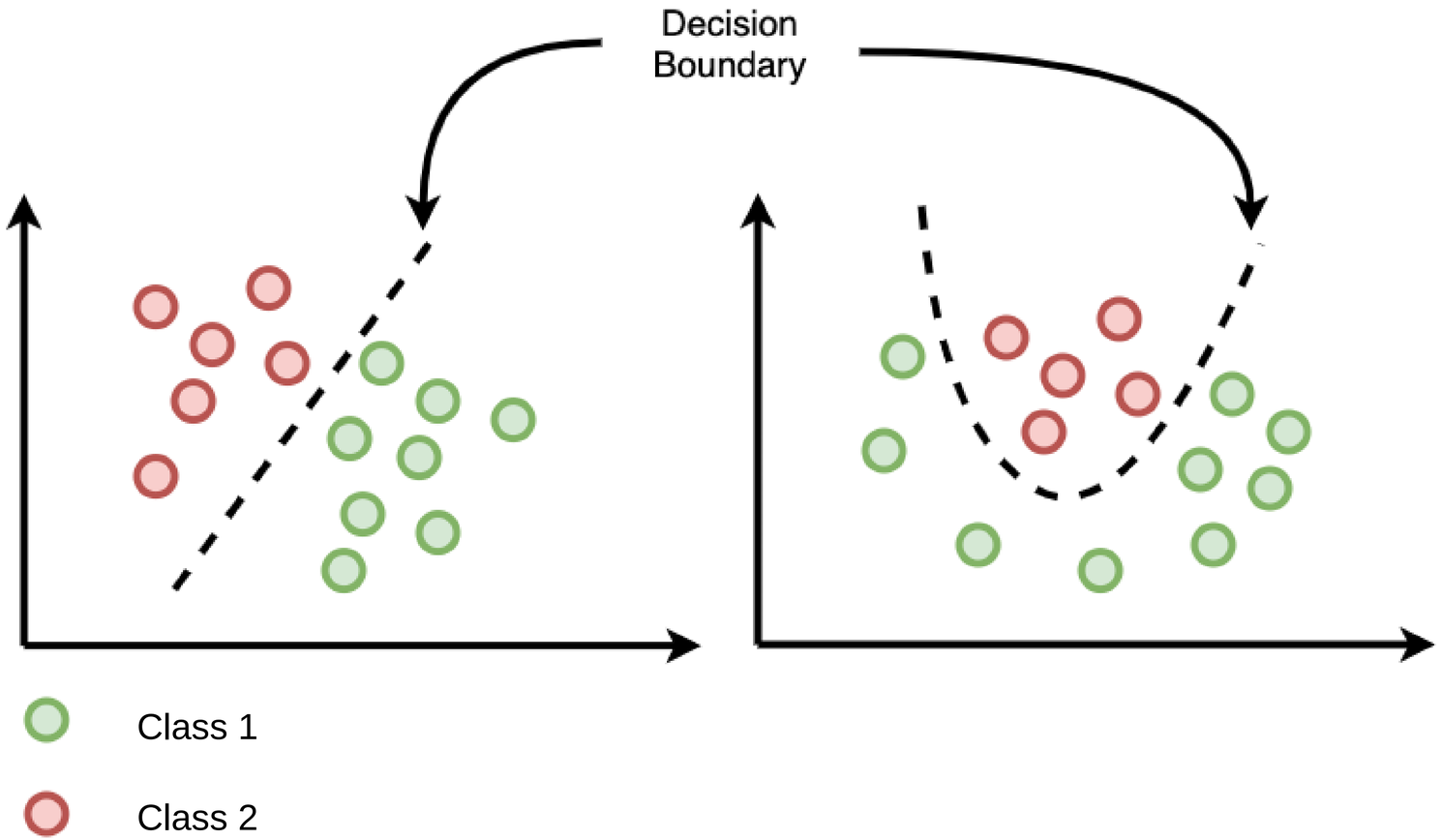
$$\boldsymbol{\theta}^T \boldsymbol{x} < 0$$



Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Decision boundary



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Cost function

We do NOT minimize the Mean Squared Error

Instead we minimize the **Cross-Entropy Error:**

- Intuitively: distance between two probability distributions (target and estimated)
- The Squared Error has a non-convex landscape when used with Logistic Regression
- The CE Error has a convex landscape

To find $\boldsymbol{\theta}$ we use gradient descent (or more advanced optimization algorithms, e.g., conjugate gradient, L-BFGS).

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Cost function

Cross-Entropy Error

$$L(\boldsymbol{\theta}) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)} log f_{\boldsymbol{\theta}}(x^{(i)}) + (1 - y^{(i)}) log(1 - f_{\boldsymbol{\theta}}(x^{(i)}))]$$

Gradient

$$\nabla_{\theta_j} L(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m}[f_{\boldsymbol{\theta}}(x^{(i)}) - y^{(i)})]x_j^{(i)}$$

same as in

Linear Regression: $f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^T \boldsymbol{x}$

Logistic Regression: $f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{(1 + e^{-\boldsymbol{\theta}^T \boldsymbol{x}})}$

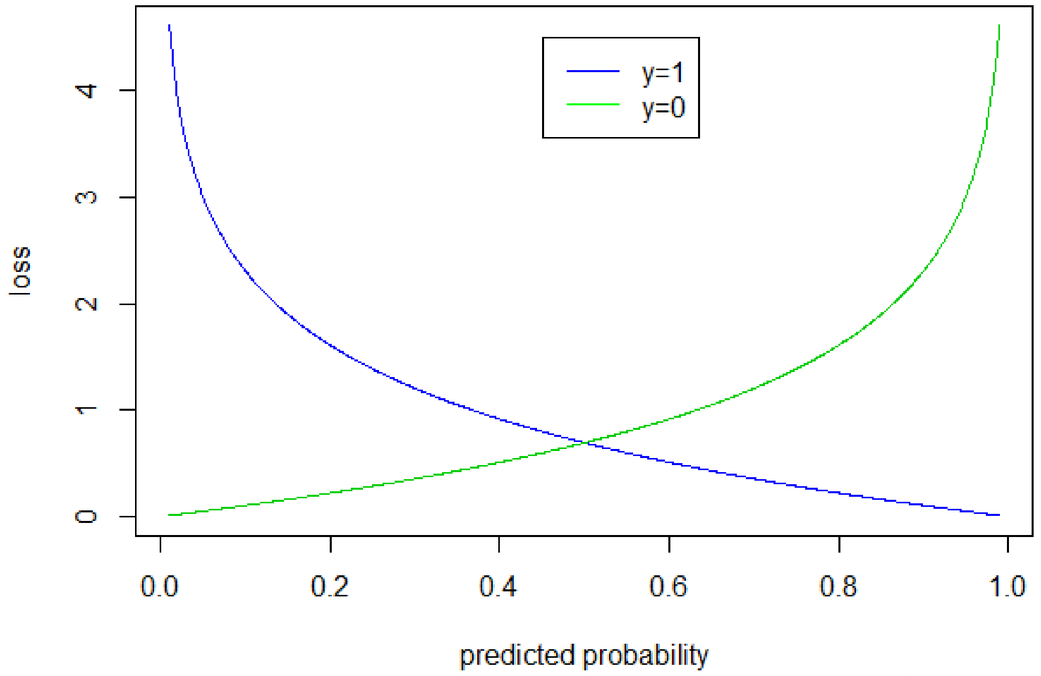
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Cross Entropy Error

$$L(\boldsymbol{\theta}) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)} log f_{\boldsymbol{\theta}}(x^{(i)}) + (1 - y^{(i)}) log(1 - f_{\boldsymbol{\theta}}(x^{(i)}))$$

$$L(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m}[loss(f_{\boldsymbol{\theta}}(x^{(i)}), y^{(i)})]$$

$$loss(f_{\boldsymbol{\theta}}(x^{(i)}), y^{(i)}) = \begin{cases} -log(f_{\boldsymbol{\theta}}(x^{(i)})) & if\ y^{(i)} = 1 \\ -log(1 - f_{\boldsymbol{\theta}}(x^{(i)})) & if\ y^{(i)} = 0 \end{cases}$$



source

Cost increases to infinity as predicted probability is very far away from the actual

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example

```
In [1]: X = [[0], [1], [2], [3]]
   ...: y = [0, 0, 1, 1]
   ...:
   ...: from sklearn.linear_model import LogisticRegression
   ...: clf = LogisticRegression(penalty='none').fit(X, y)

In [2]: print(clf.predict([[1.1]]))
[0]

In [3]: print(clf.predict_proba([[0.9]]))
[[9.99979559e-01 2.04410118e-05]]

In [4]: print(clf.score(X, y))
1.0
```


This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Error Analysis

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Error Analysis

In binary classification we may characterize the performance of a classifier based on a metric called **accuracy**

- number of correct predictions divided by the number of samples in the set
- gives a value between 0 and 1.

However, we can go beyond Accuracy by investigating the types of error the classifier does.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Error Analysis

Example (source): Given a sample of 12 individuals, 8 that have been diagnosed with cancer and 4 that are cancer-free, where individuals with cancer belong to class 1 (positive) and non-cancer individuals belong to class 0 (negative), we can display that data as follows:

Individual Number	1	2	3	4	5	6	7	8	9	10	11	12
Actual Classification	1	1	1	1	1	1	1	1	0	0	0	0

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Error Analysis

Example continued: Assume that we have a classifier that distinguishes between individuals with and without cancer in some way, we can take the 12 individuals and run them through the classifier. The classifier then makes 9 accurate predictions and misses 3: 2 individuals with cancer wrongly predicted as being cancer-free (sample 1 and 2), and 1 person without cancer that is wrongly predicted to have cancer (sample 9).

Individual Number	1	2	3	4	5	6	7	8	9	10	11	12
Actual Classification	1	1	1	1	1	1	1	1	0	0	0	0
Predicted Classification	0	0	1	1	1	1	1	1	1	0	0	0

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Error Analysis

Example continued:

Individual Number	1	2	3	4	5	6	7	8	9	10	11	12
Actual Classification	1	1	1	1	1	1	1	1	0	0	0	0
Predicted Classification	0	0	1	1	1	1	1	1	1	0	0	0
Result	FN	FN	TP	TP	TP	TP	TP	TP	FP	TN	TN	TN

True Positive (TP): Positive actual classification, positive predicted classification

True Negative (TN): Negative actual classification, negative predicted classification

False Positive (FP): Negative actual classificiation, positive predicted classification

False Negative (FN): Positive actual classificiation, negative predicted classification

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Confusion matrix

		Predicted condition	
	Total population = P + N	Positive (PP)	Negative (PN)
Actual condition: Positive (P)	Positive (P)	True positive (TP)	False negative (FN)
Actual condition: Negative (N)	Negative (N)	False positive (FP)	True negative (TN)

		Predicted condition	
	Total 8 + 4 = 12	Cancer 7	Non-cancer 5
Actual condition: Cancer 8	Cancer 8	6	2
Actual condition: Non-cancer 4	Non-cancer 4	1	3

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Metrics

Accuracy = (TP + TN) / (P + N)

Precision = TP / (TP + FP)

True positive rate = TP / (TP + FN)
- Also known as: **recall**, sensitivity, hit rate or probability of detection

False Positive Rate = FP / (FP + TN)
- Also known as: probability of false alarm

F1-score (harmonic mean of precision and sensitivity) = 2TP / (2TP + FP + N)

		Predicted condition	
	Total population = P + N	Positive (PP)	Negative (PN)
Actual condition	Positive (P)	True positive (TP)	False negative (FN)
	Negative (N)	False positive (FP)	True negative (TN)

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

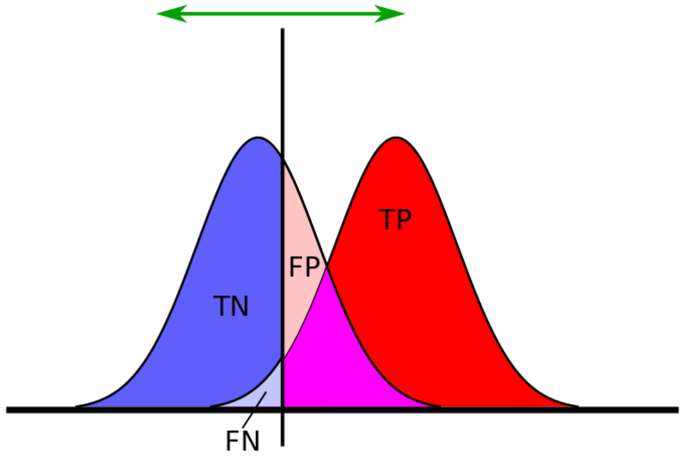
ROC Curve

A binary classifier needs a threshold to decide where to put the decision boundary

The threshold choice affects both the TPR and the FPR

We can vary the thresholds and plot the TPR with respect to FPR

This is known as a Receiver operating characteristic (ROC) Curve



TP	FP
FN	TN

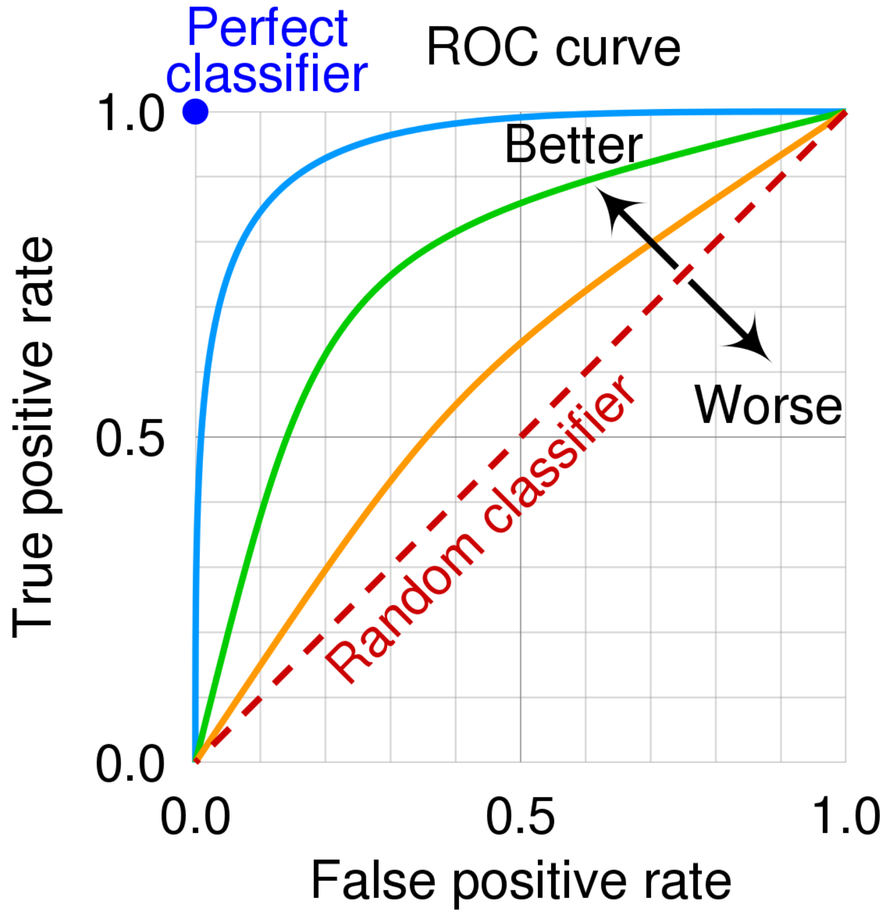
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

ROC Curve

- **ROC curve**: graph showing the performance of a binary classification model at **all** classification thresholds.

- Lowering the classification threshold classifies more items as positive, thus increasing both FPs and TPs.

- **Purpose**:
  - Analyze the strength / predictive power of a classifier
  - Compare classifiers
  - Determine the “optimal” threshold based on user’s preferences



source

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

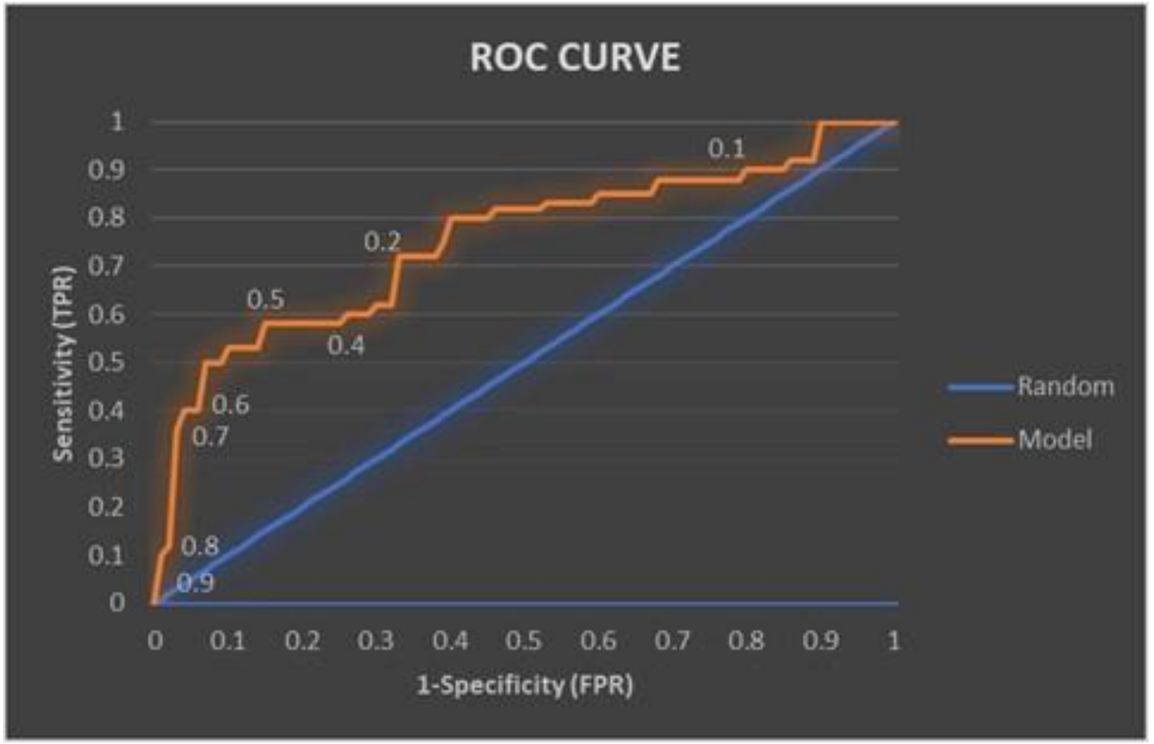
ROC Curve: It's a tradeoff

Which threshold would you choose?

- If you care more about FPR ≤ 0.1
(obtained with thresholds 0.9, 0.8, 0.7, and 0.6)

- If you care more about high TPR (≥ 0.7)

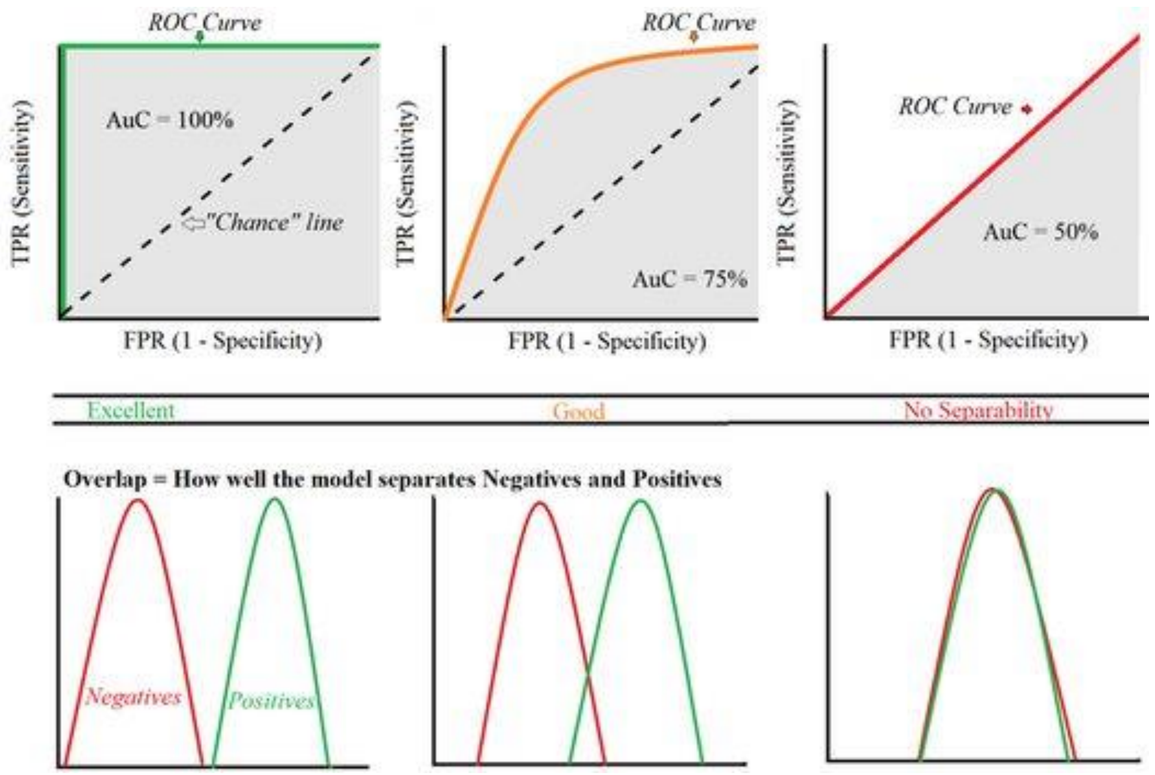
- If you are happy with a TPR ≈ 0.6
(obtained with thresholds 0.5 and 0.4)



source

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

ROC Curve: Comparing Classifiers



Source

Area Under the Curve (AUC)

- single metric (from 0 to 1)
- aggregate measure of performance across all possible classification thresholds
- measure of how much the model is capable of distinguishing the two classes

- **Scale invariant**: measures how well predictions are ranked, rather than their absolute values
- **Classification-threshold invariant**: measures quality of model's predictions irrespective of classification threshold

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Quiz

Suppose that you have a classifier that predicts all positives as negatives and all negatives as positives.

1. **Is this classifier worse than the random classifier?**

   YES

2. **What is its AUC score?**

   0

3. **Can we fix it somehow?**

   Yes. Just flip its predictions and you get a perfect classifier!

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

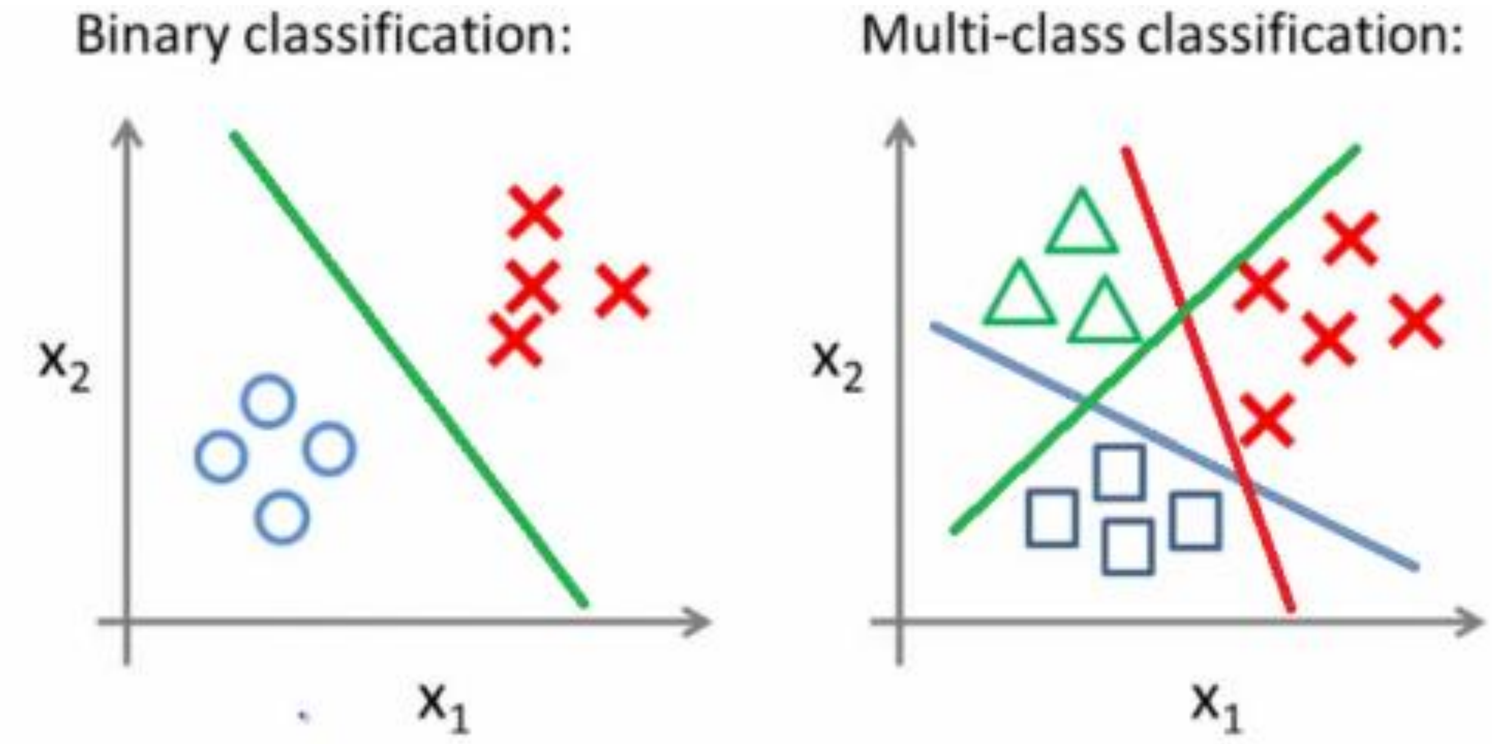
Master programmes in Artificial Intelligence 4 Careers in Europe

Multiclass Classification

Co-financed by the European Union
Connecting Europe Facility

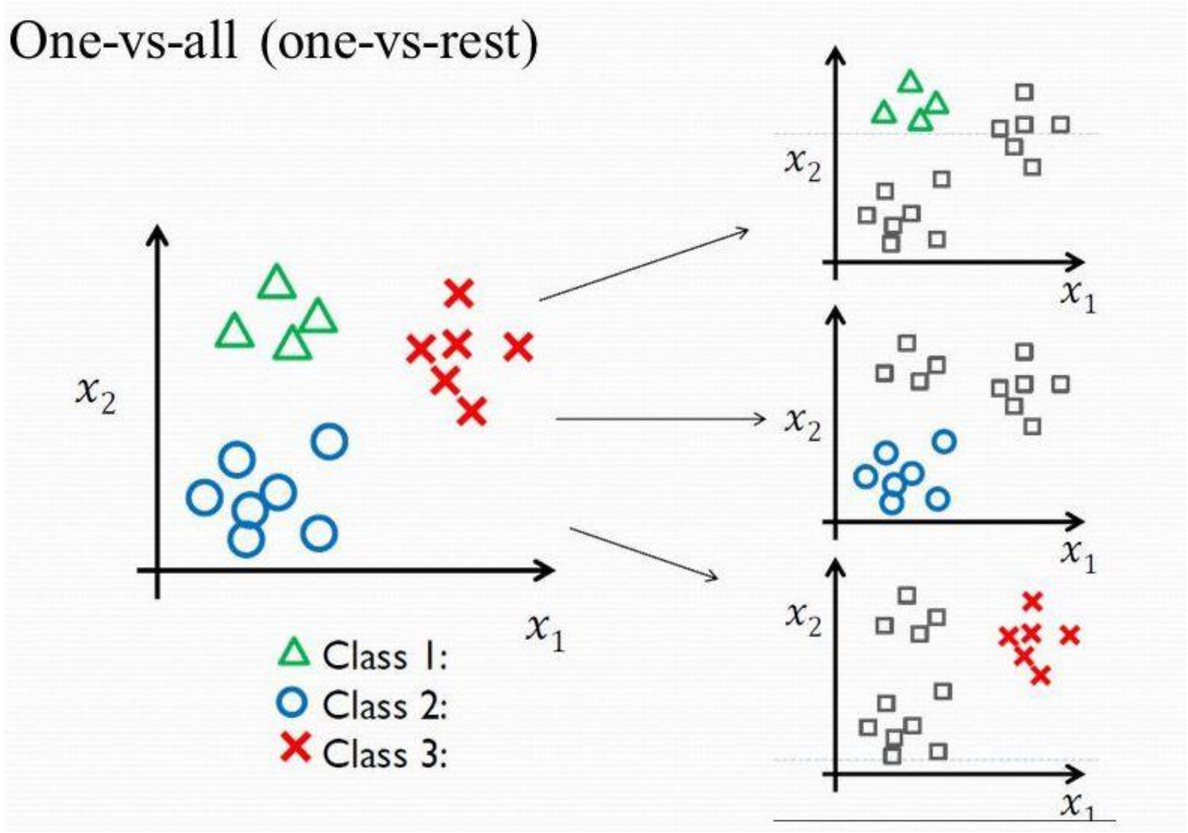
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Multiclass classification



This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Multiclass classification



Convert the problem into K binary classification problems (K number of classes)

Train a logistic regression classifier for each class.

To make a prediction pick the class whose classifier is most confident.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Softmax classifier

- Natural generalization of binary Logistic Regression classifier to multiple classes
- Also called multinomial logistic regression
- **Output**: normalized class probabilities

- **Softmax function:** converts a vector of K real numbers into a probability distribution of K possible outcomes.

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad \text{for } i = 1, \ldots, K \text{ and } \mathbf{z} = (z_1, \ldots, z_K) \in \mathbb{R}^K.$$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Cross Entropy Error for Multiple Classes

- Categorical Cross Entropy Error

- Each $\boldsymbol{y}^{(i)}$ is one-hot encoded for K classes
- The estimated $\hat{\boldsymbol{y}}^{(i)}$ is the output of the softmax classifier

$$loss(\hat{\boldsymbol{y}}^{(i)}, \boldsymbol{y}^{(i)}) = -\sum_{c=1}^{K} [y_c^{(i)} \, log \, \hat{y}_c^{(i)}]$$

$$L(\boldsymbol{\theta}) = -\frac{1}{m} \sum_{i=1}^{m} \sum_{c=1}^{K} [y_c^{(i)} \, log \, \hat{y}_c^{(i)}]$$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Softmax classifier: numeric stability issues

- When writing code for computing the Softmax function, the term e^{z_j} and $\sum_{c=1}^{K} e^{z_c}$ may be very large due to the exponentials.
- Dividing large numbers can be numerically unstable, so it is important to use a normalization trick.

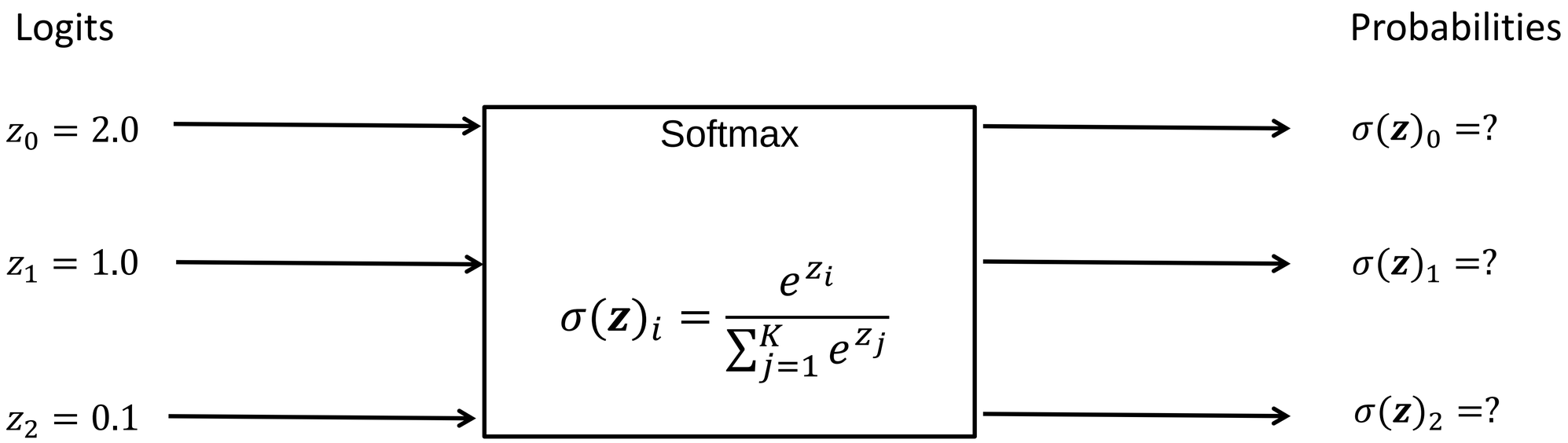
$$\frac{e^{z_j}}{\sum_{c=1}^{K} e^{z_c}} = \frac{A\, e^{z_j}}{A \sum_{c=1}^{K} e^{z_c}} = \frac{e^{z_j + logA}}{\sum_{c=1}^{K} e^{z_c + logA}}$$

$$logA = -max_c z_c$$

Shifts values of z_j so that the highest is 0

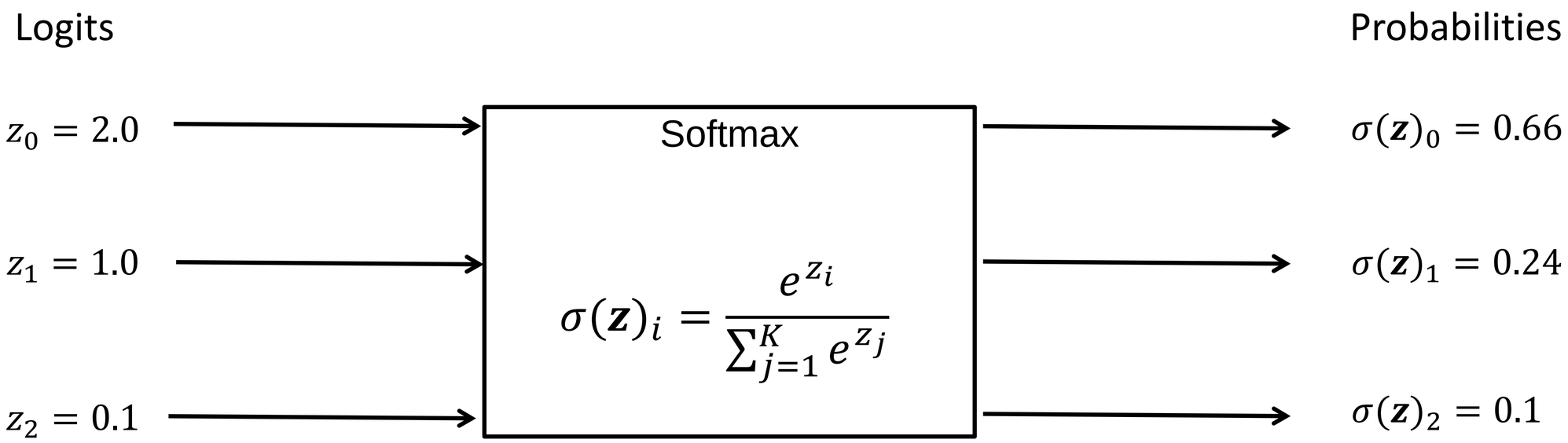
This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Softmax classifier: example



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Softmax classifier: example



This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Next Lecture

- Model Evaluation and Improvement
- Trees and Forests

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 5: Model Evaluation and Improvement

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Revision

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression

- Regression is the supervised learning problem of predicting a continuous value
- K-nearest neighbor regression
  - Nonparametric model
  - Prediction is the average of the K nearest neighbors
  - K=1: noisy, 1<K<m better captures the trend, K=m always produces the average of all
  - Strengths: no training time, handles nonlinearities, …
  - Weaknesses: prediction becomes slower as the dataset becomes larger, …
- Linear regression
  - Parametric model: 1 parameter per dimension + intercept term
  - Prediction is the dot product of parameter vector and input vector
  - Learning can be done either using gradient-based methods or computing the analytic solution

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression

- Linear regression optimizes the MSE, which has a convex shape (single optimum)
- Gradient descent is about computing the partial derivatives of a function at a point, and changing the point by adding a value proportional to the negative gradient
- A small learning rate results in slower convergence, a large learning rate may result in divergence
- Gradient vector of MSE for linear regression
- Need to use feature scaling with gradient descent
- Linear regression
  - Strengths: constant prediction time, analytic solution easy to implement,…
  - Weaknesses: cannot model nonlinear relationships,…
- Weakness can be addressed using polynomial features

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Classification

- Classification is the supervised learning problem of predicting a discrete value
- K-nearest neighbor classification
  - Prediction is the majority vote of the K nearest neighbors
  - K=1: fit the noise, K=m always predict the majority class
- Logistic regression
  - Simple method for binary classification
  - Feeds the output of linear regression through the sigmoid function which makes it in [0,1]
  - The output is seen as the estimated probability of predicting the positive class
  - Uses a default (probability) threshold of 0.5 for placing the decision boundary
  - Decision boundary of logistic regression is linear, can be nonlinear if we use polynomial features

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Classification

- Logistic regression optimizes the Cross-Entropy error which has a convex shape
- We use gradient-based methods to find the minimum
- The CE error penalizes the model a lot if its predicted probability is very far from the actual
- Gradient vector of CE error for logistic regression (similar to MSE for linear regression)
- Error analysis for binary classifiers
  - True positives, True Negatives, False Positives and False Negatives
  - Metrics: Accuracy, Precision, TP rate, FP rate, F1-score
  - ROC curve: plots the FP rate and the TP rate for all classification thresholds
  - AUC score: a single metric based on ROC which can be used to compare classifiers
- Multiclass classification
  - One-vs-rest: train K binary classifiers; prediction is the class of the most confident classifier
  - Softmax: uses one-hot encoding of classes; prediction is a probability distribution over K classes

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 5: Model Evaluation and Improvement

Learning Outcomes

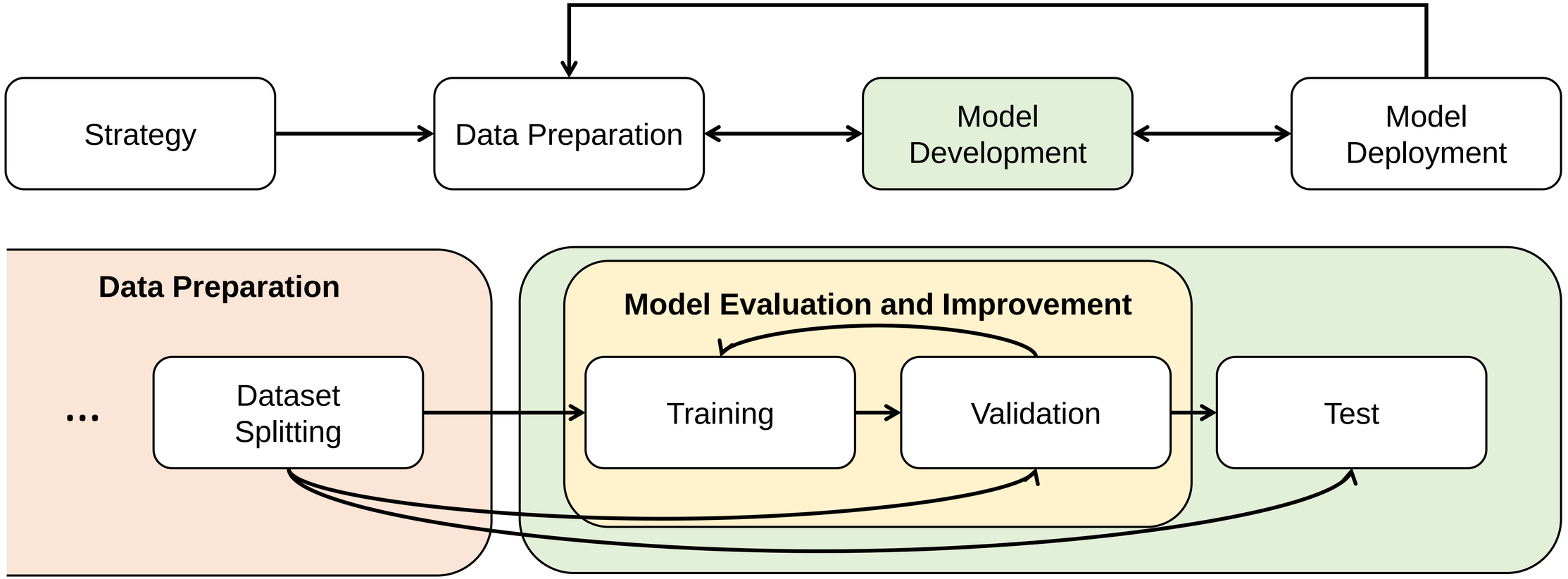
You will understand:

1. the issue of generalization in machine learning and how it is related to model evaluation
2. the concepts of overfitting, underfitting and the bias-variance tradeoff
3. the importance of training, validation and test sets
4. what k-fold cross-validation is and when to use it
5. when to acquire more data to improve ML models
6. how to improve ML models using L1 and L2 regularization
7. how to improve ML models by tuning their hyperparameters

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model Development



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

The issue of generalization

What is generalization?

- Not memorizing the very details of our training samples
- Model's ability to have “good performance” on previously unseen data, drawn from the same distribution as the one used to train the model

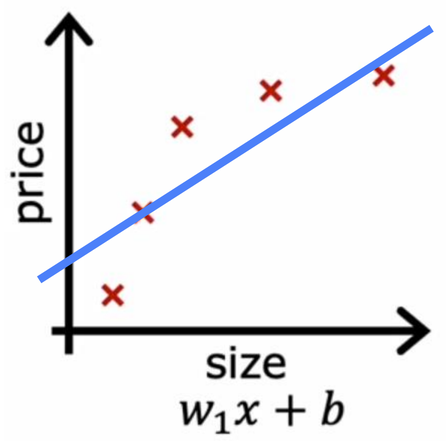
Why we want generalization?

- Our dataset is a sample from a broader distribution
- We want to train a model in a way that does not focus on the “noise” in the samples
- Ignoring the noise will allow the model to have predictive power over the broader distribution

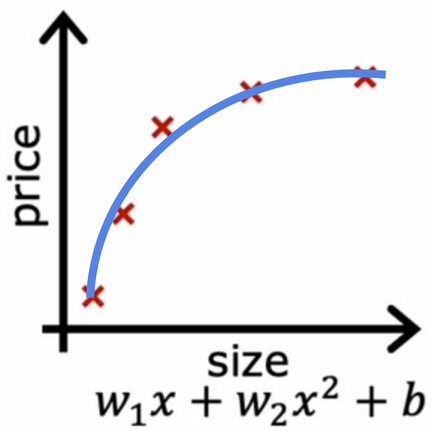
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression Example

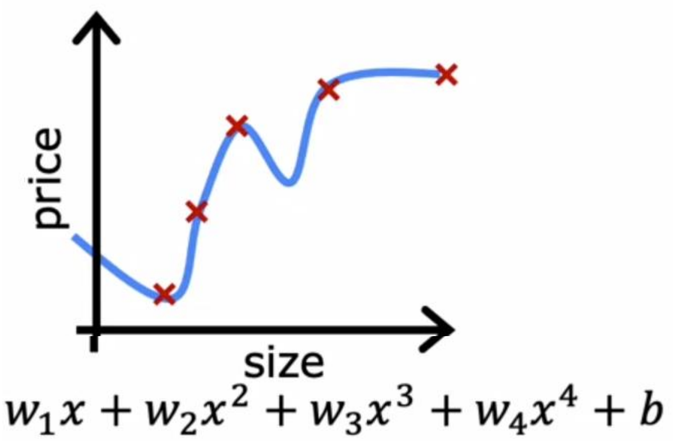
Source: Andrew Ng, Machine Learning - Coursera



(a)



(b)



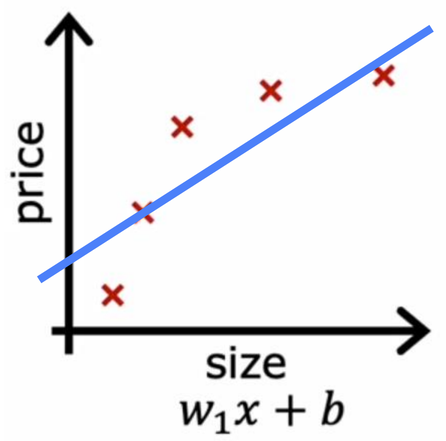
(c)

Which model is better, i.e., provides better generalization?

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression Example: Underfitting vs Overfitting

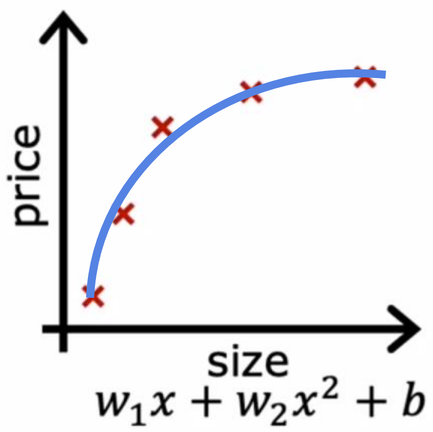
Source: Andrew Ng, Machine Learning - Coursera



Model too simple

Does not fit the training set well

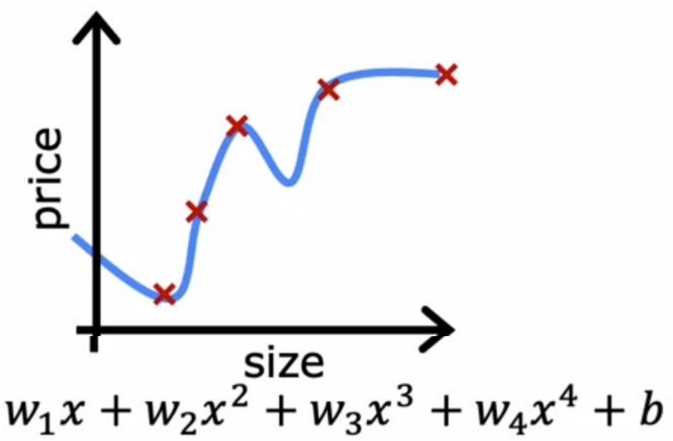
Underfitting



Model just right

Fits training set well

Generalization



Model too complex

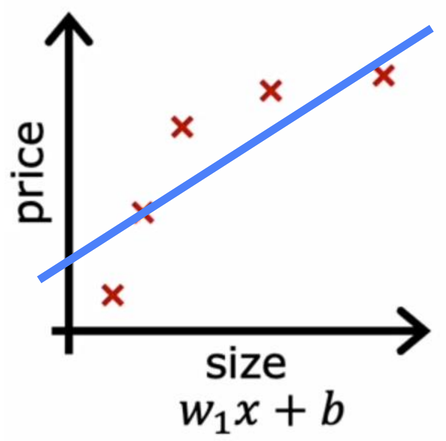
Fits training set extremely well

Overfitting

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression Example: Underfitting vs Overfitting

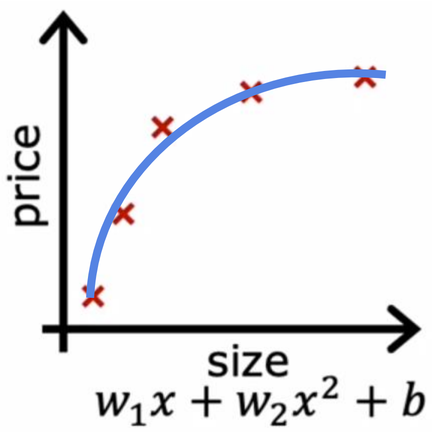
Source: Andrew Ng, Machine Learning - Coursera



Model too simple

Does not fit the training set well

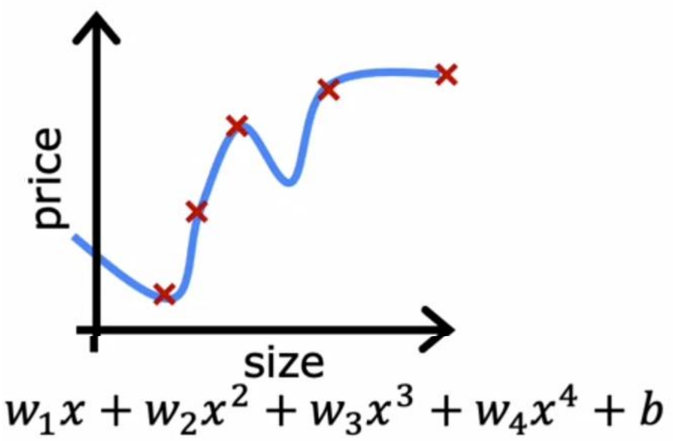
High Bias



Model just right

Fits training set well

Generalization



Model too complex

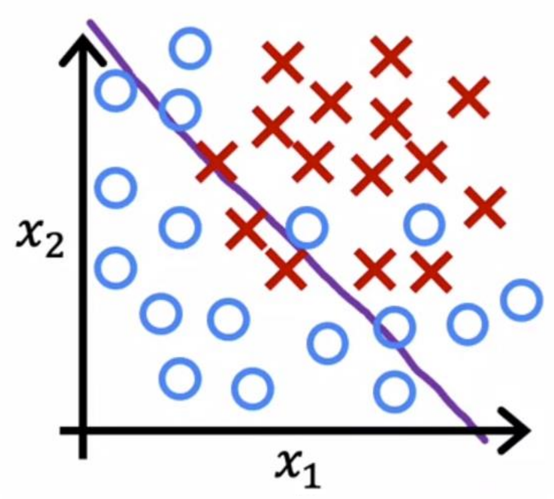
Fits training set extremely well

High Variance

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

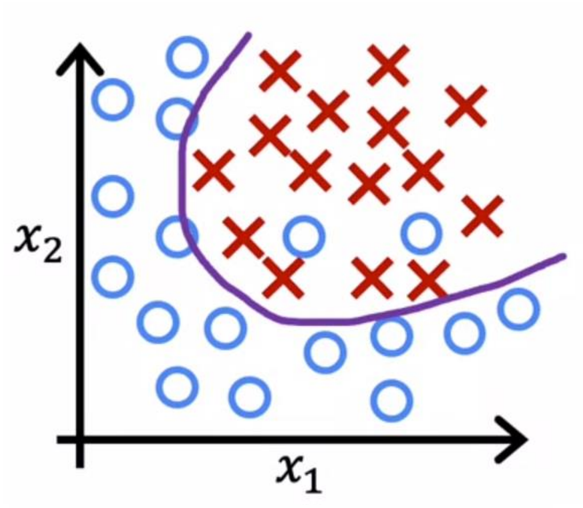
Classification Example

Source: Andrew Ng, Machine Learning - Coursera



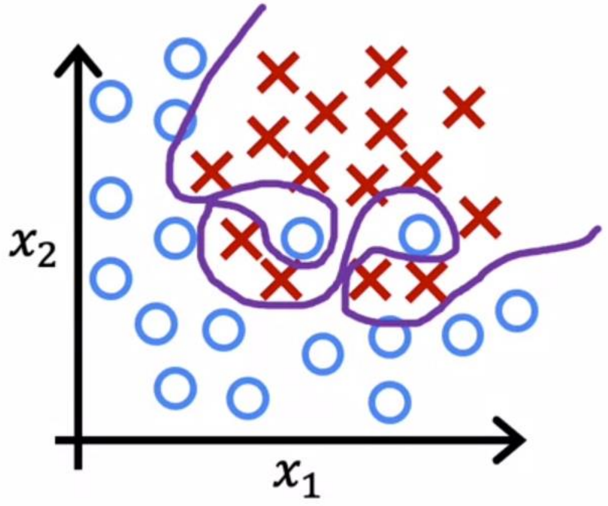
$z = \theta_0 + \theta_1 x_1 + \theta_2 x_2$
$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = g(z)$
g is the sigmoid function

Underfitting
High Bias



$z = \theta_0 + \theta_1 x_1 + \theta_2 x_2$
$+\theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2$

Generalization



$z = \theta_0 + \theta_1 x_1 + \theta_2 x_2$
$+\theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2$
$+\theta_5 x_1^2 x_2^3 + \ldots$

Overfitting
High Variance

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Quiz

Source: Andrew Ng, Machine Learning - Coursera

Our goal when creating a model is to be able to use the model to predict outcomes correctly for **new examples**. A model which does this is said to **generalize** well.

When a model fits the training data well but does not work well with new examples that are not in the training set, this is an example of:

1. Underfitting (high bias)
2. Overfitting (high variance)
3. A model that generalizes well (neither high variance nor high bias)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Bias-Variance Tradeoff

Bias: error from erroneous (simplifying) assumptions in the learning algorithm.

- High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting).

Variance: error from sensitivity to small fluctuations in the training set.

- High variance may result from an algorithm modeling the random noise in the training data (overfitting).

Bias-variance tradeoff: conflict in trying to simultaneously minimize these two sources of error that prevent supervised learning algorithms from generalizing beyond their training set.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Bias-Variance Decomposition of the MSE

Assume there is a function with noise: $y = f(x) + \varepsilon$, where $\varepsilon \sim \mathrm{N}(0, \sigma^2)$

We want to find a function $\hat{f}_\theta(x; D)$ that approximates the true function $f(x)$ as well as possible using a learning algorithm and a training set $D = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$, where $D \sim P(x, y)$

We measure the MSE between y and $\hat{f}_\theta(x; D)$: we want to minimize $\left(y - \hat{f}_\theta(x; D)\right)^2$ for all points in D and outside

We cannot do so perfectly, because $y^{(i)}$ contains noise ε so we need to accept some *irreducible error*.

We can decompose the expected error of a model $\hat{f}_\theta$ on an unseen sample x as follows:

$$E_{D,\varepsilon}\left[\left(y - \hat{f}_\theta(x; D)\right)^2\right] = \left(Bias_D\left[\hat{f}_\theta(x; D)\right]\right)^2 + Var_D\left[\hat{f}_\theta(x; D)\right] + \sigma^2$$

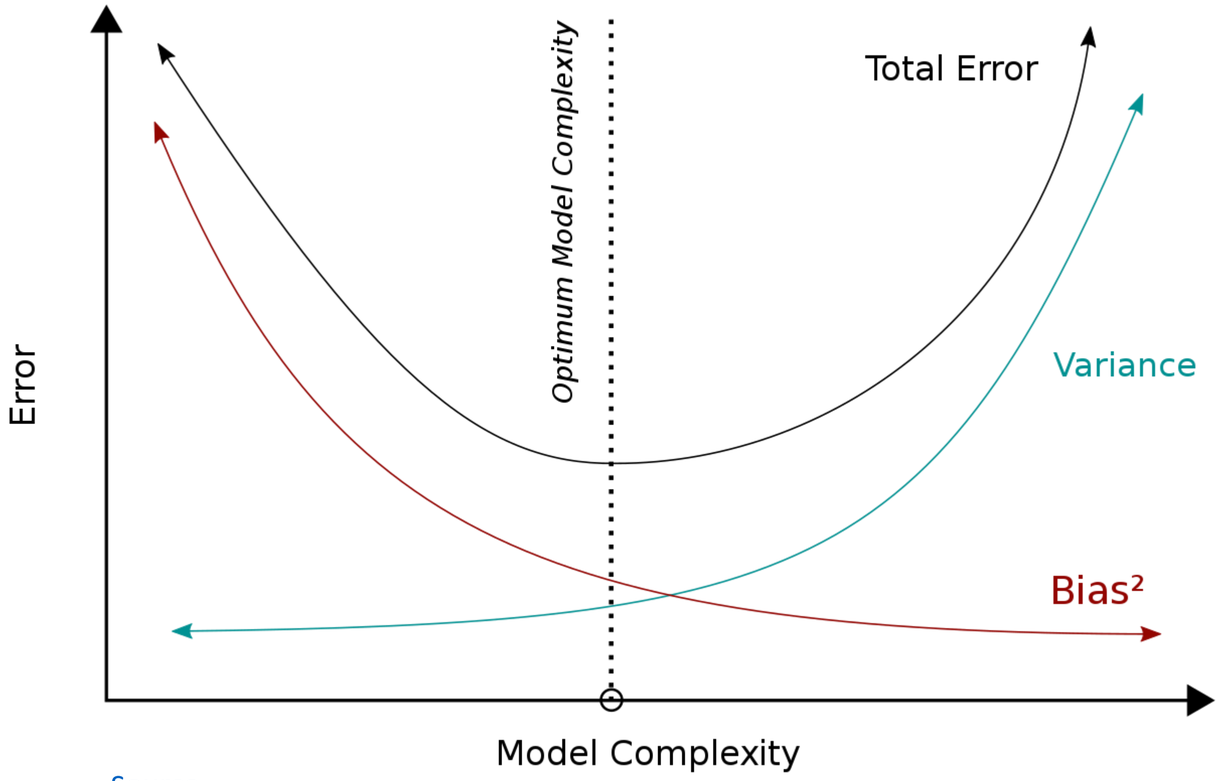
where

$$Bias_D\left[\hat{f}_\theta(x; D)\right] = E_D\left[\hat{f}_\theta(x; D)\right] - f(x)$$

$$Var_D\left[\hat{f}_\theta(x; D)\right] = E_D\left[\left(E_D\left[\hat{f}_\theta(x; D)\right] - \hat{f}_\theta(x; D)\right)^2\right]$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Bias-Variance Tradeoff



Source

As we increase the complexity of the model:

- the squared bias decreases
- the variance increases
- the total error ($Bias^2$ + Variance + ε) decreases up to a point where it starts increasing again.

This is considered the optimal model complexity with regards to the model and dataset that we have.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Bias-Variance Tradeoff

In k-nearest neighbor regression:

- k=1 : low bias, high variance
- k=m : high bias, low variance

In linear regression:

- less features: high bias, low variance
- many features: low bias, high variance

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Quiz

1. Suppose that we have a dataset that contains many features (columns). We use a linear regression model for our prediction, however, it seems that it suffers from *high variance*. **What data preparation methods can we use to decrease the variance?**

2. What we mentioned about the bias-variance tradeoff of MSE is *theoretical*. The definition of Bias includes a term about the *true function*, which we do not have access to. **What can we do in practice to derive the optimal complexity of a model?**

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Generalization evaluation on Train-Test Error

What is generalization?

- Model's ability to have “good performance” on previously **unseen data**, drawn from the same distribution

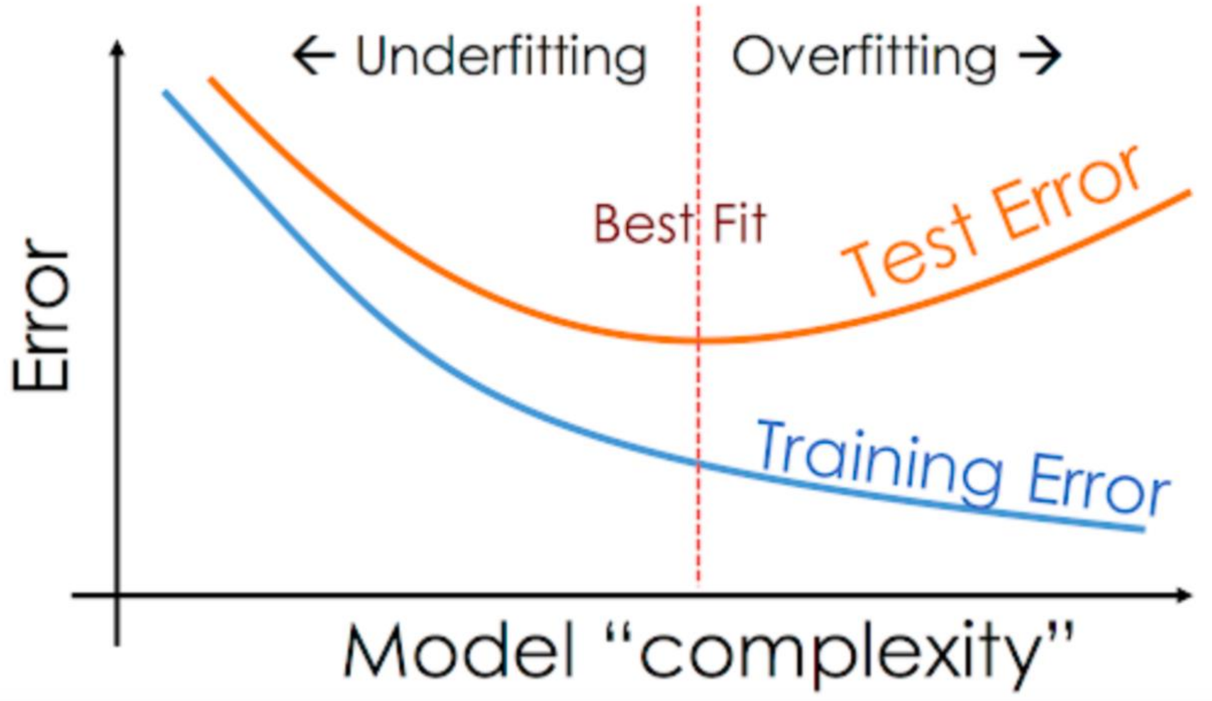
Dataset Splitting

Is there a way to measure generalization?

Let’s visualize it by plotting the train and test errors on the **same** graph!

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Generalization evaluation on Train-Test Error



Source

Increasing the model complexity:

- decreases the training error
- decreases the test error up to a point where it starts increasing again.

Best fit is for model complexity that corresponds to the **lowest test error**

➢ **We select the model that has the lowest test error**

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Revisiting Dataset Splitting

So far: Split Dataset => Training set + Test set

- We perform **model selection** on the **unseen** test set
- However, we did **see** the test set to find the optimal complexity of the model!

How do we know that our selected model does well on unseen samples?

- The selected model might happen to be optimal for our particular choice of a test set
- We should hide the test set: do not perform model selection on it
  - In ML competitions (e.g., Kaggle) the test set is not provided

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Revisiting Dataset Splitting

Instead, Split Dataset:

- **Training set**: train the model
- **Validation set**: perform model selection (hyperparameters, complexity, use different models)
- **Test set**: just report the performance - do **NOT** use for any decisions

In practice (execute in the following order):

1. Split Dataset to Training and Test sets
2. Split Training set to Training and Validation sets (or Test set to Validation and Test sets)

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Model generalization

Training-test split: e.g. 70%-30%

Do not touch the test set during training!

Procedure

1. Train on training set: $L_{train}(\boldsymbol{\theta})$

2. Predict on test set: $L_{test}(\boldsymbol{\theta})$

Model Selection on $\boldsymbol{\theta}$

Model Selection on hyperparameters

Training-validation-test split: e.g. 60%-20%-20%

Do not touch the test set during training-validation!

Procedure

1. For each model/hyperparameter:
   1.1. Train on training set: $L_{train}(\boldsymbol{\theta})$
   1.2. Predict on validation set: $L_{validation}(\boldsymbol{\theta})$

2. Choose models/hyperparameters that have lowest $L_{validation}(\boldsymbol{\theta})$

3. Predict on test set: $L_{test}(\boldsymbol{\theta})$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

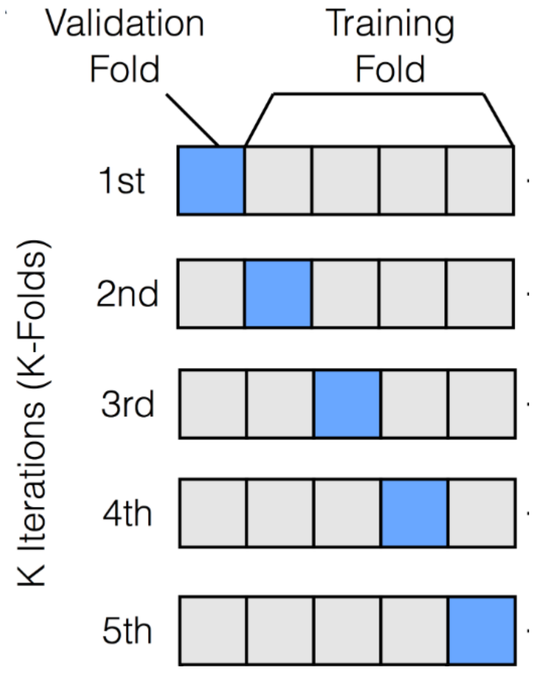
Example

```
In [1]: import numpy as np
   ...: from sklearn.model_selection import train_test_split
   ...:
   ...: X, y = np.arange(20).reshape((10, 2)), range(10)
   ...:
   ...: train_ratio=0.6
   ...: val_ratio=0.2
   ...: test_ratio=0.2
   ...:
   ...: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=1-train_ratio, shuffle=False)
   ...: X_train
Out[1]:
array([[ 0,  1],
       [ 2,  3],
       [ 4,  5],
       [ 6,  7],
       [ 8,  9],
       [10, 11]])

In [2]: np.shape(X_test)
Out[2]: (4, 2)

In [3]: X_val, X_test, y_val, y_test = train_test_split(X_test, y_test, test_size=test_ratio/(test_ratio + val_ratio), shuffle=False)
   ...: np.shape(X_test)
Out[3]: (2, 2)
```


This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

k-fold Cross-Validation

Used when we do not have many datapoints in the training + validation sets



Source

Procedure:

1. Split Training+Validation sets to k folds

2. for i = 1 to k

2.1. Use fold i as the validation set, remaining as the training set

2.2. Train on training set, validate on validation set

3. Report the average validation error over the k validation folds

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

k-fold Cross-Validation

What is the downside of this approach?

- Computationally expensive

What is an extreme/exhaustive form of k-fold cross validation?

- Leave-one-out cross-validation (*m* models trained and validated)

Choosing k:

- Need to balance size of dataset, computational capacity, time...
- Typically: k=5, k=10

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example

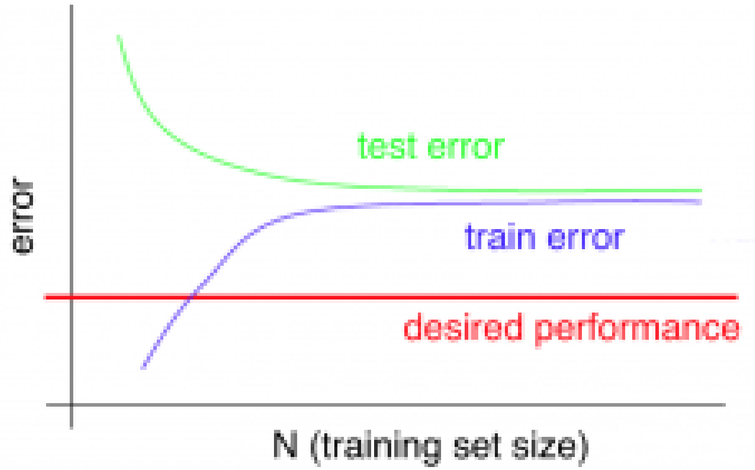
```
In [1]: import numpy as np
   ...: from sklearn.model_selection import KFold
   ...:
   ...: X, y = np.arange(40).reshape((20, 2)), range(20)
   ...:
   ...: K = 5
   ...:
   ...: kf = KFold(n_splits=K)
   ...: kf.get_n_splits(X)
Out[1]: 5

In [2]: for train_index, val_index in kf.split(X):
   ...:     print("TRAIN:", train_index, "VAL:", val_index)
   ...:
TRAIN: [ 4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19] VAL: [0 1 2 3]
TRAIN: [ 0  1  2  3  8  9 10 11 12 13 14 15 16 17 18 19] VAL: [4 5 6 7]
TRAIN: [ 0  1  2  3  4  5  6  7 12 13 14 15 16 17 18 19] VAL: [ 8  9 10 11]
TRAIN: [ 0  1  2  3  4  5  6  7  8  9 10 11 16 17 18 19] VAL: [12 13 14 15]
TRAIN: [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15] VAL: [16 17 18 19]
```

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

When to acquire more data

Source: Andrew Ng, Machine Learning - Coursera



High Bias Models

e.g. simple linear regression

Getting more data will likely NOT going to help.

High Variance Models

e.g. polynomial features, neural network

Getting more data will likely help.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Quiz

1. Having more data helps all machine learning models equally. True or False?

2. Models with high complexity suffer from high variance. Getting more data is a method for decreasing the variance in such models. True or False?

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

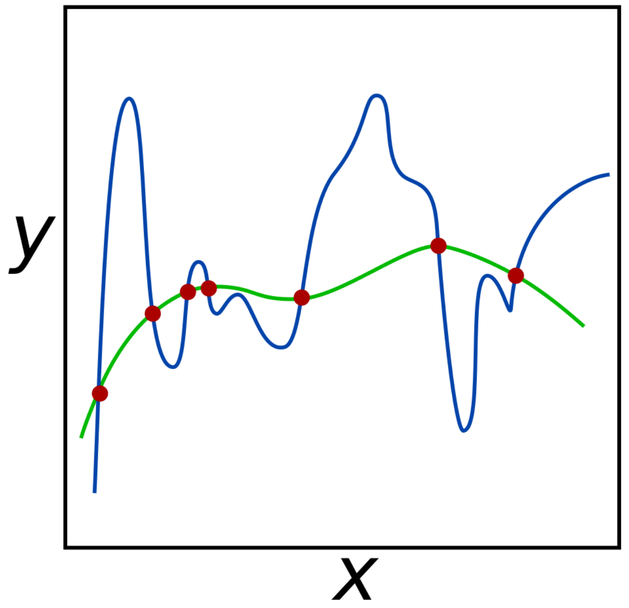
Model improvement through regularization

Complex models have the overfitting problem, i.e., high-variance (fit the noise)

- There are many complex models
- **How to choose between them?**

Solution:

- Simpler models are more likely to generalize (“Ockham’s razor”)
- Reduce variance by simplifying the model



Both blue and green lines have 0 training error

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model improvement through regularization

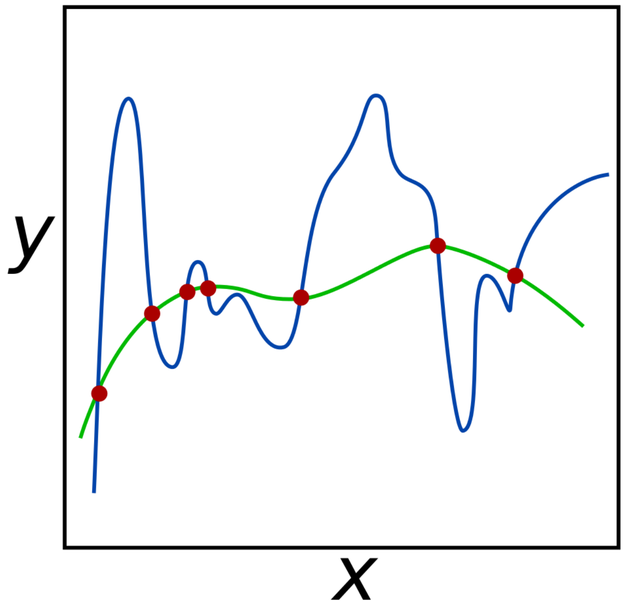
Regularization:

- way of reducing the contribution of certain features

- involves modification of the Loss function L(**θ**):

$$\text{new } L(\boldsymbol{\theta}) = L(\boldsymbol{\theta}) + \lambda * R(\boldsymbol{\theta})$$

where:
R(**θ**) : **penalty** on the **complexity** of the model
λ : parameter controlling the importance of regularization



Properly adjusted λ would choose the simpler model (green line)

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

L1 and L2 Regularization

L1 regularization:

- $R(\boldsymbol{\theta}) = || \boldsymbol{\theta} ||_1 = \sum_{i=1}^{n} | \theta_i |$

- tends to shrink parameters to zero (sparsity)

- useful for **feature selection**

- called **Lasso regression** when extending linear regression

L2 regularization:

- $R(\boldsymbol{\theta}) = || \boldsymbol{\theta} ||_2^2 = \sum_{i=1}^{n} \theta_i^2$

- tends to shrink coefficients evenly

- useful when we have codependent features (e.g., “gender” and “ispregnant”)

- called **Ridge regression** when extending linear regression

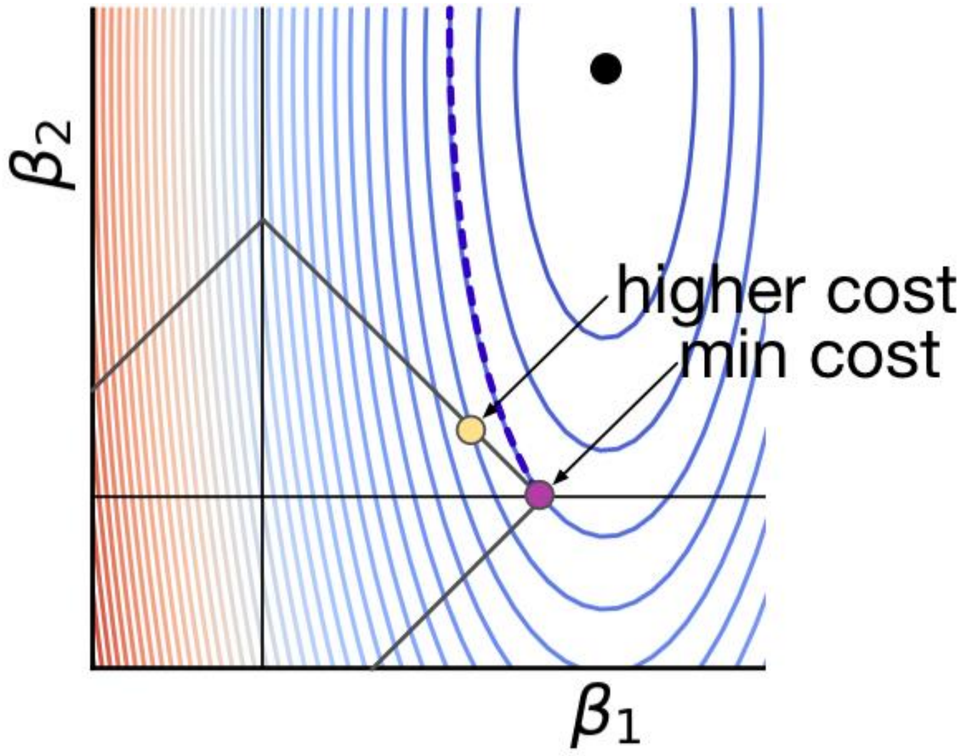
Both can be used with any model

Co-financed by the European Union
Connecting Europe Facility

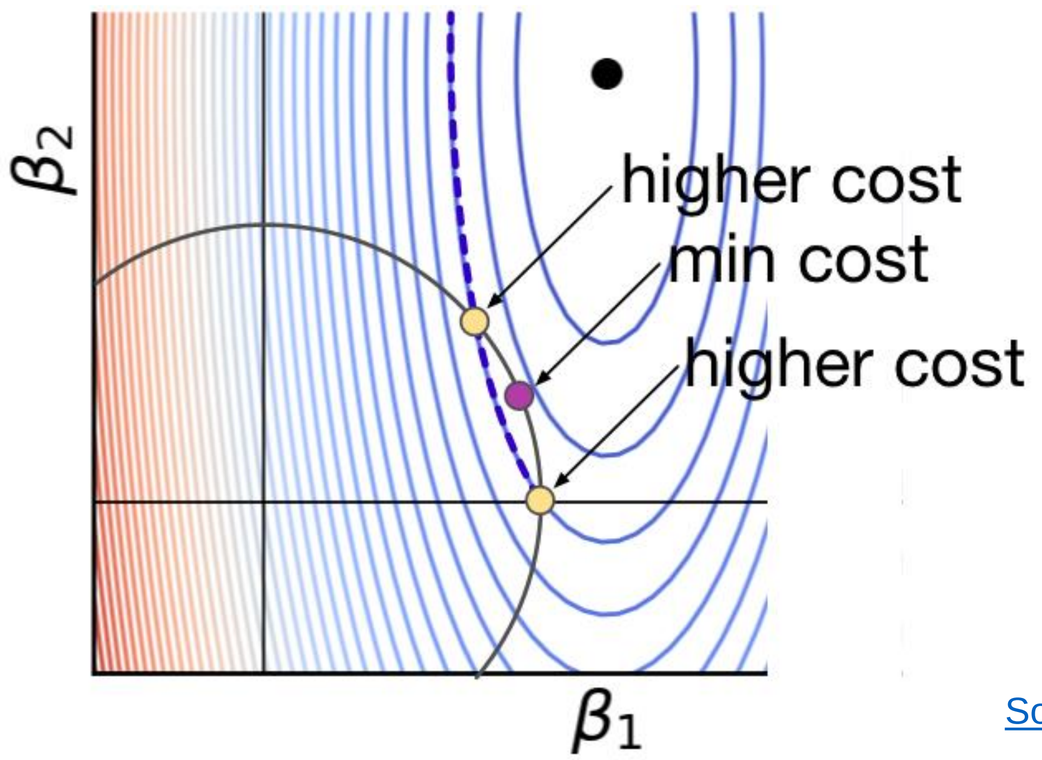
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Why is L1 more likely to zero coefficients than L2?

(a) L1 constraint diamond



(b) L2 constraint circle



Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

L1 vs L2 regularization

L1 regularization:

- Tends to produce solutions that have:
  - few big errors and
  - lots of very insignificant errors
- Distribution of errors will be very “spiky”
- More robust to outliers

L2 regularization:

- Tends to produce solutions that have:
  - very few big errors (they are penalized a lot more)
  - lots of small errors that are still significant
- Distribution of errors will be more “even”
- Produces a better “fit”

Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example

```
In [1]: from sklearn import linear_model
   ...: reg = linear_model.LinearRegression()
   ...: reg.fit([[0, 0], [1, 1], [2, 2]], [0, 1, 2])
Out[1]:
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)

In [2]: from sklearn import linear_model
   ...: reg = linear_model.Ridge(alpha=.5)
   ...: reg.fit([[0, 0], [0, 0], [1, 1]], [0, .1, 1])
Out[2]:
Ridge(alpha=0.5, copy_X=True, fit_intercept=True, max_iter=None,
   normalize=False, random_state=None, solver='auto', tol=0.001)

In [3]: from sklearn import linear_model
   ...: reg = linear_model.Lasso(alpha=0.1)
   ...: reg.fit([[0, 0], [1, 1]], [0, 1])
Out[3]:
Lasso(alpha=0.1, copy_X=True, fit_intercept=True, max_iter=1000,
   normalize=False, positive=False, precompute=False, random_state=None,
   selection='cyclic', tol=0.0001, warm_start=False)
```


This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Model improvement through hyperparameter tuning

Models have hyperparameters:
- k-nearest neighbor regression: k
- polynomial regression: polyomial degree

Learning algorithms have hyperparameters:
- stochastic gradient descent: learning rate, momentum
- regularization coefficient on the loss function

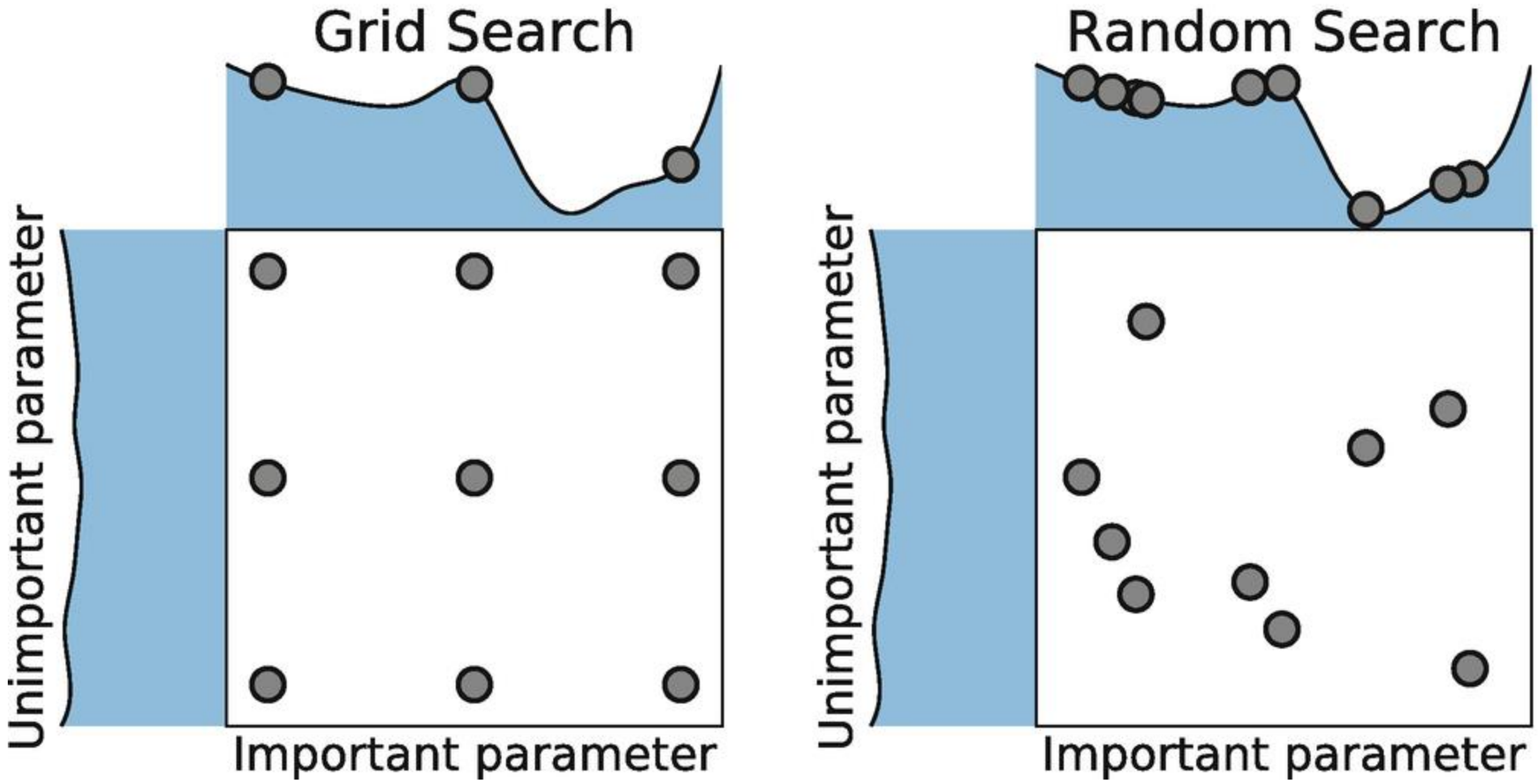
- Hyperparameter settings are important for the performance of a learning algorithm – model combination

- Tune hyperparameters using cross-validation

- Two simple methods:
  - **Grid search**: specify a set of values for each hyperparameter and try all combinations of values
  - **Random search**: randomly sample hyperparameters from specified ranges

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model improvement through hyperparameter tuning



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Hyperparameter tuning in scikit-learn

Grid Search: sklearn.model_selection.GridSearchCV

Random Search: sklearn.model_selection.RandomizedSearchCV

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Model improvement through ensembling

“More heads are better than one”

“Wisdom of the crowd”

Ensembles: train multiple models instead of one to attain better predictive performance

More in the following lecture.

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 6: Trees and Forests

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 6: Trees and Forests

Learning Outcomes

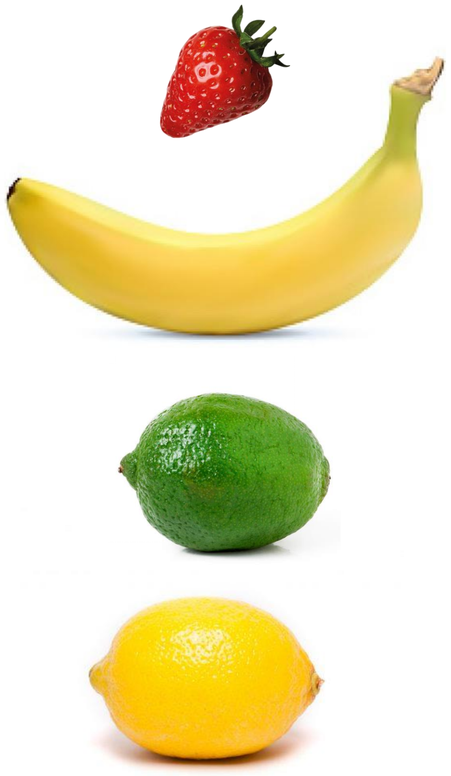
You will understand:

1. how decision tree models work
2. how to train decision trees using the concepts of entropy and information gain
3. how to use continuous variables in decision trees
4. classification and regression trees
5. why ensembles of models can achieve lower generalization errors
6. ensemble methods such as bagging, random forests, boosting, XGBoost and stacking

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Introductory example: fruit image classifier



if most pixels are yellow:
  if shape is oval-like: lemon
  else: banana
else most pixels are red: strawberry
else: lime

Pros:

+ Interpretable

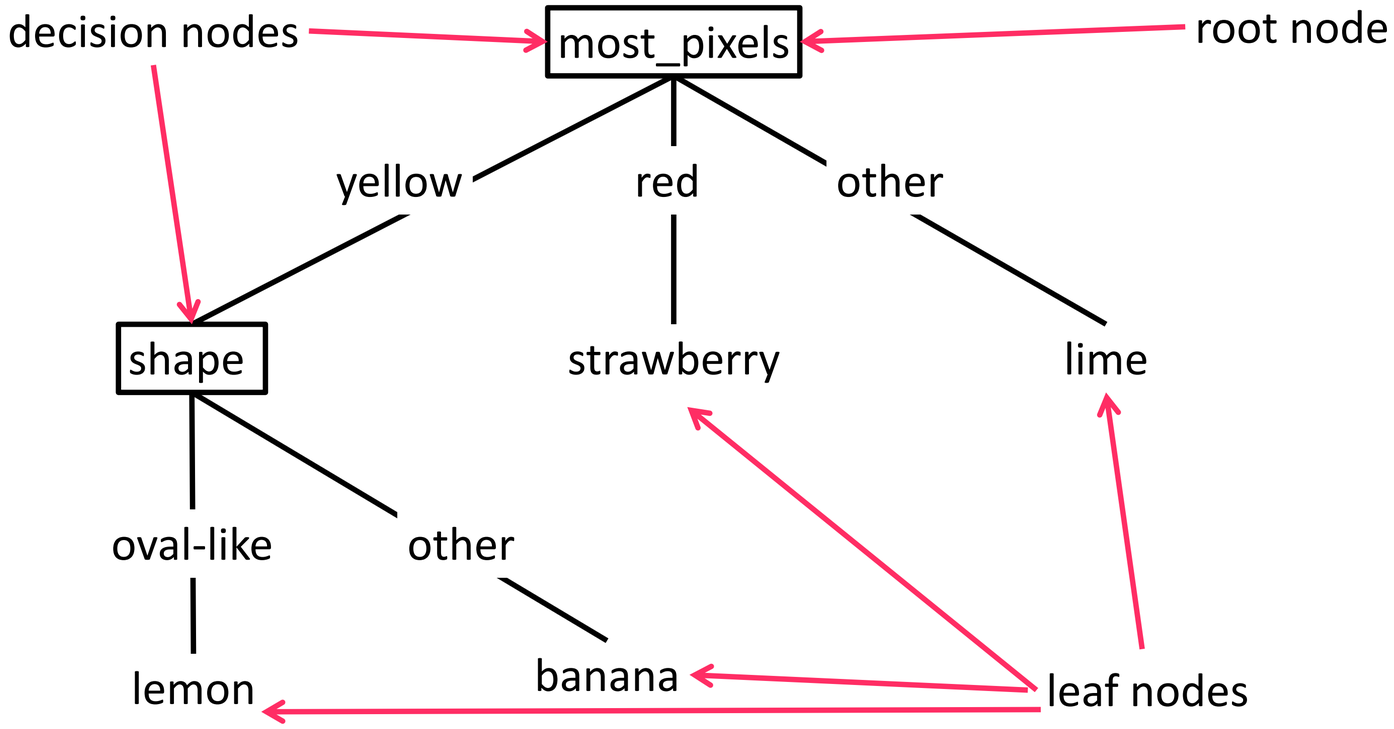
+ Fast prediction

Is there a way to learn such programs from data?

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Decision Tree Representation for Fruit Image Classifier

if most pixels are yellow:
  if shape is oval-like: lemon
  else: banana
else most pixels are red: strawberry
else: lime



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Another Example

Nissi Beach

Ayia Napa

Cyprus

Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Sunburn Data Collected

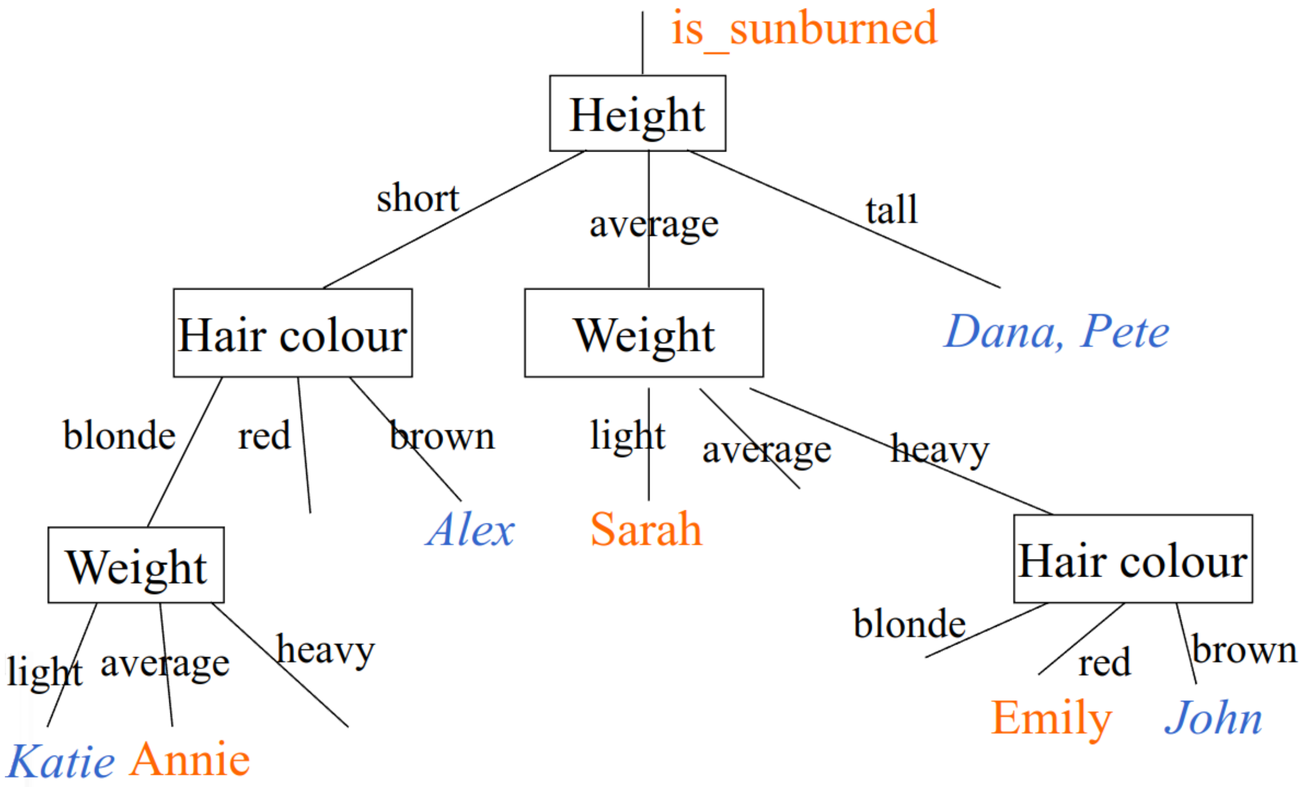
Name	Hair	Height	Weight	Lotion	Result
Sarah	Blonde	Average	Light	No	Sunburned
Dana	Blonde	Tall	Average	Yes	*None*
Alex	Brown	Short	Average	Yes	*None*
Annie	Blonde	Short	Average	No	Sunburned
Emily	Red	Average	Heavy	No	Sunburned
Pete	Brown	Tall	Heavy	No	*None*
John	Brown	Average	Heavy	No	*None*
Kate	Blonde	Short	Light	Yes	*None*

We want to predict whether some person will get a sunburn based on the other features

Slide based on Ata Kaban's Machine Learning course - University of Birmingham.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Decision Tree 1



Sunburn sufferers are...

If height="average" then
- if weight="light" then
  - return(true) ;;; Sarah
- elseif weight="heavy" then
  - if hair_colour="red" then
    - return(true) ;;; Emily

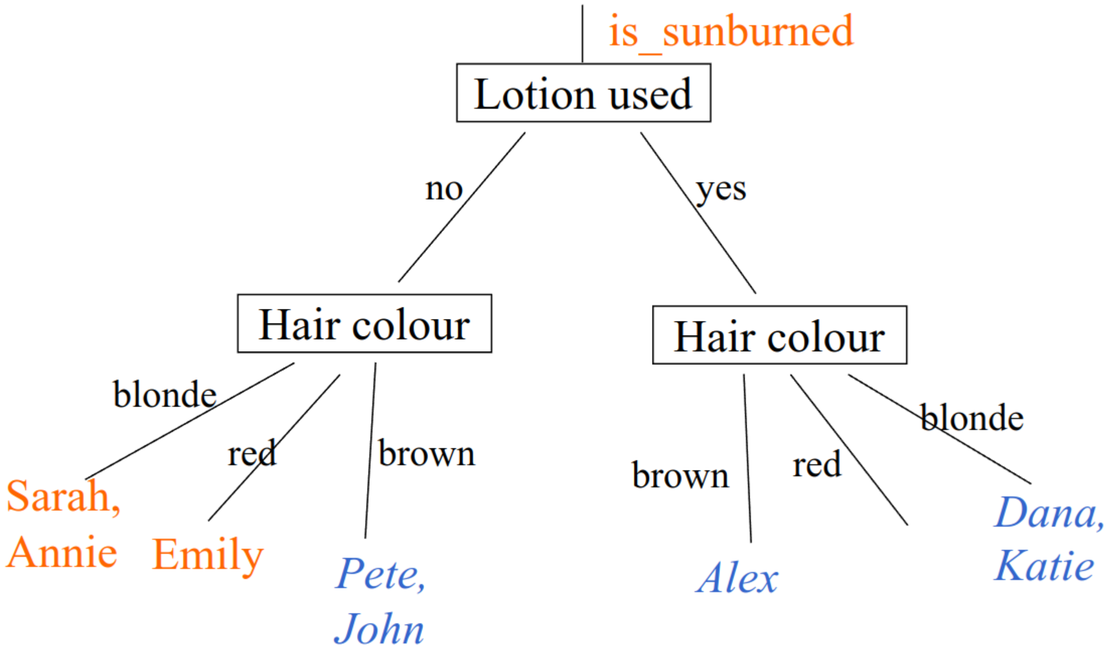
elseif height="short" then
- if hair_colour="blonde" then
  - if weight="average" then
    - return(true) ;;; Annie

else *return(false) ;;;everyone else*

Slide based on Ata Kaban's Machine Learning course - University of Birmingham.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Decision Tree 2

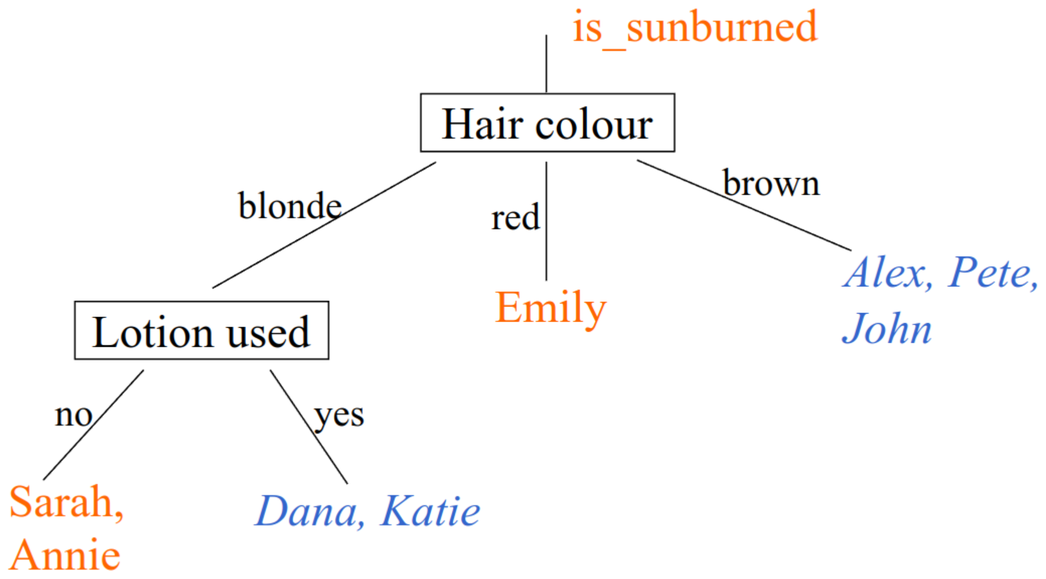


Simpler than the previous decision tree

Slide based on Ata Kaban's Machine Learning course - University of Birmingham.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Decision Tree 3

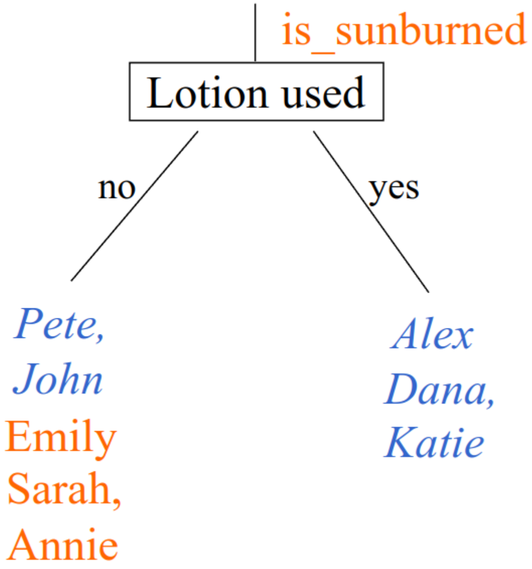


Even simpler than the previous decision tree

Slide based on Ata Kaban's Machine Learning course - University of Birmingham.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

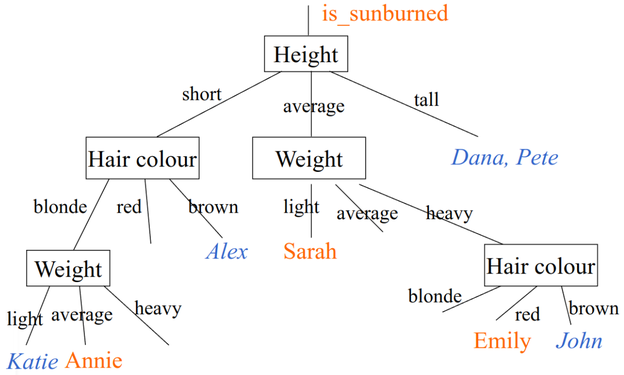
Decision Tree 4

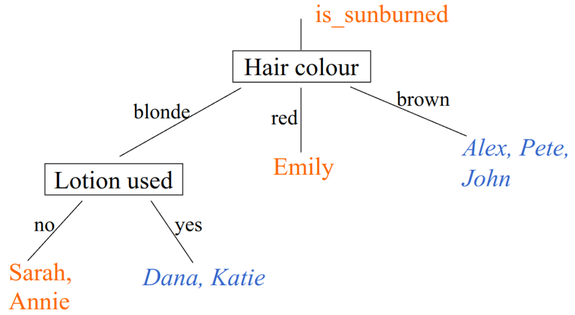


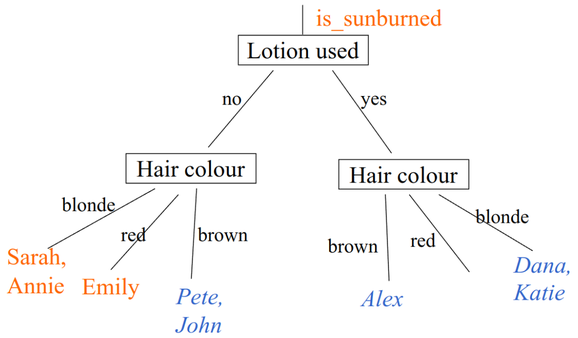
We cannot tell much from this

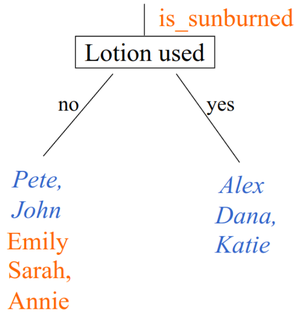
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Many Possibilities









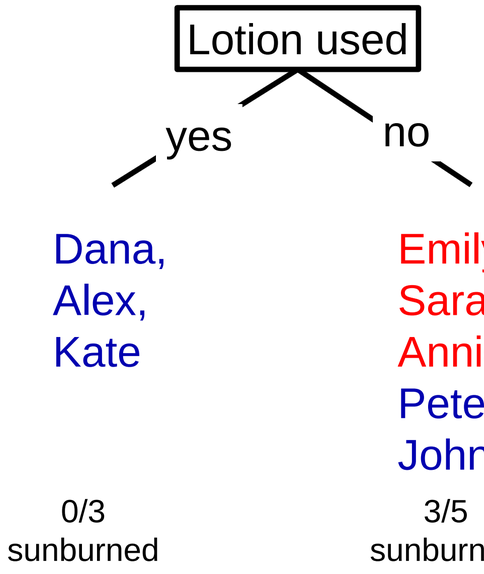
- Among these decision trees some do better than others on the training, validation and test sets.
- The learning algorithm needs to pick one out of all the possibilities in order to do well on the training set and ideally to generalize well on the validation and test sets.

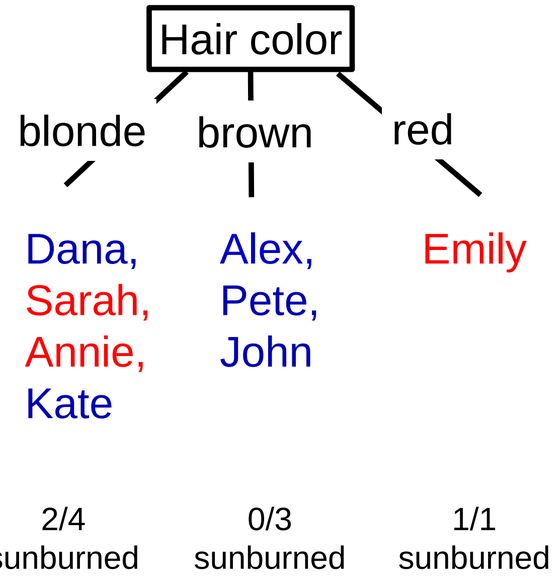
This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

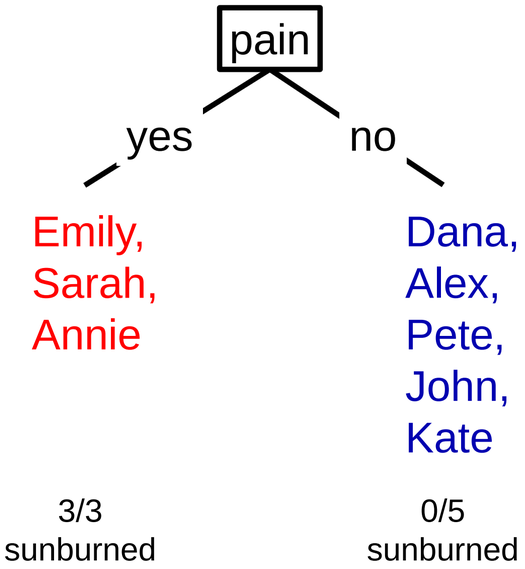
Decision Tree Learning

Decision 1: How to choose what feature to split on at each node

maximize order (or minimize disorder)







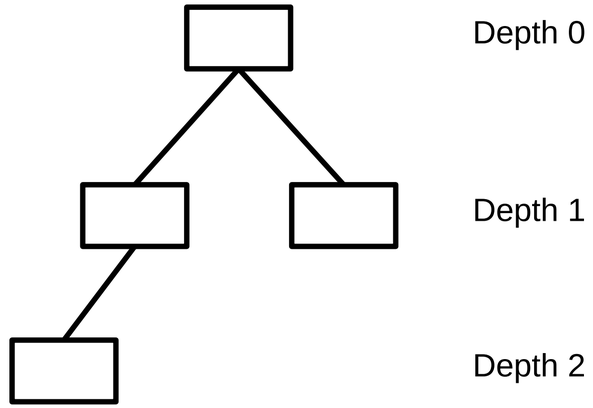
Slide based on Andrew Ng's Machine Learning course - Coursera

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Decision Tree Learning

Decision 2: When do you stop splitting?

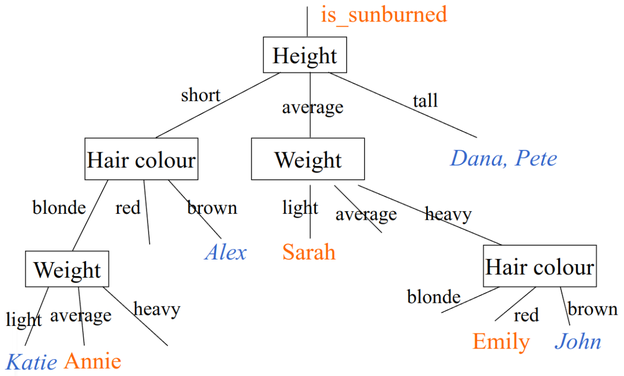
- When a node is 100% one class
- When splitting a node will result in the tree exceeding a maximum depth
- When improvements in ‘order’ score are below a threshold
- When number of examples in node is below a threshold
- When the error in the validation set starts increasing

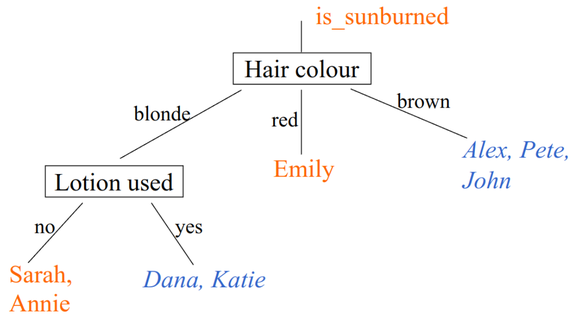


Slide based on Andrew Ng’s Machine Learning course - Coursera

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Decision 1: Choosing the feature to split





- **Irrelevant** features do not classify the data well
- Using irrelevant features thus causes larger decision trees
- A learning algorithm could look for simpler decision trees
- Q: How?

Slide based on Ata Kaban's Machine Learning course - University of Birmingham.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

A: How did WE do it?

- Q: Which is the best feature for splitting up the data?
- A: The one which is **most informative** for the classification we want to get

- Q: What does ‘more informative’ mean?
- A: The feature which best **reduces the disorder**

- Q: How can we measure something like that?
- A: See next slides

Slide based on Ata Kaban’s Machine Learning course - University of Birmingham.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Disorder

- We need a quantity to measure the disorder in a set of examples

$S = \{s_1, s_2, \ldots, s_m\}$

where s_1 ='Sarah', s_2 ='Dana', ...

- What properties should the Disorder (D) have?
  - When D(S)=0 → all examples in S have the same class
  - When D(S)=1 → half of the examples in S are of one class and half are the opposite class
  - We need a function that takes as input the **proportion** of positive examples
    - As the proportion increases from 0 to 0.5, D needs to go from 0 to 1
    - As the proportion increases from 0.5 to 1, D needs to go from 1 to 0

Slide based on Ata Kaban's Machine Learning course - University of Birmingham.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Examples

- D({‘Dana’, ‘Pete’}) = 0
- D({‘Sarah’, ‘Annie’, ‘Emily’}) = 0
- D({‘Sarah’, ‘Emily’, ‘Alex’, ‘John’}) = 1
- D({‘Sarah’, ‘Emily’, ‘Alex’}) = ?

Slide based on Ata Kaban’s Machine Learning course - University of Birmingham.

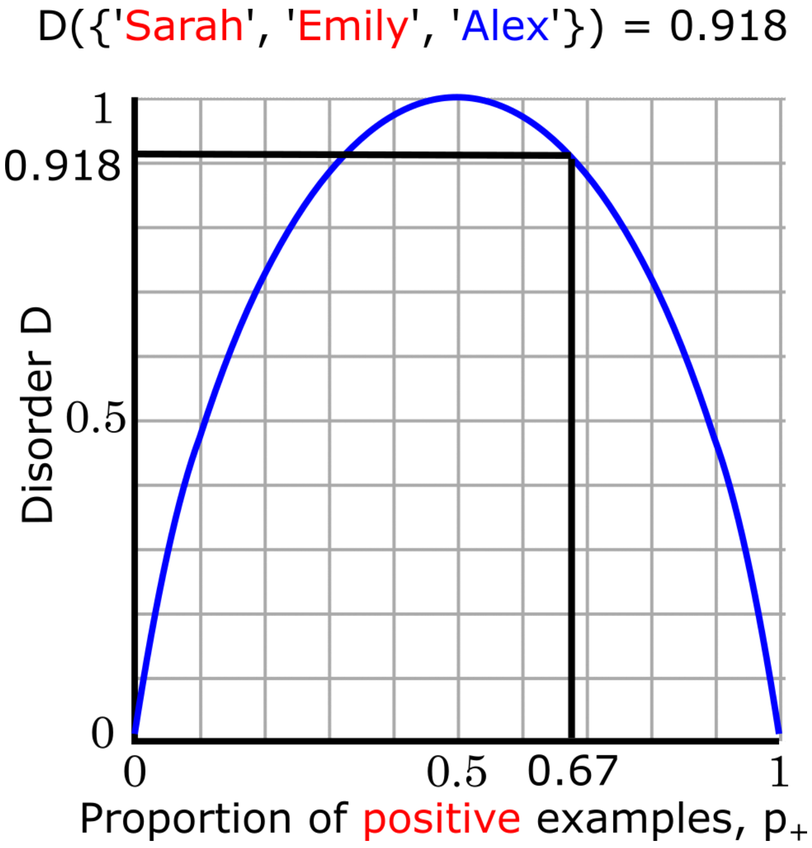
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Disorder function intuition

We need a function that takes as input the **proportion** of positive examples

- As the proportion increases from 0 to 0.5, D needs to go from 0 to 1
- As the proportion increases from 0.5 to 1, D needs to go from 1 to 0



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Entropy as a measure of disorder

The **Entropy** measures the disorder of a set S containing a total m examples of which m_+ are positive and m_- are negative and it is given by

$$H(p_+) = -p_+ log_2 p_+ - p_- log_2 p_-$$

where

$$p_+ = {}^{m_+}/_{m} \quad and \quad p_- = {}^{m_-}/_{m} = 1 - p_+$$

- H(0) = 0
- H(1) = 0
- H(0.5) = 1

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Back to the beach

D({‘Sarah’, ‘Dana’, ‘Alex’, ‘Annie’, ‘Emily’, ‘Pete’, ‘John’, ‘Katie’}) = $H(3/8)$

$$H(3/8) = -\frac{3}{8}log_2\frac{3}{8} - \frac{5}{8}log_2\frac{5}{8}$$

= 0.954

Slide based on Ata Kaban’s Machine Learning course - University of Birmingham.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Minimizing the disorder

- What we know:
  - How to measure the disorder of a set

- What's left:
  - We want to measure how much the disorder of a set would reduce by knowing the value of a particular feature
  - This will help us figure out which feature to use for splitting up the data

Slide based on Ata Kaban's Machine Learning course - University of Birmingham.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Information Gain

The **Information Gain** measures the expected reduction in entropy due to splitting on a feature A

$$Gain(S, A) = D(S) - \underbrace{\sum_{v \in Values(A)} \frac{|S_v|}{|S|} D(S_v)}$$

The **average disorder** is just the weighted sum of the disorders in the branches (subsets) created by the values of A

We want the largest gain

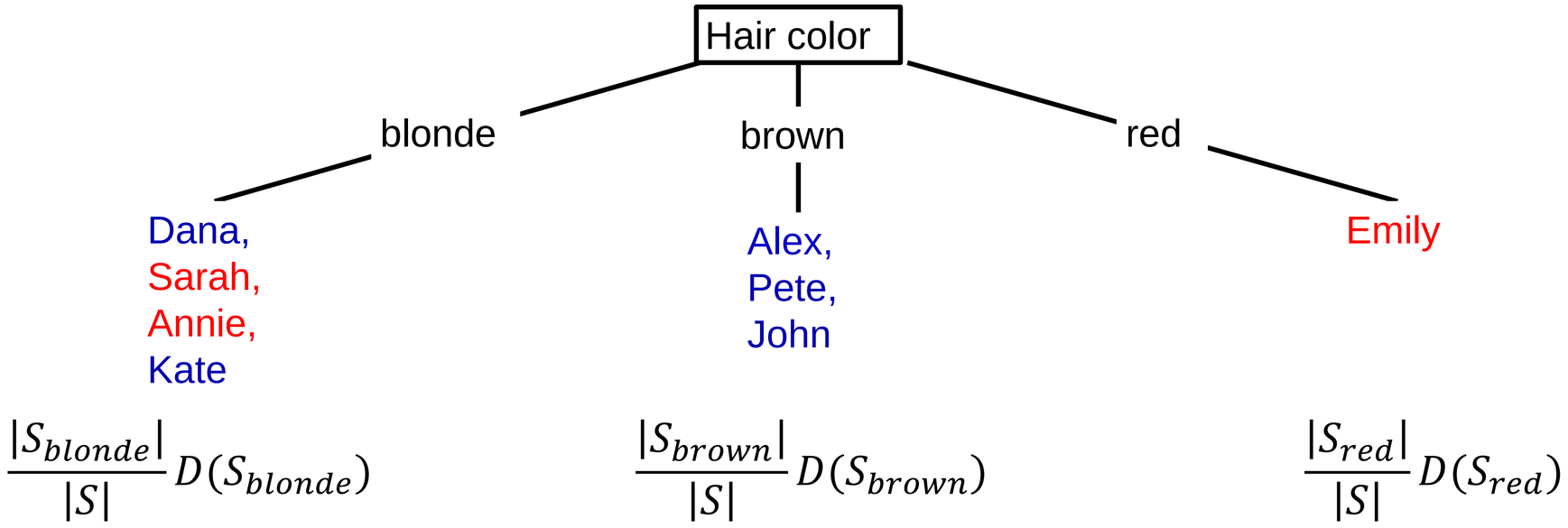
➢ The same as smallest average disorder

Slide based on Ata Kaban's Machine Learning course - University of Birmingham.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

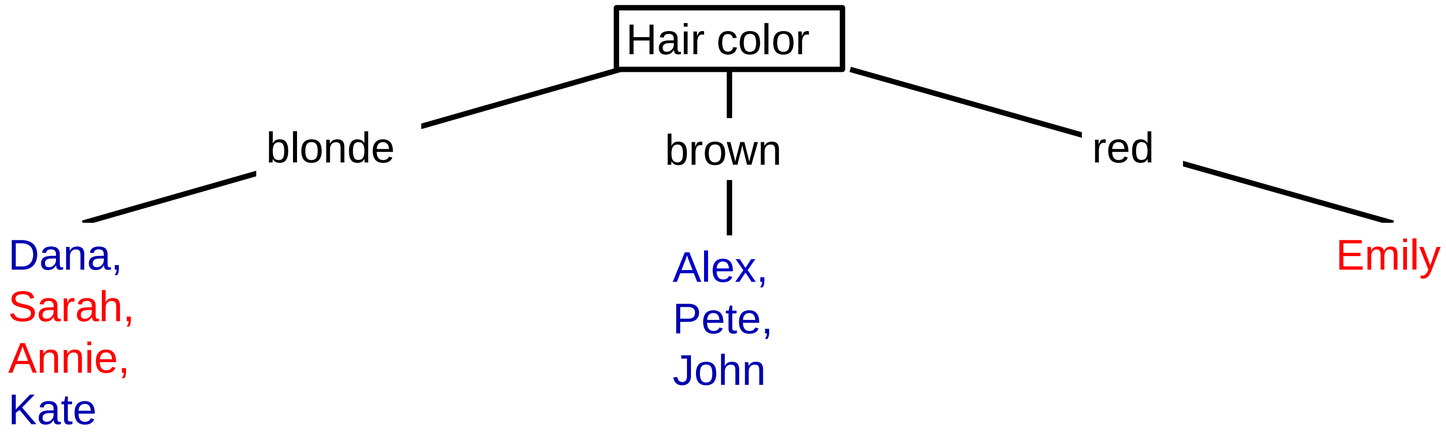
Back to the beach: calculate the average disorder associated with hair color



Slide based on Ata Kaban's Machine Learning course - University of Birmingham.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Back to the beach: calculate the average disorder associated with hair color



$$\frac{4}{8}H(2/4) = 0.5 * 1$$

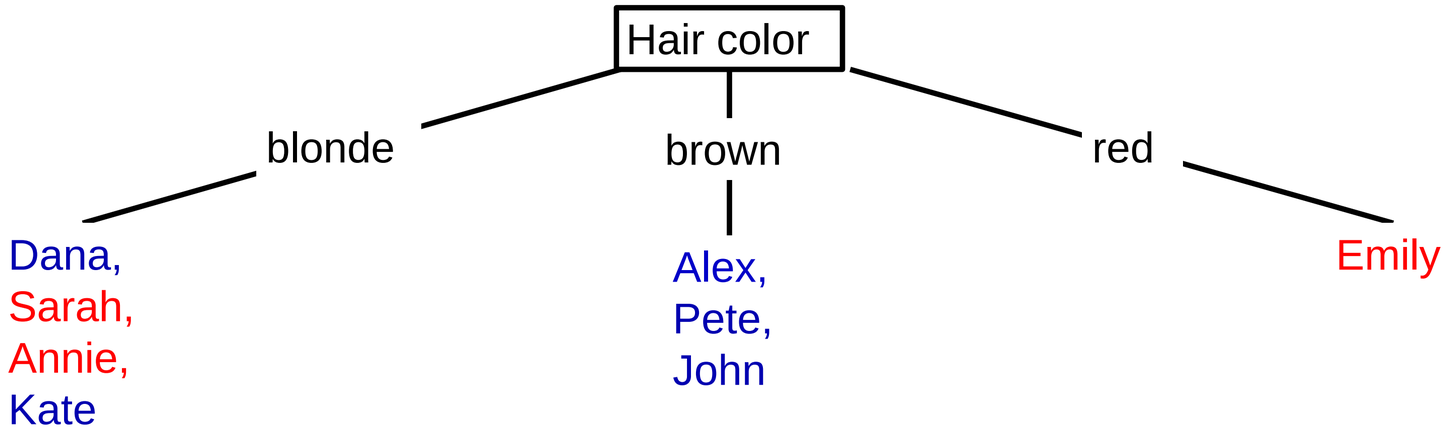
$$\frac{|S_{brown}|}{|S|}D(S_{brown})$$

$$\frac{|S_{red}|}{|S|}D(S_{red})$$

Slide based on Ata Kaban's Machine Learning course - University of Birmingham.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Back to the beach: calculate the average disorder associated with hair color



0.5 $\frac{3}{8}H(0/3) = 0$ $\frac{1}{8}H(1/1) = 0$

So the average disorder created when splitting on hair color is 0.5 + 0 + 0 = 0.5

Slide based on Ata Kaban's Machine Learning course - University of Birmingham.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Which decision variable minimizes the disorder?

Test	Average Disorder	Information Gain
Hair Color	0.5	0.954 – 0.5 = 0.454
Height	0.69	0.954 – 0.69 = 0.264
Weight	0.94	0.954 – 0.94 = 0.014
Lotion used	0.61	0.954 – 0.61 = 0.344

This is what we have just computed

These are the average disorders of the other attributes, computed in the same way

- Which decision variable maximizes the Information Gain then?
- The one which minimizes the average disorder

Hair Color

Slide based on Ata Kaban's Machine Learning course - University of Birmingham.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

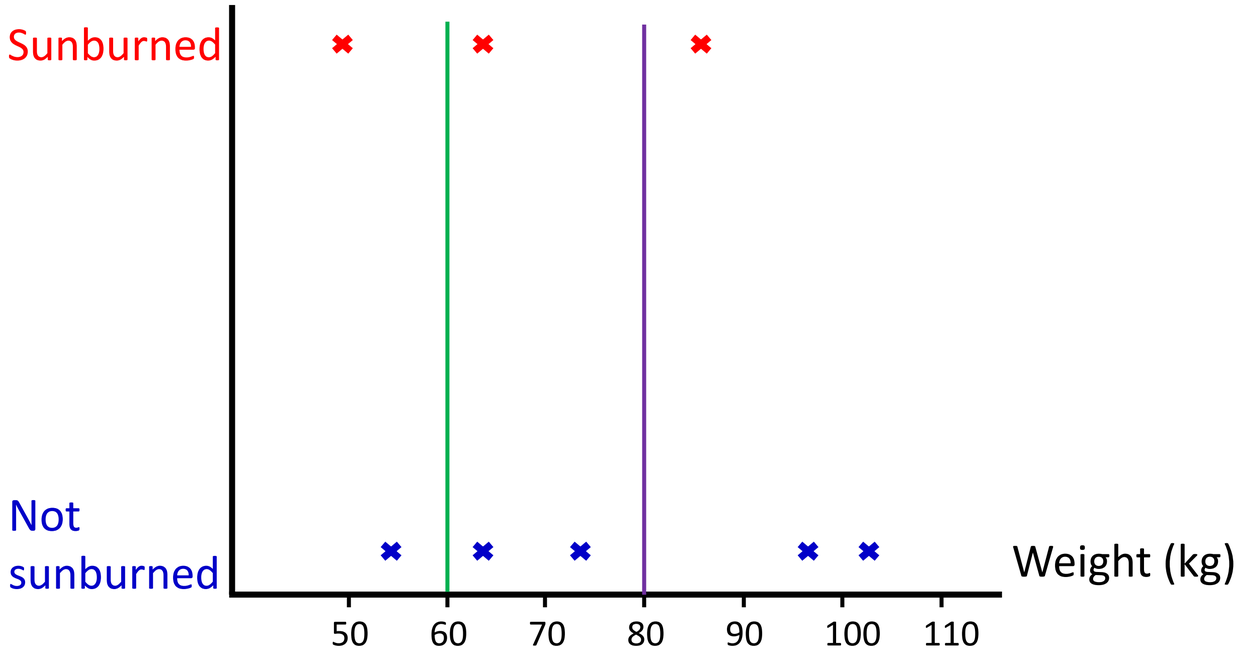
Splitting on a continuous variable

- Suppose that a feature (e.g., weight) is represented with continuous values
- How do we split this feature?

Name	Hair	Height	Weight	Lotion	Result
Sarah	Blonde	Average	49.3 kg	No	Sunburned
Dana	Blonde	Tall	63.6 kg	Yes	*None*
Alex	Brown	Short	72.1 kg	Yes	*None*
Annie	Blonde	Short	65.8 kg	No	Sunburned
Emily	Red	Average	86.7 kg	No	Sunburned
Pete	Brown	Tall	97.3 kg	No	*None*
John	Brown	Average	102.2 kg	No	*None*
Kate	Blonde	Short	53.4 kg	Yes	*None*

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Splitting on a continuous variable



Weight < 60 kg

$$\frac{2}{8}H(1/2) + \frac{6}{8}H(2/6) = 0.94$$

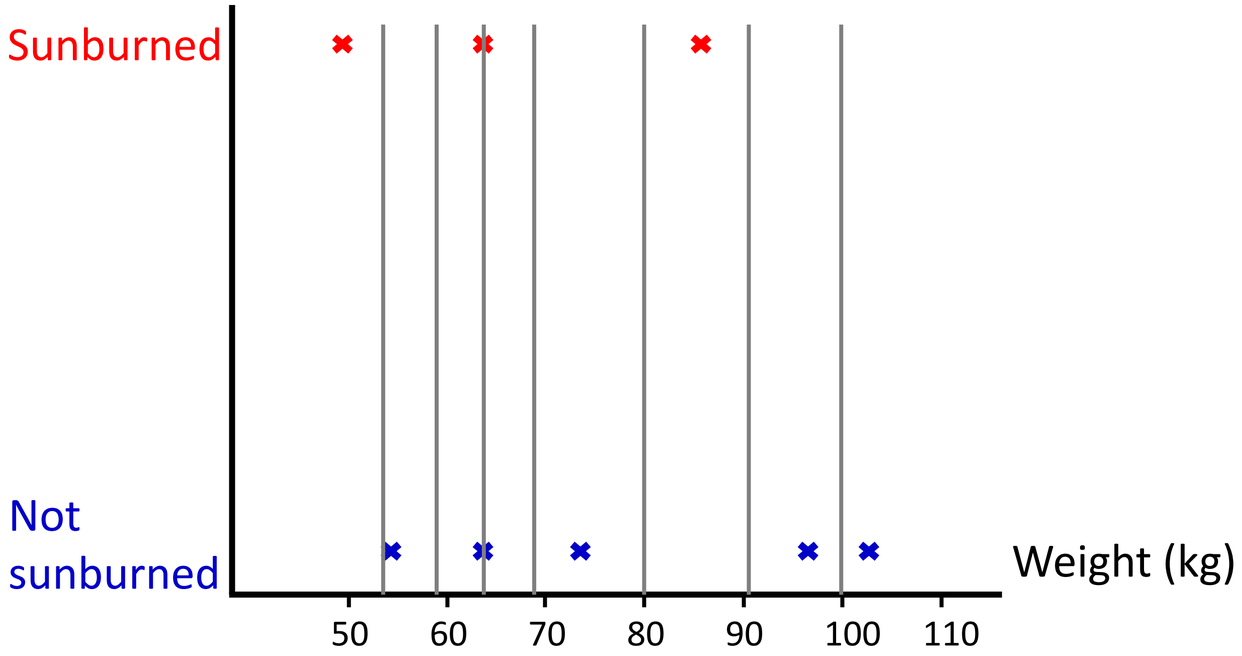
Weight < 80 kg

$$\frac{5}{8}H(2/5) + \frac{3}{8}H(1/3) = 0.95$$

Based on these two splits, which would you choose?

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Splitting on a continuous variable



Procedure

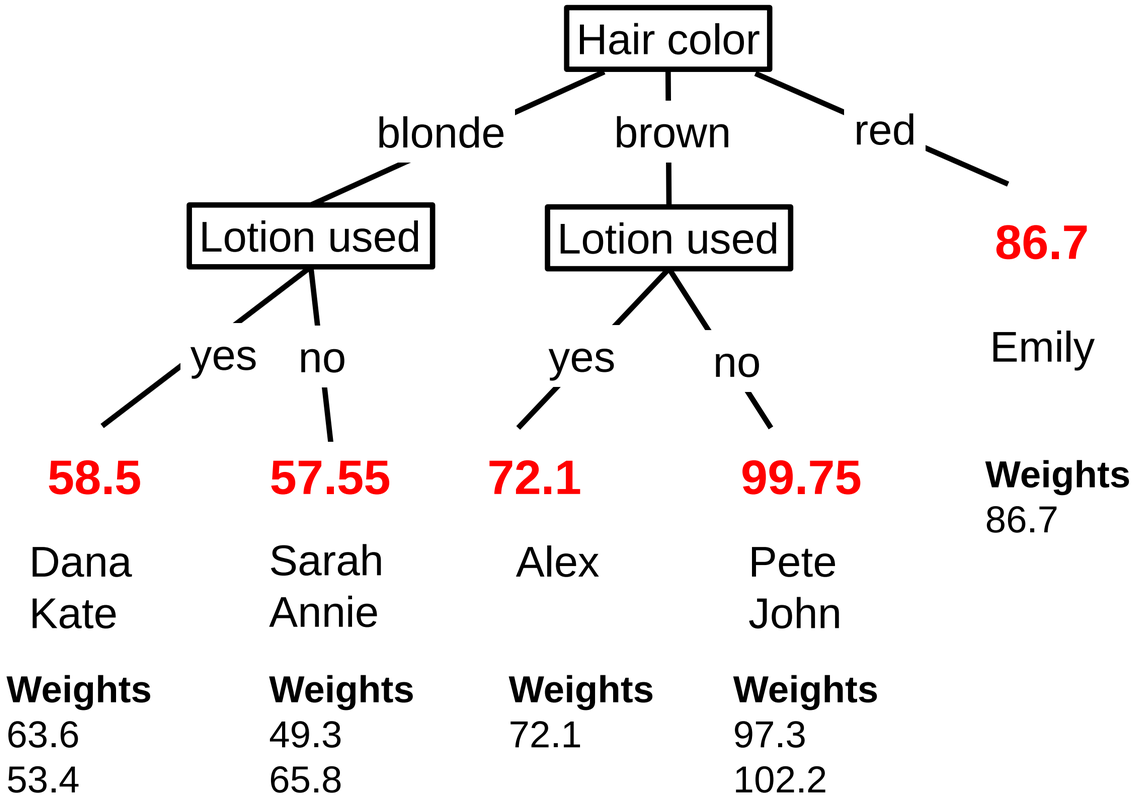
- Get pairs of sorted points and choose the midpoint as the value to split
- Calculate the information gain for all splits
- Select the threshold with the highest information gain

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression Trees

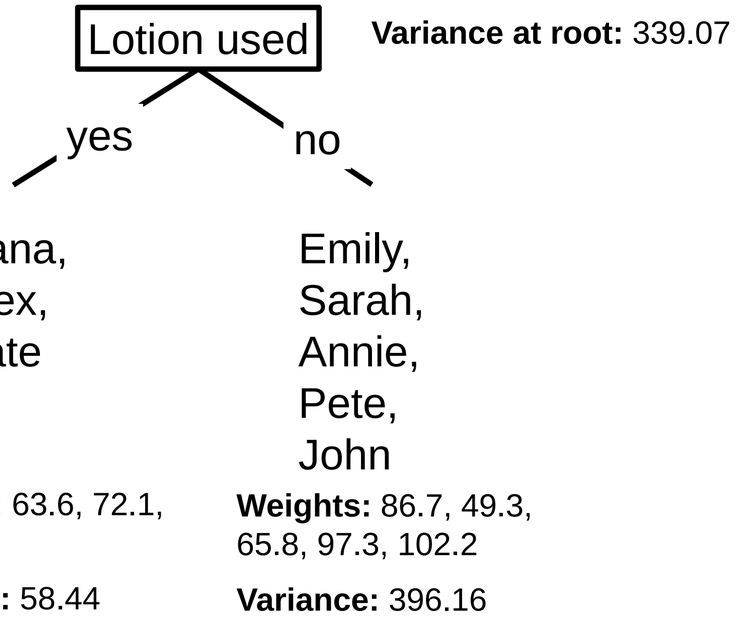
Name	Hair	Height	Weight	Lotion	Result
Sarah	Blonde	Average	49.3 kg	No	Sunburned
Dana	Blonde	Tall	63.6 kg	Yes	None
Alex	Brown	Short	72.1 kg	Yes	None
Annie	Blonde	Short	65.8 kg	No	Sunburned
Emily	Red	Average	86.7 kg	No	Sunburned
Pete	Brown	Tall	97.3 kg	No	None
John	Brown	Average	102.2 kg	No	None
Kate	Blonde	Short	53.4 kg	Yes	None

x_1 x_2 y x_3



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression Trees: choosing a split



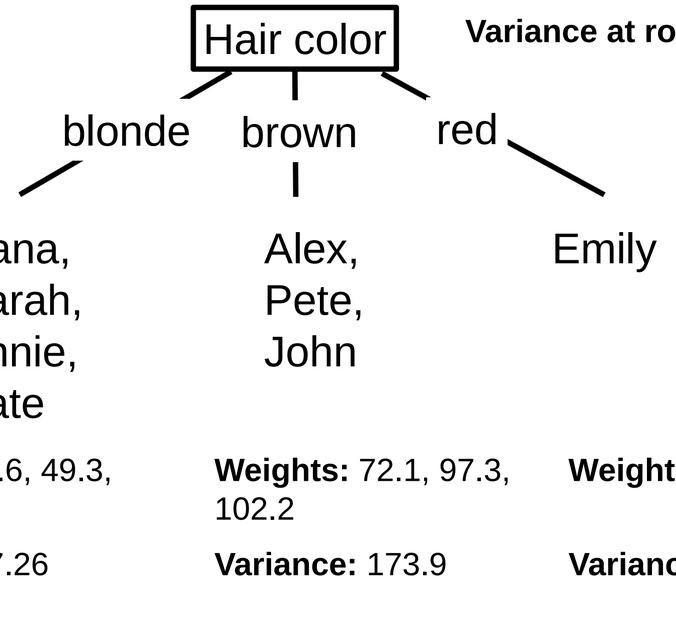
Variance at root: 339.07

Weights: 63.6, 72.1, 53.4

Variance: 58.44

Weights: 86.7, 49.3, 65.8, 97.3, 102.2

Variance: 396.16



Variance at root: 339.07

Weights: 63.6, 49.3, 65.8, 53.4

Variance: 47.26

Weights: 72.1, 97.3, 102.2

Variance: 173.9

Weights: 86.7

Variance: 0.0

$$\frac{3}{8} * 58.44 + \frac{5}{8} * 396.16 = 269.52$$

$$\frac{4}{8} * 47.26 + \frac{3}{8} * 173.9 + \frac{1}{8} * 0.0 = 88.84$$

339.07-269.52 = 69.55 ← **Variance Reduction** → 339.07-88.84 = 250.23

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Ensemble Learning

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Ensemble Learning

“Two (or more) heads are better than one”
“Wisdom of the crowd”

- The judgement of each individual has noise
- Collective opinion (through averaging the responses) is often superior to the one of any individual as it cancels out the effects of this noise

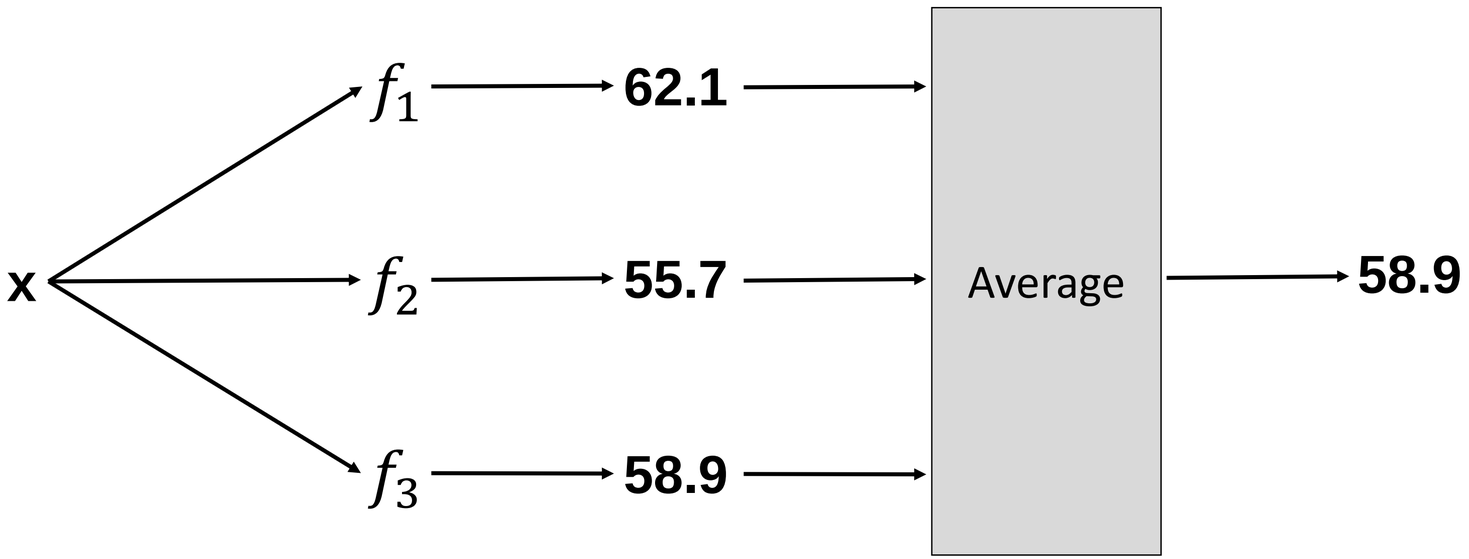
Ensemble methods:

- train multiple (typically weak) models instead of one
- each model will produce different errors at different instances
- diversity helps promote generalization and build a strong learner
- need a way to **combine/aggregate** the output from different models:
  - for regression: take the average
  - for classification: take the majority vote

Co-financed by the European Union
Connecting Europe Facility

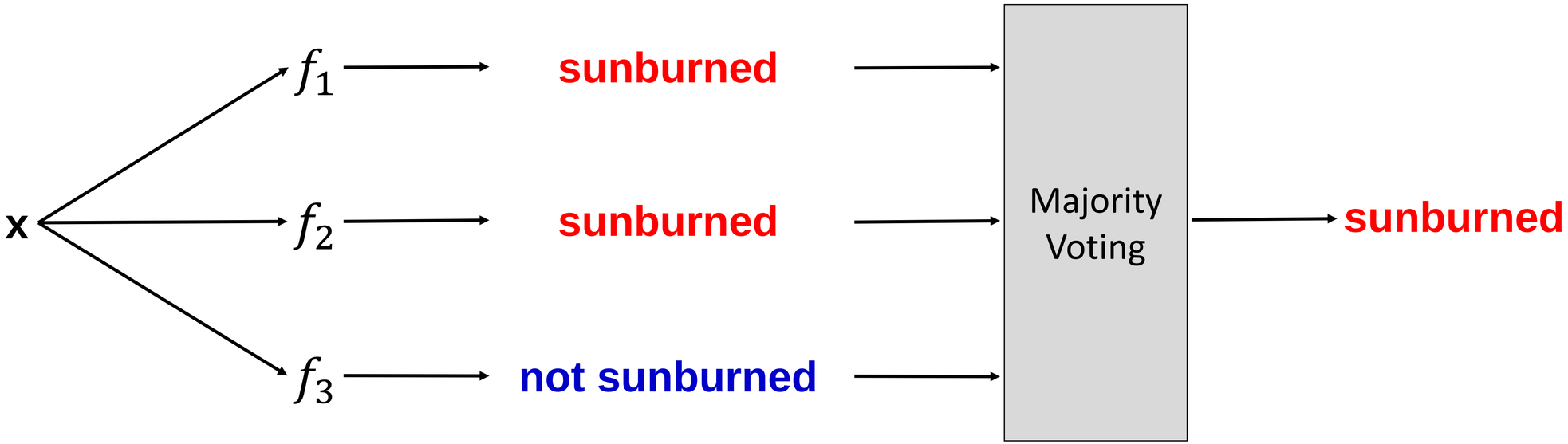
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression Aggregation Example



This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Classification Aggregation Example



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Ensemble Learning Algorithms

- Ensembles tend to produce better results when there is significant diversity among the models
- Ensemble algorithms **seek to promote diversity** among the models they combine

Categories of Algorithms:

- Parallel methods (e.g., Bagging)
- Sequential methods (e.g., Boosting)
- Stacking

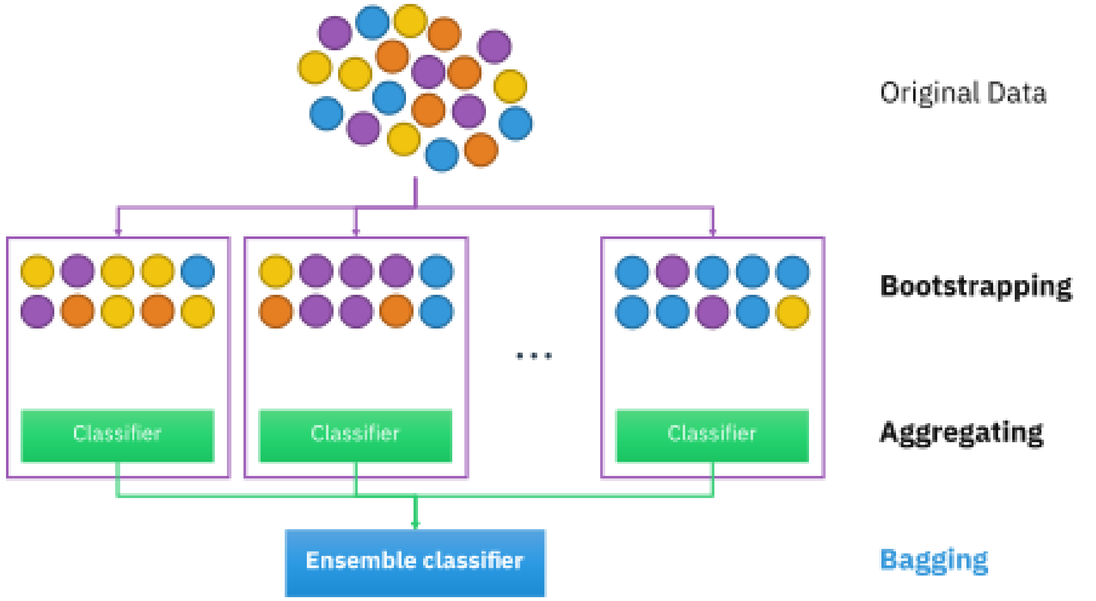
Ensemble learning can be used with any model

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Bagging (Bootstrap aggregating)

- Seeks diversity of the ensemble members **by varying their training data**

- **Boostrapping**: Given a training set D of m instances create B new training sets D_i each of size m' by sampling from D uniformly with replacement
  - This ensures the samples are independent
- **Aggregating**: Fit B models on their respective training sets D_i and combine the results by averaging or voting



Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

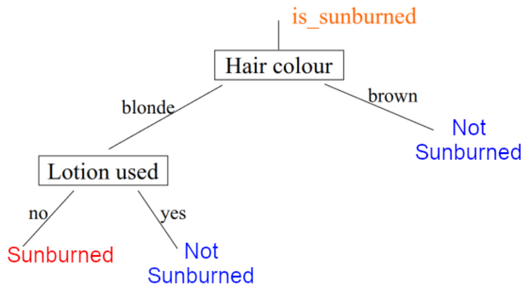
Bagged decision tree

Procedure

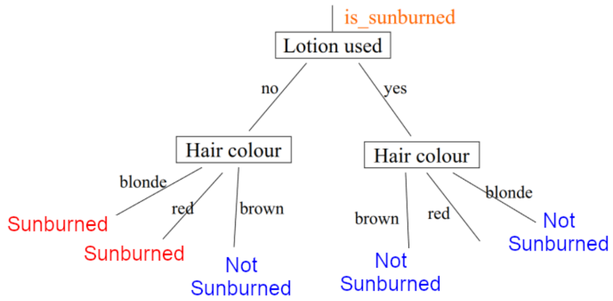
- *Given a training set D of size m*
- *For b = 1 to B*
  - *Create a new training set D_b of size m using sampling with replacement*
  - *Train a decision tree on D_b*
- *Aggregate the results over the B trees*

Setting B to a large number (e.g., 100) results in better performance, however, a very large value (e.g., 1000) slows down computation without offering much improvement.

Name	Hair	Height	Weight	Lotion	Result
Annie	Blonde	Short	Average	No	Sunburned
Dana	Blonde	Tall	Average	Yes	None
John	Brown	Average	Heavy	No	None
Dana	Blonde	Tall	Average	Yes	None
Sarah	Blonde	Average	Light	No	Sunburned
Pete	Brown	Tall	Heavy	No	None
Annie	Blonde	Short	Average	No	Sunburned
Kate	Blonde	Short	Light	Yes	None



Name	Hair	Height	Weight	Lotion	Result
Pete	Brown	Tall	Heavy	No	None
Kate	Blonde	Short	Light	Yes	None
Sarah	Blonde	Average	Light	No	Sunburned
Annie	Blonde	Short	Average	No	Sunburned
Emily	Red	Average	Heavy	No	Sunburned
Emily	Red	Average	Heavy	No	Sunburned
Alex	Brown	Short	Average	Yes	None
Emily	Red	Average	Heavy	No	Sunburned



⋮

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Random Forests

Additional step:

Randomize the feature choice

At each node, when choosing a feature for splitting, if n features are available, pick a random subset of $k < n$ features and allow the algorithm to only choose from that subset

Typically: $k = \sqrt{n}$

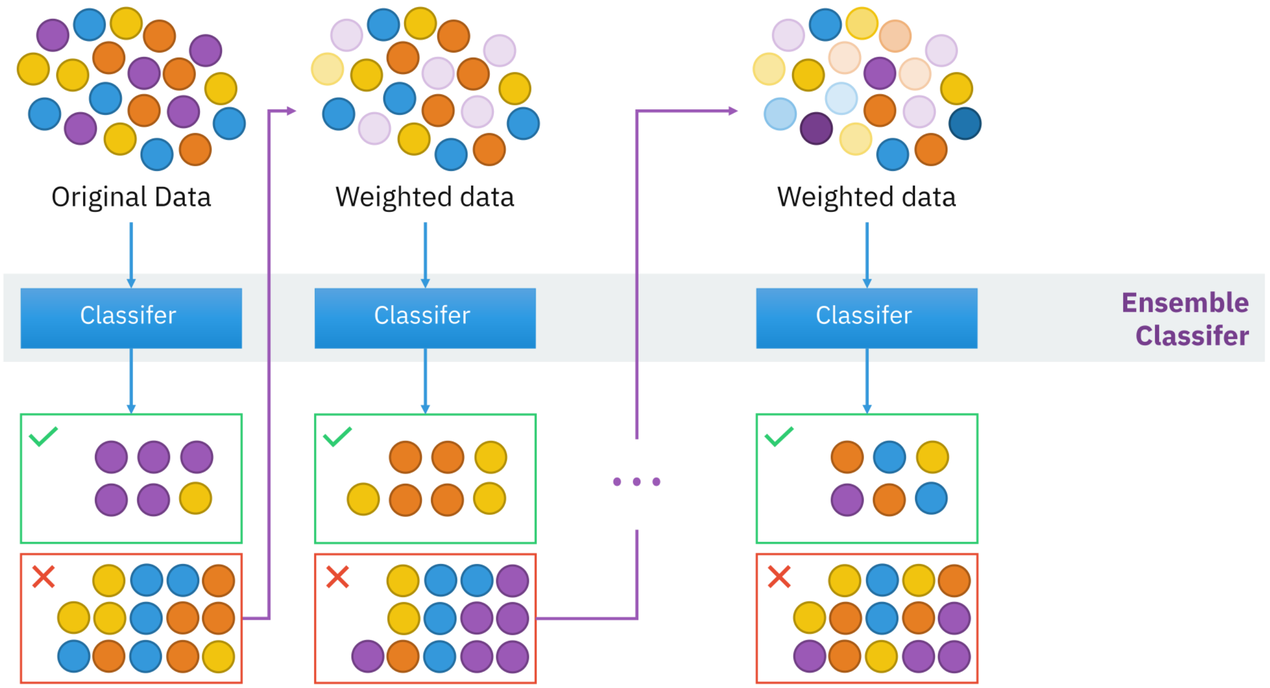
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Boosting

- **Incrementally build the ensemble** where each new model focuses on previously misclassified examples

- Initially, all training instances (D1) are given the same weight
- The training data (D1) are given to a base learner (L1)
- Misclassified instances by L1 now have **higher weight** than correctly classified ones
- Boosted data (D2) are given to a second learner (L2) and so on

- Results are aggregated from all models using a **weighted average**



Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

XGBoost (eXtreme Gradient Boosting)

- Open source software library of boosted trees
- Efficient, scalable and portable implementation
- Built in regularization to prevent overfitting
- Runs on a single machine or on distributed environments
- Very successful and highly competitive in machine learning competitions (e.g., Kaggle)

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

XGBoost Examples

Classification

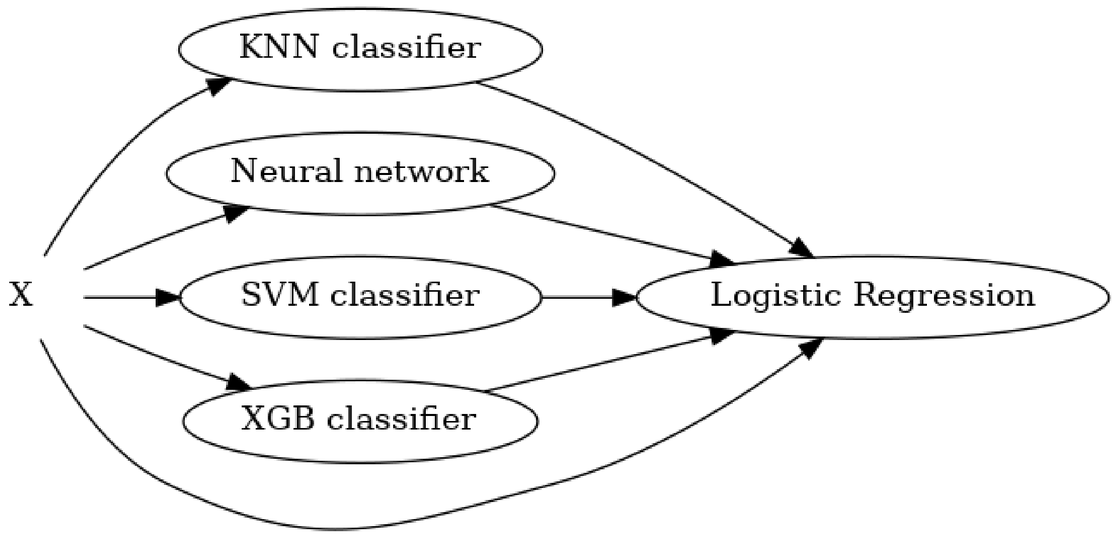
```
from xgboost import XGBClassifier
model = XGBClassifier()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

Regression

```
from xgboost import XGBRegressor
model = XGBRegressor()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```


This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Stacking



Source

- Train multiple models on the training data
- The predictions of the models become the training data of a 2nd level (typically logistic regression) that learns how to combine them

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Next Lecture

- Kernel-based methods

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 7: Kernel-based methods 1

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Revision

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model Evaluation and Improvement

- We want our models to exhibit generalization capabilities instead of memorizing the training set.
- Generalization: good performance on unseen data, drawn from the same distribution
- Modeling simplistic assumptions about the relationship of the data with the target (e.g., using a linear model) might lead to underfitting. In this case we say that the model suffers from high bias.
- On the other hand, if the model is too complex, it fits the noise and does not capture the trend in the data, leading to overfitting. In this case, we say that the model suffers from high variance.
- The bias-variance tradeoff is the conflict of trying to minimize both bias and variance
- The optimal complexity of a model is at the point where the total error ($Bias^2$ + Variance + ε) is at its minimum
- Practically, we achieve that by splitting the dataset into a training, validation and test sets, and selecting the model that has the lowest validation error.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model Evaluation and Improvement

- k-fold cross validation:
  - splits the training+validation dataset into k subsets, and trains k independent models where subset i is used for validation and the remaining as training set
  - the performance of the model is the average validation error over all k subsets
  - used when the dataset is small, because of its complexity in training k models rather than just one.
- Learning curves can be used to inspect whether we need to acquire more data
  - More data typically benefits high variance models but not high bias ones
- Regularization is a penalty given to the loss function of high variance models to reduce the magnitude of their parameters, and thus, their complexity
  - L1 regularization uses the absolute value norm and can sometimes be used for feature selection
  - L2 regularization users the Euclidean norm and typically produces a better fit than L1
- Hyperparameter tuning is the process of varying the hyperparameters of models and learning algorithms, and select the combination that results in the lowest validation error.
- Model improvement can be achieved using ensembles: training multiple models instead of one and aggregating their outputs

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Trees and Forests

- Decision trees are models that have a natural if-then-else structure, thus being interpretable and fast
- For a given dataset, there can be multiple decision trees that classify the data
- An algorithm for learning decision trees needs to choose one that generalizes well by deciding the feature to use for splitting at each node, and when to stop splitting
- Using irrelevant features creates larger decision trees, thus simplicity is preferred
- The best feature to use for splitting is the one that is most informative, i.e., the one that minimizes the disorder aka entropy
- The entropy H takes as input the proportion of positive examples p_+, and as p_+ goes from 0 to 0.5, H increases from 0 to 1, and as p_+ goes from 0.5 to 1, H decreases from 1 to 0.
- Information gain measures the expected reduction in entropy due to splitting on some feature A
  - It is measured as the difference between the entropy of the initial set, and the weighted sum of the entropies in the branches
  - We want to maximize information gain, or equivalently minimize the weighted sum
- To split on a continuous variable, we first calculate all possible unique thresholds (using the midpoints of pairs of sorted points), and select the one with the highest information gain.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Trees and Forests

- Regression trees:
  - predict the average of the values in their leaf nodes
  - use the variance instead of the entropy, and the variance reduction instead of the information gain
- Ensemble methods:
  - typically have lower generalization error than single learners
  - they rely on diverse learners that produce different errors
  - aggregation function: average for regression, majority vote for classification
- Bagging trains models in parallel by varying their training data using sampling with replacement
- Random forests use bagging with the additional step of randomizing the feature choice
- Boosting trains models incrementally by focusing on previously misclassified examples
- Stacking is a method that trains models typically at 2 levels, where the predictions of the models at level 1 become training data for a model at level 2 which learns how to combine them.

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 7: Kernel-based methods 1

Learning Outcomes

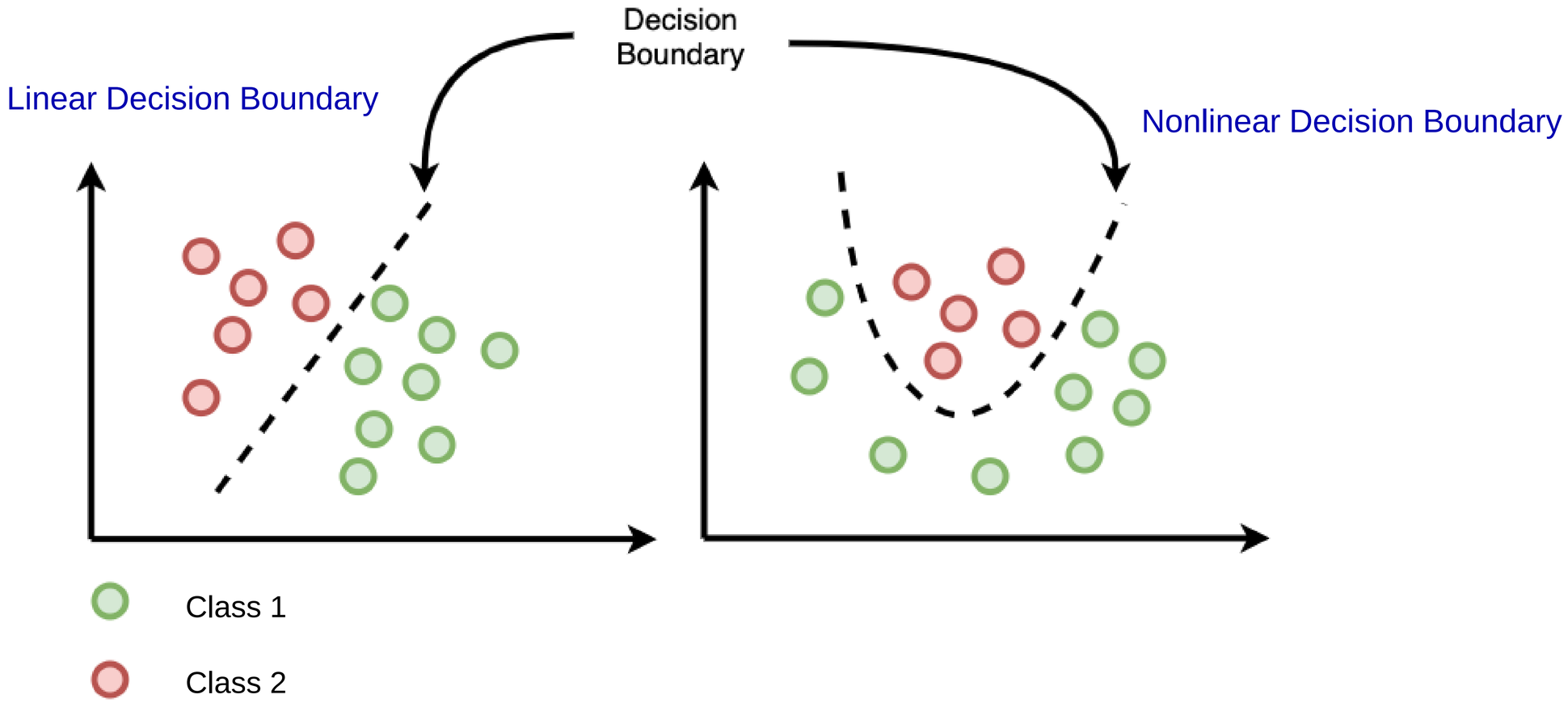
You will understand:

1. The concept of a linear and non-linear decision boundary.
2. What kernel methods are and the kernel trick.
3. Different kernels and how to construct valid kernels.
4. The support vector machine (SVM) algorithm.
5. The concept of the maximum margin classifier.
6. The objective function of SVMs and how it relates to logistic regression.
7. How to create non-linear SVMs using kernels.
8. The basics of Support Vector Regression.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Decision Boundaries



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

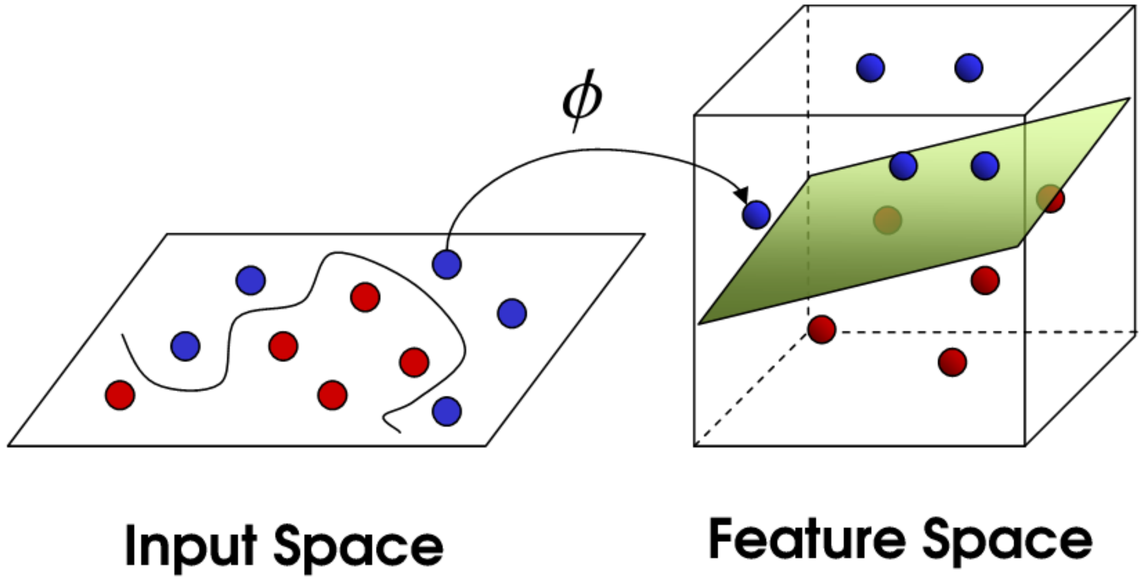
Learning nonlinear decision boundaries

- We can use polynomial features to transform raw data to feature vectors that would allow us to learn nonlinear decision boundaries.
- Examples:
  - 2 variables (x,y) , polynomial degree = 2 : x, y, x^2, xy, y^2
  - 3 variables (x,y,z), polynomial degree = 2 : x, y, z, x^2, xy, xz, y^2, yz, z^2

- **Observation**: Each new feature increases the dimensionality of the input by one which could make the problem “easier” to be solved in the new feature space

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Linear separability in higher dimensions



Source

Adding a new feature dimension may be enough to linearly separate the two classes.

Cover's Theorem on the Separability of Patterns:

"a complex classification problem cast in high-dimensional space nonlinearly is more likely to be *linearly separable* than in a low-dimensional space" (Cover, 1965).

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Learning nonlinear decision boundaries

- We can use polynomial features to transform raw data to feature vectors that would allow us to learn nonlinear decision boundaries.
- Examples:
  - 2 variables (x,y) , polynomial degree = 2 : x, y, x^2, xy, y^2
  - 3 variables (x,y,z), polynomial degree = 2 : x, y, z, x^2, xy, xz, y^2, yz, z^2

- However, we have a **combinatorial blowup** in the number of parameters to be learned.
- **Is there a more efficient way to learn nonlinear decision boundaries?**

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel Methods

- In many ML algorithms, raw data need to be explicitly transformed into feature vectors via a user-specified **feature map**.
- In contrast, kernel methods require only a user-specified **kernel**, i.e., a **similarity function over pairs of raw data points**.
- This enables them to operate in a high-dimensional, **implicit feature space** without computing the coordinates of the data in that space.
- They compute the inner products between the images of all pairs of training data in the feature space: often computationally cheaper than the explicit computation of coordinates.
- This is known as the “**kernel trick**”.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

The Kernel Trick

- Allows to operate in the original feature space without computing the coordinates of the data in the higher dimensional space
- Example: 2D input space, 3D feature space

$$
\begin{aligned}
k(\mathbf{x}, \mathbf{z}) &= \left(\mathbf{x}^{\mathrm{T}}\mathbf{z}\right)^2 = (x_1 z_1 + x_2 z_2)^2 \\
&= x_1^2 z_1^2 + 2 x_1 z_1 x_2 z_2 + x_2^2 z_2^2 \\
&= (x_1^2, \sqrt{2} x_1 x_2, x_2^2)(z_1^2, \sqrt{2} z_1 z_2, z_2^2)^{\mathrm{T}} \\
&= \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{z}).
\end{aligned}
$$

$$\phi(\mathbf{x}) = (x_1^2, \sqrt{2} x_1 x_2, x_2^2)^T$$

$$\phi(\mathbf{z}) = (z_1^2, \sqrt{2} z_1 z_2, z_2^2)^T$$

$O(n^2)$

Computing $\overbrace{\phi(\mathbf{x}), \phi(\mathbf{z}) \text{ and } \phi(\mathbf{x})^T \phi(\mathbf{z})}$

is more computationally expensive than

computing $\underbrace{(\mathbf{x}^T \mathbf{z})^2}$

$O(n)$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernels

- Polynomial kernel

$$k(\boldsymbol{x}, \boldsymbol{z}) = (\boldsymbol{x}^T \boldsymbol{z} + c)^d$$

- Gaussian or Radial Basis Function (RBF) kernel

$$k(\boldsymbol{x}, \boldsymbol{z}) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{z}\|^2}{2\sigma^2}\right)$$

$$k(\boldsymbol{x}, \boldsymbol{z}) = \exp(-\gamma\|\boldsymbol{x} - \boldsymbol{z}\|^2)$$

where $\gamma = \frac{1}{2\sigma^2}$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Making new kernels

- We can make new kernels from valid kernels by allowed operations
- For example, addition, multiplication, rescaling of kernels gives a proper kernel as long as the resulting Gram matrix is symmetric, positive semi-definite

$$k(\boldsymbol{x}, \boldsymbol{z}) = k_1(\boldsymbol{x}, \boldsymbol{z}) + k_2(\boldsymbol{x}, \boldsymbol{z})$$

$$k(\boldsymbol{x}, \boldsymbol{z}) = k_1(\boldsymbol{x}, \boldsymbol{z}) k_2(\boldsymbol{x}, \boldsymbol{z})$$

$$k(\boldsymbol{x}, \boldsymbol{z}) = \lambda\, k_1(\boldsymbol{x}, \boldsymbol{z}) \qquad , \qquad \lambda > 0$$

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Quiz

Given that k_1 and k_2 are proper kernels, decide which of the following formulae define proper kernels:

$k_3(\boldsymbol{x}, \boldsymbol{z}) = 5k_1(\boldsymbol{x}, \boldsymbol{z}) + 3k_2(\boldsymbol{x}, \boldsymbol{z})$

Proper

$k_4(\boldsymbol{x}, \boldsymbol{z}) = -k_1(\boldsymbol{x}, \boldsymbol{z})k_2(\boldsymbol{x}, \boldsymbol{z})$

Not proper

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel Methods (or Kernel Machines)

- They are **instance-based** learners:
  - they learn a weight, w_i, for each training example $(\boldsymbol{x}_i, y_i)$
  - prediction for unlabeled inputs is done by applying the similarity function k (i.e., the kernel) between the unlabeled input $\boldsymbol{z}$ and each of the training inputs $\boldsymbol{x}_i$
  - for instance, a kernelized binary classifier typically computes a weighted sum of similarities:

$$\hat{y} = \operatorname{sgn} \sum_{i=1}^{m} w_i y_i k(\boldsymbol{x}_i, \boldsymbol{z})$$

where $\hat{y} \in \{-1, +1\}$, $k: X \times X \to \mathbb{R}$, $\operatorname{sgn} x := \begin{cases} -1 \ \ if \ x < 0, \\ +1 \ \ if \ x \geq 0. \end{cases}$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Support Vector Machines

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Support Vector Machines

- Powerful supervised learning models for classification

- Two key ideas
  - Assuming linearly separable classes, learn separating hyperplane with maximum margin
    - Maximum margin: minimal generalization error

  - Use kernels to expand input into high-dimensional space to deal with linearly non-separable cases

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Separating Hyperplane

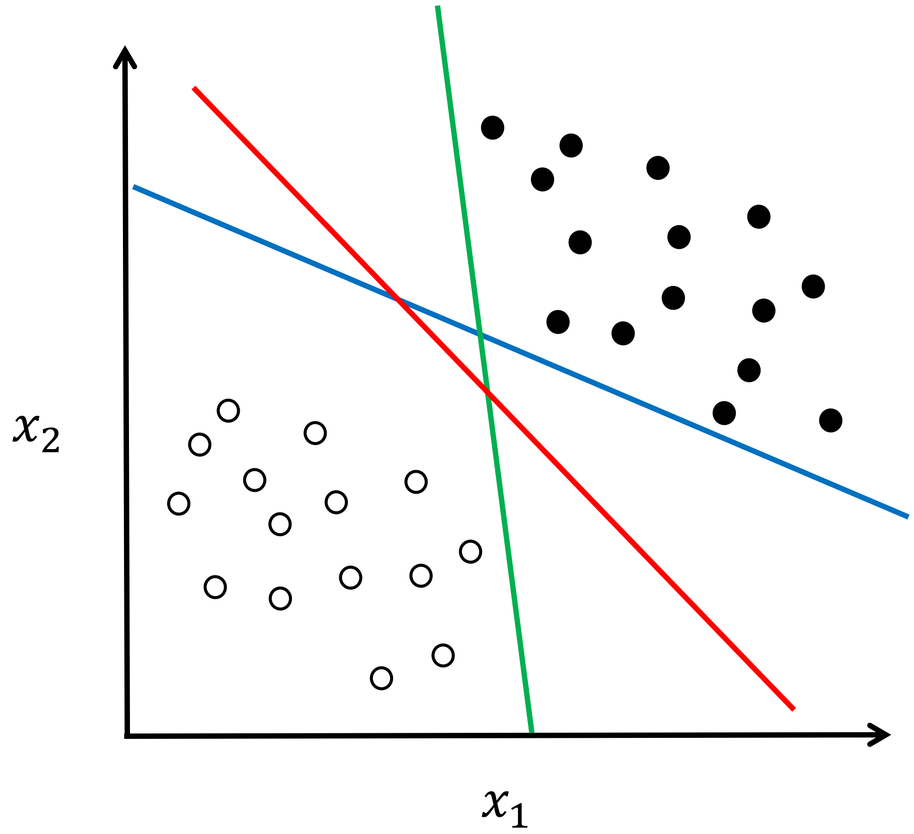
- Training set: $\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)$,

$i = 1,2, \ldots, m;\ y^{(i)} \in \{+1, -1\}$

- Hyperplane: $\boldsymbol{\theta}^T \boldsymbol{x} = 0$

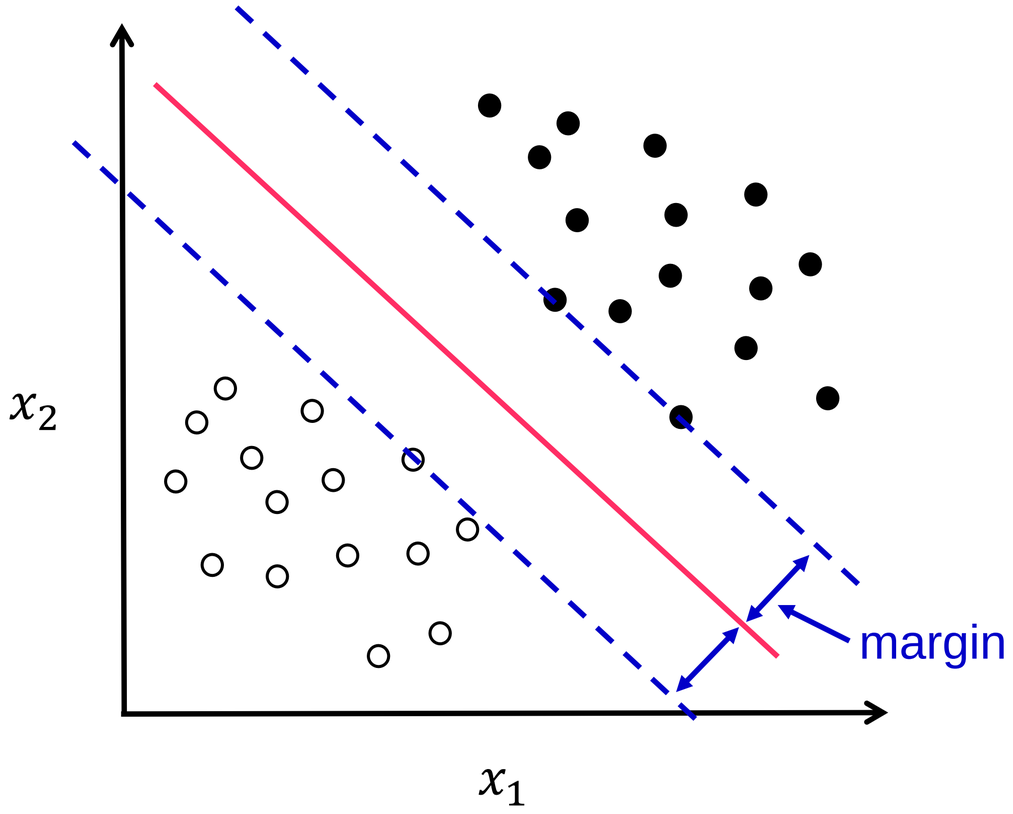
$[\theta_0, \theta_1, \cdots, \theta_n]^T$ $\quad$ $[1, x_1, \cdots, x_n]^T$

Which line would you choose?



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

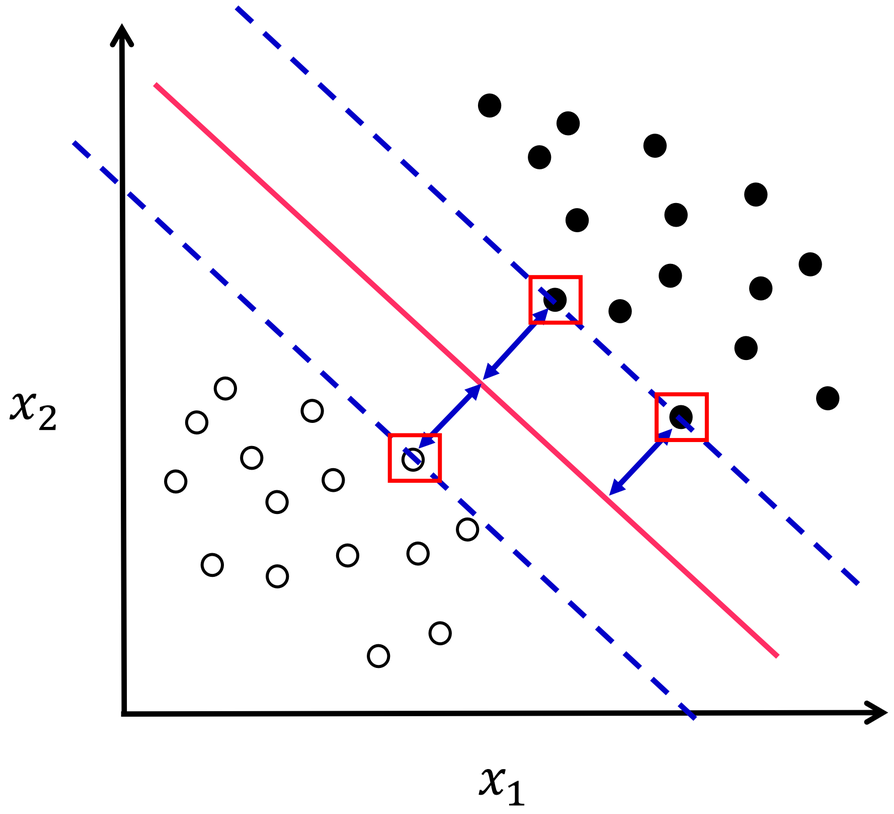
Maximum Margin



- We need to be able to correctly separate new data points
- According to theorem from Learning Theory, from all possible linear decision boundaries the one that maximizes the margin of the training set will minimize the generalization error.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Support Vectors

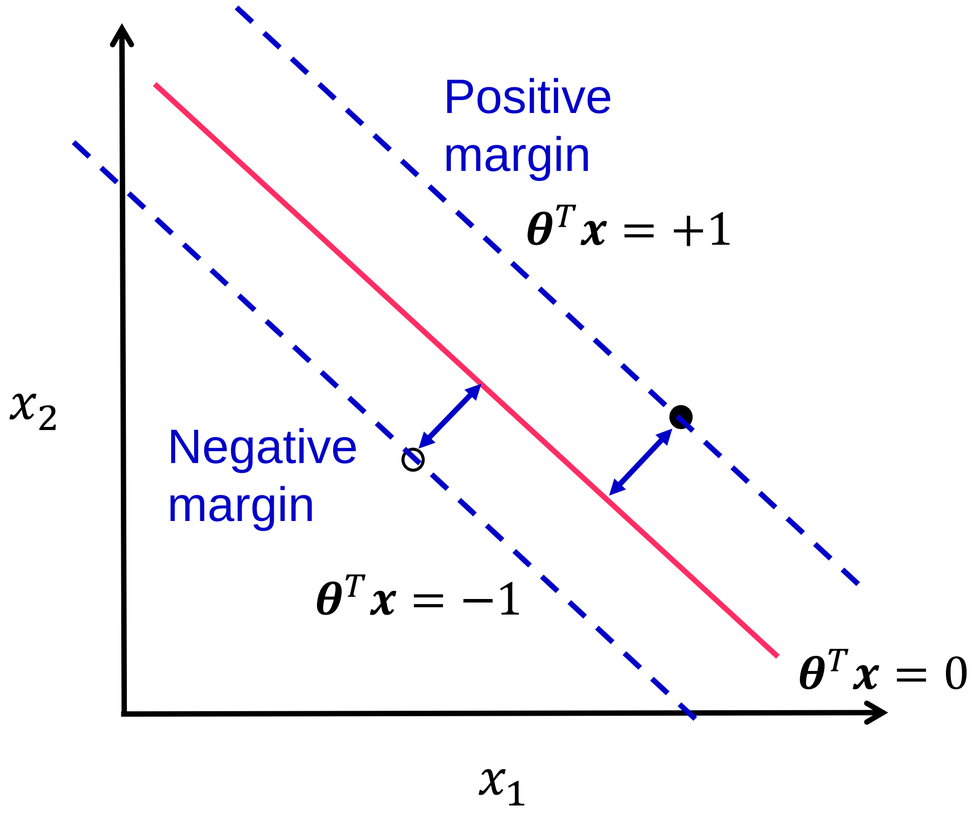


- The training points that are closest to the separating hyperplane are called support vectors.
- These are the most important points as they determine which hyperplane will be chosen by the optimization procedure.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Maximum margin



If $y = 1$, we want $\boldsymbol{\theta}^T\boldsymbol{x} \geq 1$ (not just ≥ 0)

If $y = -1$, we want $\boldsymbol{\theta}^T\boldsymbol{x} \leq -1$ (not just < 0)

We set the margin to a fixed value and want to constrain the data points to lie beyond that margin

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Optimization Objective

$$\min_{\theta} \; C \sum_{i=1}^{m} loss\left(\boldsymbol{\theta}^T \boldsymbol{x}^{(i)}, y^{(i)}\right) + \frac{1}{2} || \boldsymbol{\theta} ||_2^2$$

where:

$$loss\left(\boldsymbol{\theta}^T \boldsymbol{x}^{(i)}, y^{(i)}\right) := \max\left(0, 1 - y^{(i)} \boldsymbol{\theta}^T \boldsymbol{x}^{(i)}\right) \longrightarrow$$

Hinge Loss
(non-differentiable)

$$f_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) = \text{sgn}\left(\boldsymbol{\theta}^T \boldsymbol{x}^{(i)}\right) = \begin{cases} +1 & if \quad \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \geq 0 \\ -1 & otherwise \end{cases}$$

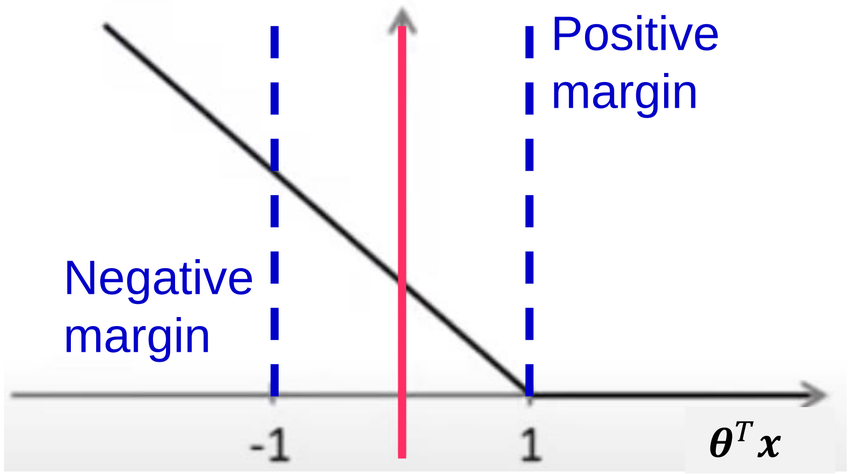
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

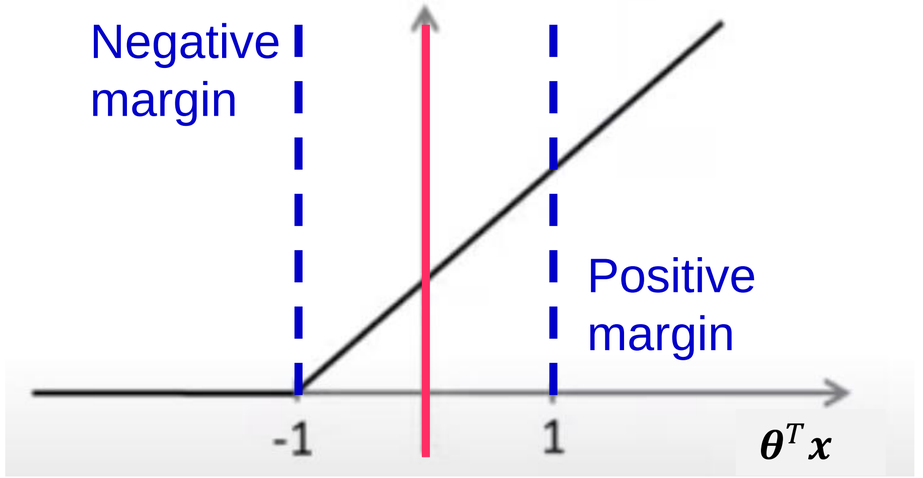
Hinge Loss

$$\max(0, 1 - y\boldsymbol{\theta}^T\boldsymbol{x})$$

If $y = 1$



If $y = -1$



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Optimization Objective compared to Logistic Regression

Regularized Logistic Regression

$$\min_{\theta} \underbrace{\frac{1}{m}\sum_{i=1}^{m} loss_{LR}\left(\boldsymbol{\theta}^T \boldsymbol{x}^{(i)}, y^{(i)}\right)}_{A} + \underbrace{\frac{\lambda}{2m} || \boldsymbol{\theta} ||_2^2}_{B}$$

$A + \lambda B$

$CA + B$

$$C = \frac{1}{\lambda}$$

so a small C corresponds to giving B a larger weight

Support Vector Machine

$$\min_{\theta} C\sum_{i=1}^{m} loss_{SVM}\left(\boldsymbol{\theta}^T \boldsymbol{x}^{(i)}, y^{(i)}\right) + \frac{1}{2} || \boldsymbol{\theta} ||_2^2$$

Large C: Lower bias, high variance

Small C: Higher bias, low variance

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Optimization Objective

$$\min_{\boldsymbol{\theta}} \ \frac{1}{2} || \boldsymbol{\theta} ||_2^2$$

s.t.

$$y^{(i)} \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} - 1 \geq 0, \qquad i = 1, \ldots, m$$

This is a **quadratic programming** problem with linear inequality constraints

There are well known procedures for solving it, where data enters only in the form of dot products.

The optimal parameter vector can be written in the following form:

$$\boldsymbol{\theta}^* = \sum_{i=1}^{m} a_i^* y^{(i)} \boldsymbol{x}^{(i)}$$

where $a_i^* \geq 0$ are the Langrangian coefficients and they are non-zero only for the support vectors.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Classifying new data points

$$f_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(new)}\right) = \operatorname{sgn}\left(\boldsymbol{\theta}^T \boldsymbol{x}^{(new)}\right) = \begin{cases} +1 & if \quad \boldsymbol{\theta}^T \boldsymbol{x}^{(new)} \geq 0 \\ -1 & otherwise \end{cases}$$

Once the parameters are found by solving the required QP problem on the training set, the SVM can be used to classify new points using **only the support vectors**, their label and the associated coefficients.

$$\boldsymbol{\theta}^T \boldsymbol{x}^{(new)} = \sum_{i=1}^{m} a_i^* y^{(i)} \boldsymbol{x}^{(i)} \, \boldsymbol{x}^{(new)} = \sum_{i \in SV} a_i^* y^{(i)} \boldsymbol{x}^{(i)} \, \boldsymbol{x}^{(new)}$$

Data enters in the form of dot products

SVMs work well for linearly separable data.

What if the data are not linearly separable?

Co-financed by the European Union
Connecting Europe Facility

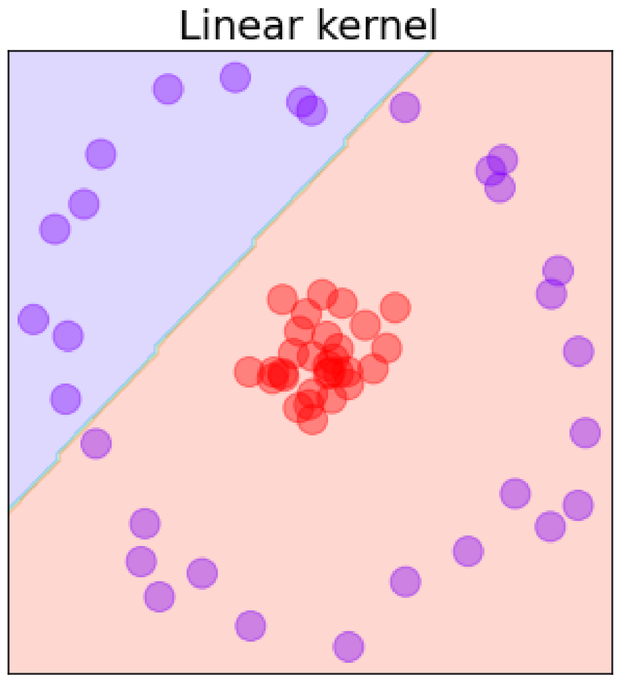
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

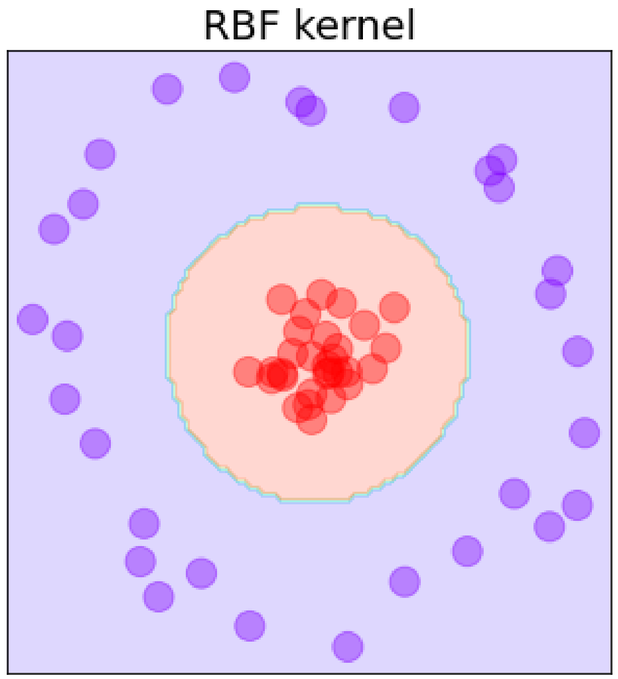
Non-linear SVMs

- The solution of the SVM has the nice property that the data enters only in the form of dot products.
- This allows us to make SVMs non-linear without complicating the algorithm, using kernels.
- The linear SVM depends on $\boldsymbol{x}^{(i)}\boldsymbol{x}^{(j)}$
- The nonlinear SVM replaces $\boldsymbol{x}^{(i)}\boldsymbol{x}^{(j)}$ with $k(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)})$

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Non-linear SVMs





Source

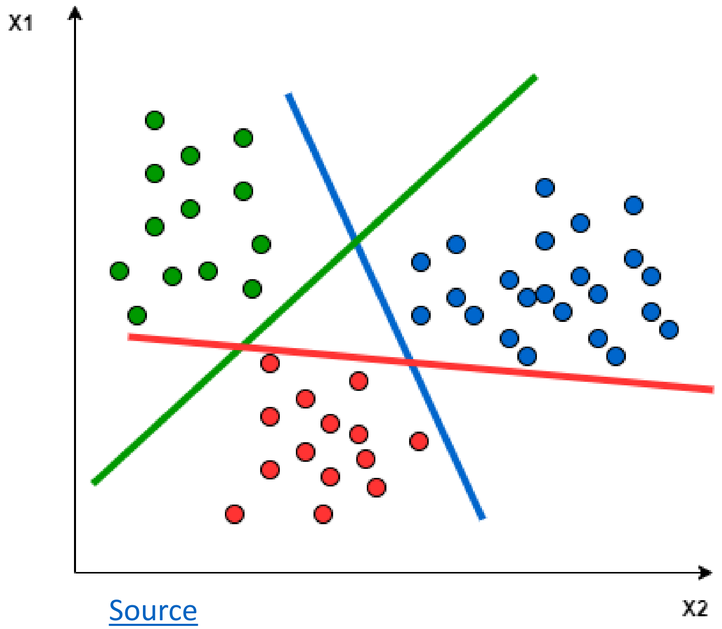
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

SVM with a polynomial kernel visualization

Youtube Video

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Multi-class classification



Source

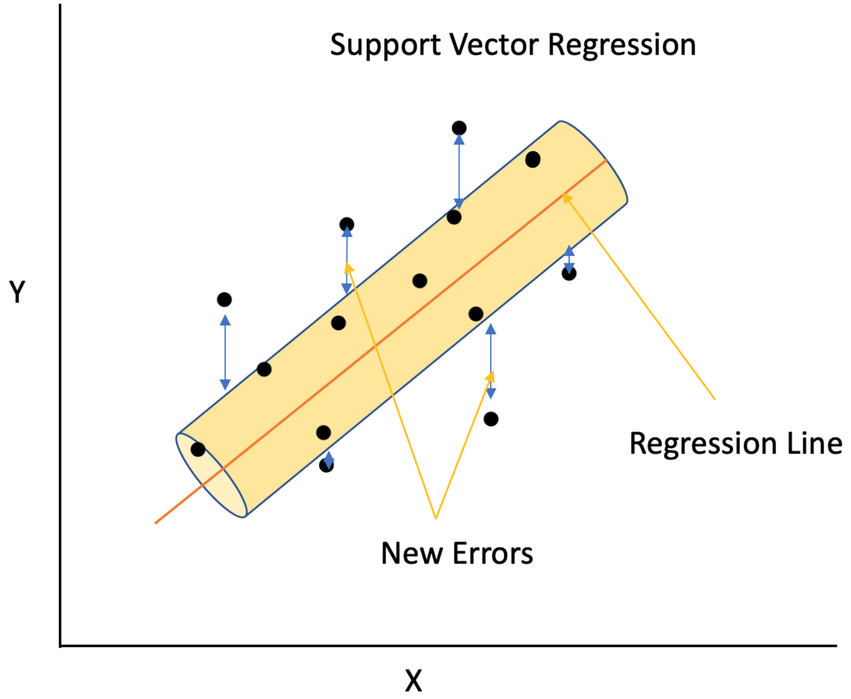
- Many SVM packages have built-in functionality for multi-class classification

- Use one-vs-all method
  - Train K SVMs (K=number of classes)
  - Get parameter vectors $\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(K)}$
  - Pick class with largest $\left(\boldsymbol{\theta}^{(i)}\right)^T \boldsymbol{x}$

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Support Vector Regression

- Extension of SVM to regression problems
- ε-insensitive tube: margin of error we are allowing our model to have
  - Do not care about error inside the tube
  - Ignore samples whose prediction is close to their target
- Slack Variables: points outside the tube
  - Only care about their errors

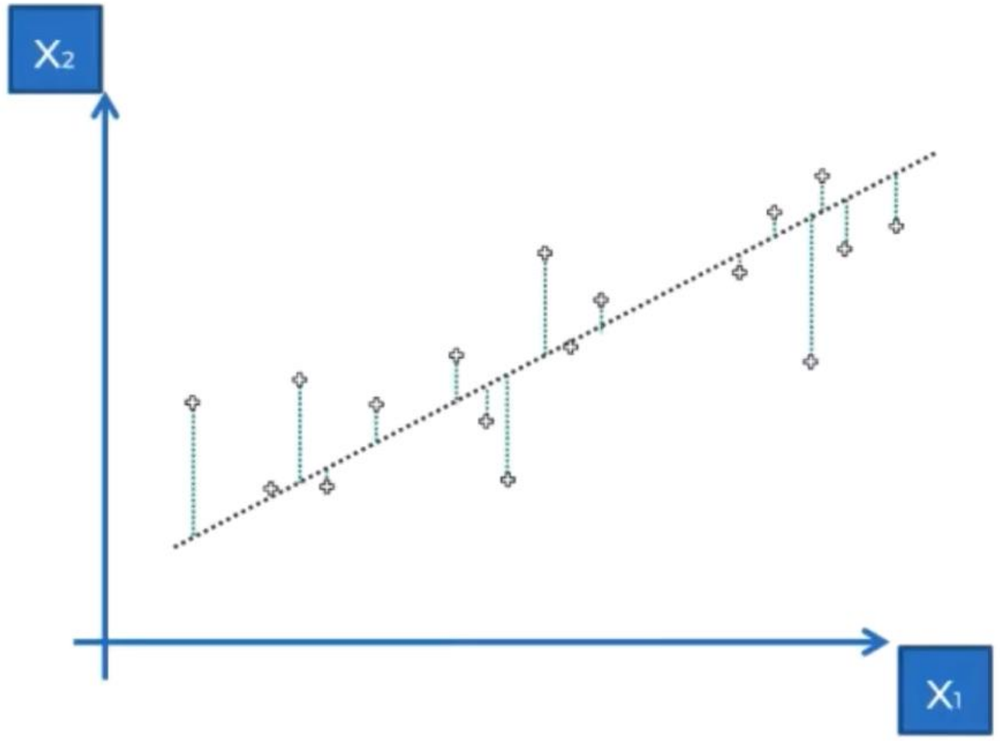


Source

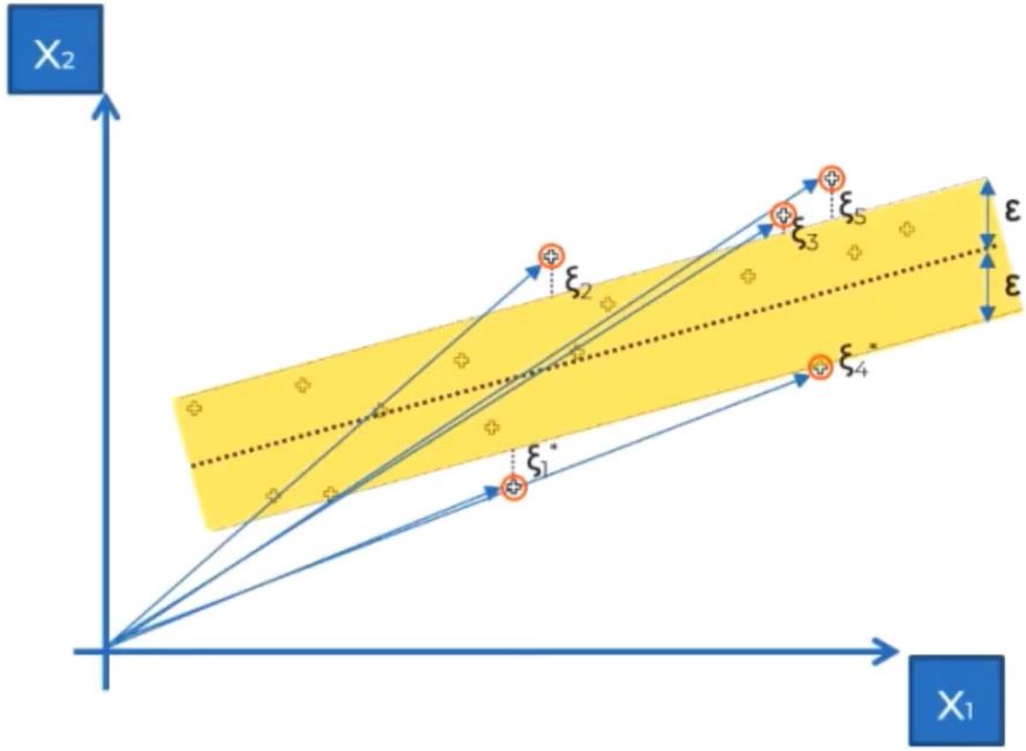
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Support Vector Regression

Simple Linear Regression



Support Vector Regression



Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Logistic regression vs SVMs

- If n is large relative to m, e.g., $n = 10{,}000, m = 10, \ldots 1000$
  - Use logistic regression, or SVM without a kernel (linear kernel)

- If n is small, m is intermediate, e.g., $n = 1 - 1000, m = 100 - 10{,}000$
  - Use SVM with Gaussian or Polynomial kernel

- If n is small, m is large, e.g., $n = 1 - 1000, m \geq 50{,}000$
  - Create/add more features, then use logistic regression or SVM without a kernel

Slide adapted from Andrew Ng's Machine Learning course, Coursera

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Summary

- A nonlinearly separable problem can be made linearly separable in higher dimensions
- The Kernel trick allows us to operate in a high-dimensional feature space without explicitly computing the coordinates of the data in that space
- The Support Vector Machine uses the concept of maximum margin in order to improve generalization error
- SVMs can use the kernel trick to learn non-linear decision boundaries
- SVMs can be used for multiclass classification problems and have been extended to regression problems

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 8: Kernel-based methods 2

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 8: Kernel-based methods 2

Learning Outcomes

You will understand:

1. About exact interpolation vs approximation
2. What Radial Basis Functions (RBF) are and how they can be used in RBF networks
3. How to train an RBF network for exact interpolation
4. How to train an RBF network for approximation
5. How to automatically optimize the parameters of the RBF using gradient descent
6. Normalized RBF networks
7. RBF networks for classification
8. The XOR problem
9. The basics about the Gaussian Process model

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Radial Basis Function Networks

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel methods for regression

- Placing a kernel at each training point allows to do **exact interpolation**
  - the model gives **exactly** the correct outputs for all training data

- What if we have a **huge** training set?
- What if the data points are **noisy**?

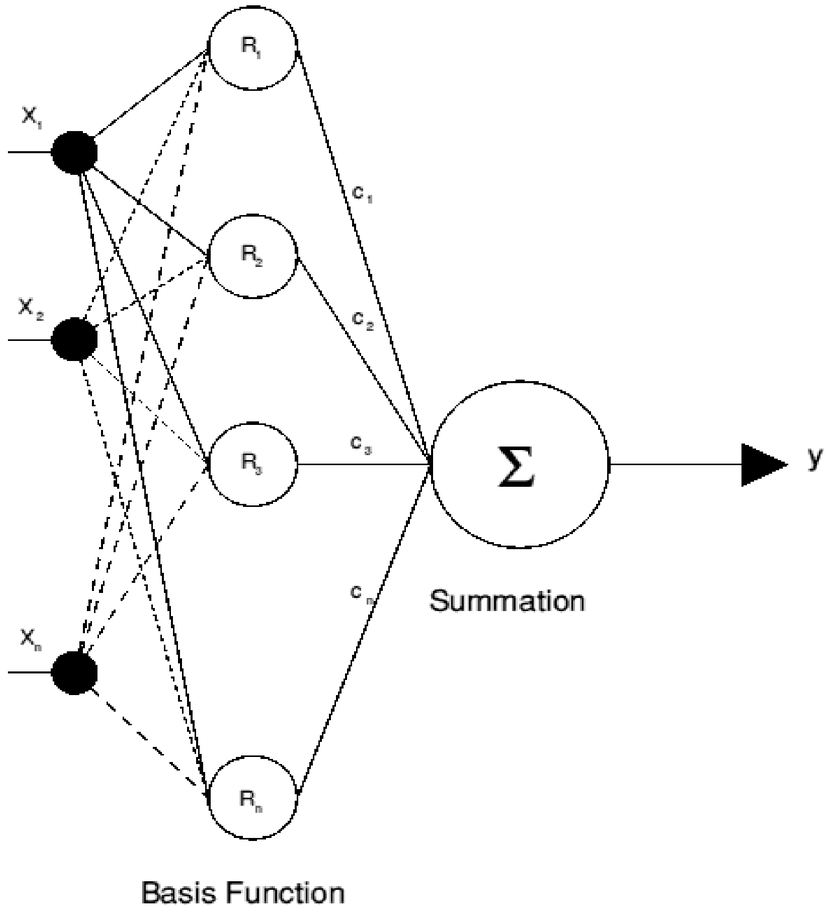
- **Approximation**:
  - we have less kernels than training data points
  - the model relates well the training vectors to the desired outputs, but does not produce the exactly correct outputs

- **Radial Basis Function (RBF) Networks**: models for interpolation and approximation

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Radial Basis Function Networks

- Layered network architecture
- Input layer:
  - distributes the input vector to all hidden nodes
- Hidden layer:
  - each node is a kernel containing a **centre** and a **Basis Function**
  - performs a non-linear mapping from input space (n) typically into a higher dimensional space (k>n)
- Output layer:
  - multiplies output of each hidden layer by a coefficient and sums the resulting values



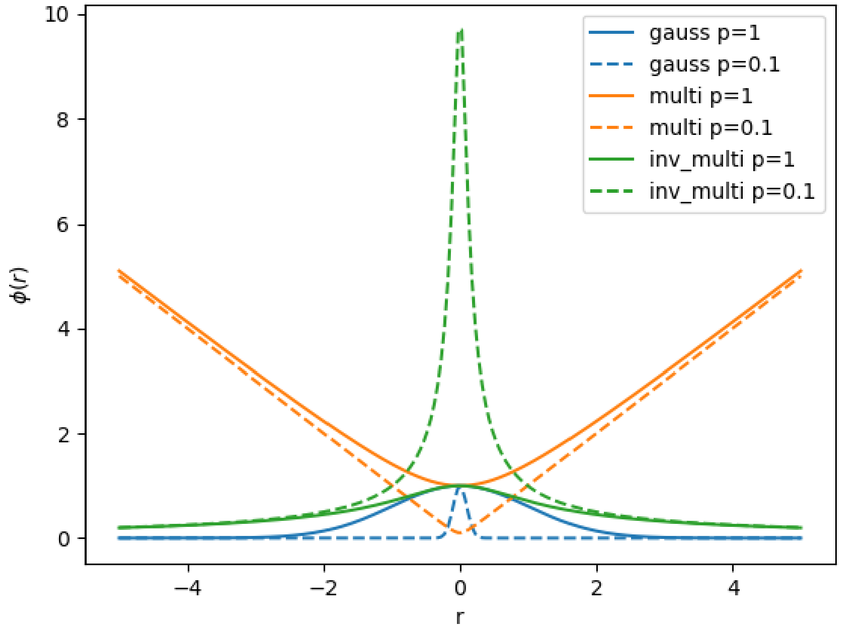
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Radial Basis Function (RBF)

Real-valued function φ whose value depends only on the **distance** between the input $\boldsymbol{x}$ and some fixed point $\boldsymbol{\mu}$, so that $\varphi(\boldsymbol{x}) := \varphi(r)$, where $r = \|\boldsymbol{x} - \boldsymbol{\mu}\|$, and distance is usually Euclidean distance.

Examples

- Gaussian: $\varphi(r) = \exp\left(-{r^2}/{p^2}\right)$
- Multiquadric: $\varphi(r) = \sqrt{r^2 + p^2}$
- Inverse multiquadric: $\varphi(r) = {1}/{\sqrt{r^2+p^2}}$



source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

RBF Network Main Idea

Given $\boldsymbol{x}$ the estimated output $\hat{y}$ is the linear combination of coefficients and features.

$$\hat{y} = \sum_{i=1}^{k} c_i \, \varphi_i(\boldsymbol{x})$$

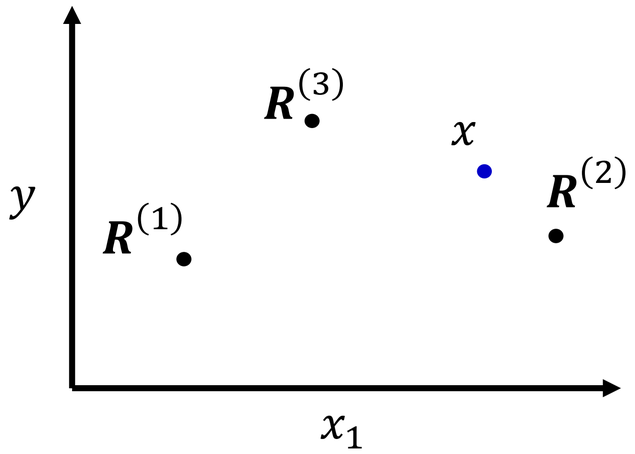
Features are based on **proximity** to centre $\boldsymbol{R}^{(i)}$:

$$\varphi_i(\boldsymbol{x}) = k\left(\boldsymbol{x}, \boldsymbol{R}^{(i)}\right)$$

Compare with simple linear regression:

$$\hat{y} = \sum_{i=1}^{k} c_i x_i \quad where\ k = n$$

$$\varphi_1(\boldsymbol{x}) = k\left(\boldsymbol{x}, \boldsymbol{R}^{(1)}\right)$$
$$\varphi_2(\boldsymbol{x}) = k\left(\boldsymbol{x}, \boldsymbol{R}^{(2)}\right)$$
$$\varphi_3(\boldsymbol{x}) = k\left(\boldsymbol{x}, \boldsymbol{R}^{(3)}\right)$$



Assuming a Gaussian kernel, which feature has the largest and which has the smallest value?

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gaussian RBF kernel and proximity

$$\varphi_1(\boldsymbol{x}) = k\left(\boldsymbol{x}, \boldsymbol{R}^{(1)}\right) = \exp\left(-\frac{\left\|\boldsymbol{x} - \boldsymbol{R}^{(1)}\right\|^2}{2\sigma^2}\right) = \exp\left(-\frac{\sum_{j=1}^{n}\left(x_j - R_j^{(1)}\right)^2}{2\sigma^2}\right)$$

$\boldsymbol{R}^{(1)}$ is the n-dimensional centre

If $\boldsymbol{x} \approx \boldsymbol{R}^{(1)}$:

$$\varphi_1(\boldsymbol{x}) \approx \exp\left(-\frac{0^2}{2\sigma^2}\right) \approx 1$$

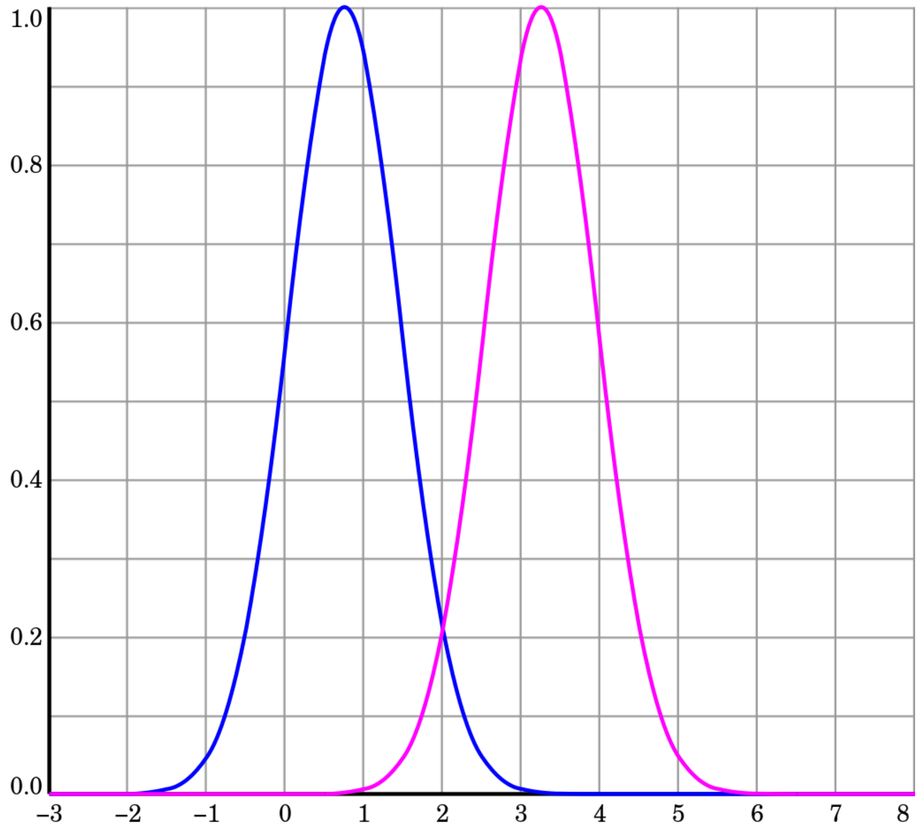
If $\boldsymbol{x}$ is far from $\boldsymbol{R}^{(1)}$:

$$\varphi_1(\boldsymbol{x}) \approx \exp\left(-\frac{(large\ number)^2}{2\sigma^2}\right) \approx 0$$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

RBF kernel: centre coordinates



Source

Keeping σ fixed

$R^{(1)} = 0.75$
$R^{(2)} = 3.25$

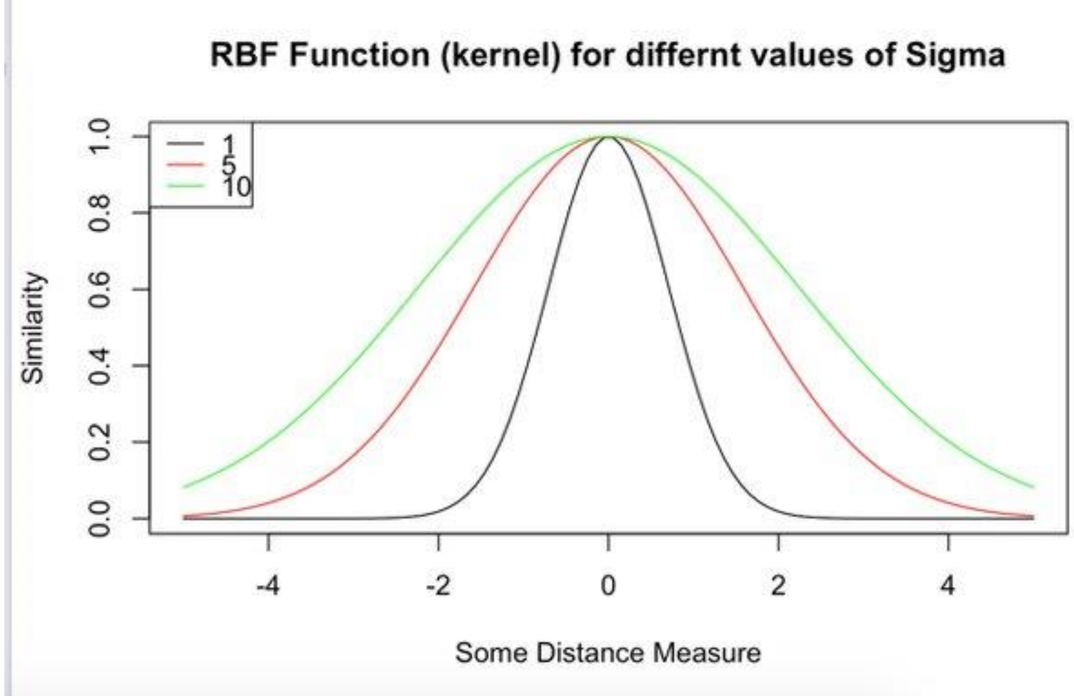
When designing an RBF network it is crucial to cover the whole data space with kernels.

The influence of each kernel diminishes with distance.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

RBF kernel: effect of σ



source

σ = width of Gaussian

Large σ: Features vary more smoothly

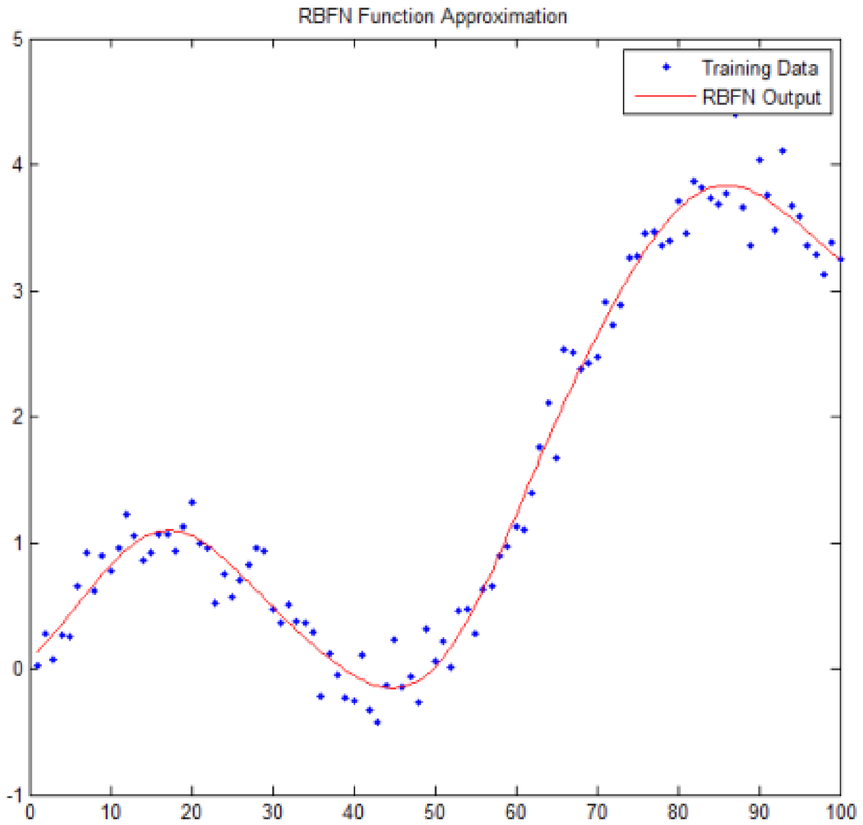
Higher bias, lower variance

Small σ: Features vary less smoothly

Lower bias, higher variance

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

What does a trained RBF network learn



source

Example

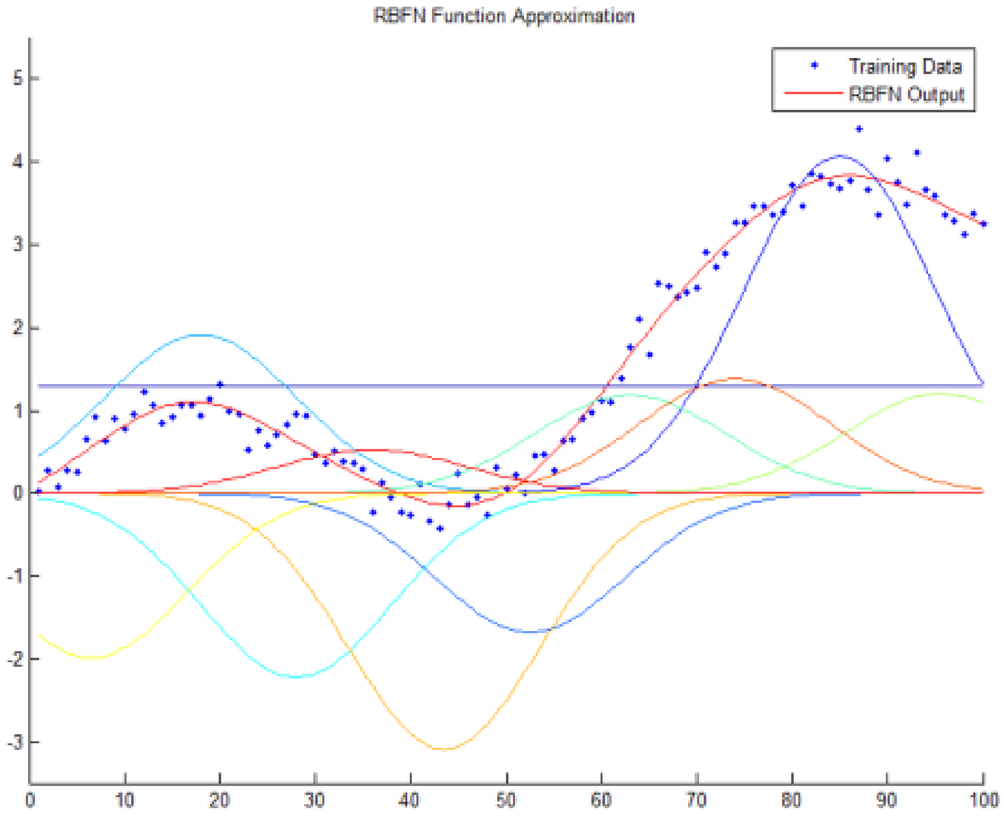
1D input

Dataset of 100 points

RBF with 10 basis functions

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

How does a trained RBF network learn



source

The output of the RBF network is the weighted sum of 10 Gaussians

Each Gaussian has a different center

The height is determined by the learned coefficient (weight) which can be negative

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Training an RBF network for interpolation

1. Create m centres, where each centre is placed at a training point: $\boldsymbol{R}^{(i)} = \boldsymbol{x}^{(i)}$
2. Set σ to some appropriate value
3. Compute parameters **c**

For each training pair $(\boldsymbol{x}^{(i)}, y^{(i)})$ we have one equation:

$$y^{(1)} = c_1 k\left(\boldsymbol{x}^{(1)}, \boldsymbol{R}^{(1)}\right) + c_2 k\left(\boldsymbol{x}^{(1)}, \boldsymbol{R}^{(2)}\right) + \cdots + c_m k\left(\boldsymbol{x}^{(1)}, \boldsymbol{R}^{(m)}\right)$$
$$y^{(2)} = c_1 k\left(\boldsymbol{x}^{(2)}, \boldsymbol{R}^{(1)}\right) + c_2 k\left(\boldsymbol{x}^{(2)}, \boldsymbol{R}^{(2)}\right) + \cdots + c_m k\left(\boldsymbol{x}^{(2)}, \boldsymbol{R}^{(m)}\right)$$

…

$$y^{(m)} = c_1 k\left(\boldsymbol{x}^{(m)}, \boldsymbol{R}^{(1)}\right) + c_2 k\left(\boldsymbol{x}^{(m)}, \boldsymbol{R}^{(2)}\right) + \cdots + c_m k\left(\boldsymbol{x}^{(m)}, \boldsymbol{R}^{(m)}\right)$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Training an RBF network for interpolation

Represent equations in matrix form:

$$\underbrace{\begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}}_{\boldsymbol{y}} = \underbrace{\begin{bmatrix} k(\boldsymbol{x}^{(1)}, \boldsymbol{R}^{(1)}) & \dots & k(\boldsymbol{x}^{(1)}, \boldsymbol{R}^{(m)}) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}^{(m)}, \boldsymbol{R}^{(1)}) & \dots & k(\boldsymbol{x}^{(m)}, \boldsymbol{R}^{(m)}) \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix}}_{\boldsymbol{c}}$$

Solve for **c**:

$$\boldsymbol{c} = (A^T A)^{-1} A^T \boldsymbol{y}$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Training an RBF network for approximation using fixed centres

1. Choose k<m appropriate centres:

- equidistantly spaced
- randomly sampled (e.g., uniformly over space or from set of examples)
- use clustering algorithms (e.g., k-means clustering)

2. Set σ to some appropriate value

- manually
- determined from cluster radius (could have different σ for each centre)

3. Compute parameters **c**

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Training an RBF network for approximation using fixed centres

In matrix form:

$$\underbrace{\begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}}_{\boldsymbol{y}} = \underbrace{\begin{bmatrix} k(\boldsymbol{x}^{(1)}, \boldsymbol{R}^{(1)}) & \dots & k(\boldsymbol{x}^{(1)}, \boldsymbol{R}^{(k)}) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}^{(m)}, \boldsymbol{R}^{(1)}) & \dots & k(\boldsymbol{x}^{(m)}, \boldsymbol{R}^{(k)}) \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \end{bmatrix}}_{\boldsymbol{c}}$$

Solve for **c**:

$$\boldsymbol{c} = (A^T A)^{-1} A^T \boldsymbol{y}$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Width of the Gaussian (σ)

Interpolation task:

Small σ: bumpy interpolations

Larger σ: extend the influence of each sample

Very large σ: too global influence of each sample

Approximation task:

To get a good generalisation, the width should be larger than the average distance between adjacent vectors, but smaller than the distance across all input space.

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Training an RBF network for approximation using adjustable centres

- We can train coefficients c, centres R, and variances σ
  - This means that the centre coordinates and the radii of the basis functions could be better adapted to the data

- We can use (Stochastic) Gradient Descent to achieve this
  - Need: loss function, derivatives of the loss w.r.t. c, R and σ

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Training an RBF network for approximation using adjustable centres

1. Choose the number (k) and initial coordinates of the centres (R).
2. Choose the initial value of the spread parameter (σ) for each centre (R).
3. Initialize the weights/coefficients (c) to small random values [-1,1].
4. Repeat until stopping criteria are satisfied (e.g., maximum number of epochs)
   1. Give an input $(\boldsymbol{x}^{(p)}, y^{(p)})$ to the network and calculate the network's output

$$\widehat{y^{(p)}} = c_0 + \sum_{h=1}^{k} c_h \varphi(\boldsymbol{x}^{(p)}, \boldsymbol{R}^{(h)}, \sigma_h) \text{ where } \varphi\left(\boldsymbol{x}^{(p)}, \boldsymbol{R}^{(h)}, \sigma_h\right) = \exp\left(-\frac{\left\|\boldsymbol{x}^{(p)} - \boldsymbol{R}^{(h)}\right\|^2}{2\sigma_h^2}\right)$$

   2. Update network's parameters (c, **R**, σ) (next slide)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Training an RBF network for approximation using adjustable centres

Update c:

$$c_0 := c_0 - \eta_c \varepsilon^{(p)}$$
$$c_h := c_h - \eta_c \varepsilon^{(p)} \varphi(\boldsymbol{x}^{(p)}, \boldsymbol{R}^{(h)}, \sigma_h) \rightarrow \frac{\partial L}{\partial c}$$

$$\varepsilon^{(p)} = \widehat{y^{(p)}} - y^{(p)}$$

$$\mathrm{L}(\boldsymbol{c}, \boldsymbol{R}, \boldsymbol{\sigma}) = \frac{1}{2}[\varepsilon^{(p)}]^2$$

Update $\boldsymbol{R}$:

$$\boldsymbol{R}^{(h)} := \boldsymbol{R}^{(h)} - \eta_R \varepsilon^{(p)} c_h \varphi\left(\boldsymbol{x}^{(p)}, \boldsymbol{R}^{(h)}, \sigma_h\right)(\boldsymbol{x}^{(p)} - \boldsymbol{R}^{(h)})/\sigma_h^2 \rightarrow \frac{\partial L}{\partial R}$$

Learning Rates

$0 < \eta_c \leq 1$
$0 < \eta_{\mathrm{R}} \leq 1$
$0 < \eta_\sigma \leq 1$

Update σ:

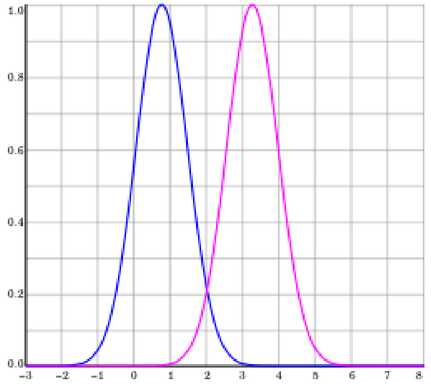
$$\sigma_h := \sigma_h - \eta_\sigma \varepsilon^{(p)} c_h \varphi\left(\boldsymbol{x}^{(p)}, \boldsymbol{R}^{(h)}, \sigma_h\right) \left\|\boldsymbol{x}^{(p)} - \boldsymbol{R}^{(h)}\right\|^2 / \sigma_h^3 \rightarrow \frac{\partial L}{\partial \sigma}$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Normalized RBF network

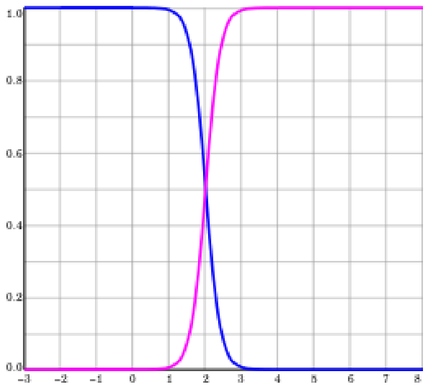
Unnormalized:

$$\hat{y} = \sum_{i=1}^{k} c_i \, \varphi_i(\boldsymbol{x})$$



Normalized:

$$\hat{y} = \frac{\sum_{i=1}^{k} c_i \, \varphi_i(\boldsymbol{x})}{\sum_{i=1}^{k} \varphi_i(\boldsymbol{x})}$$



Unnormalized:

- more localized

Normalized:

- may exhibit better generalization

Both are universal function approximators: given enough hidden units, they can approximate any continuous function.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

RBF networks for classification

We can use RBF networks for classification by feeding the output into the logistic/sigmoid function:

$\hat{y} = \sigma\left(\sum_{i=1}^{k} c_i \, \varphi_i(\boldsymbol{x})\right)$, where $\sigma(z) = \frac{1}{(1+e^{-z})}$

This is the same as logistic regression, where the features are computed using the Gaussian RBF kernel.

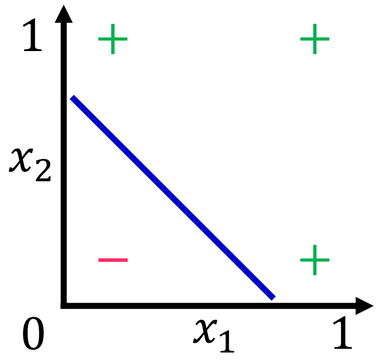
➢Creates nonlinear decision boundaries

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

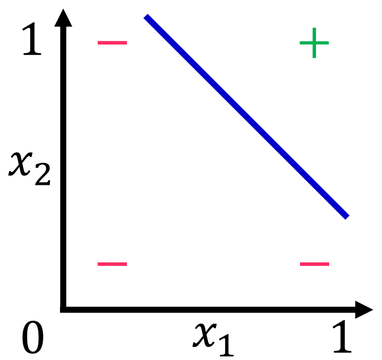
XOR problem

OR



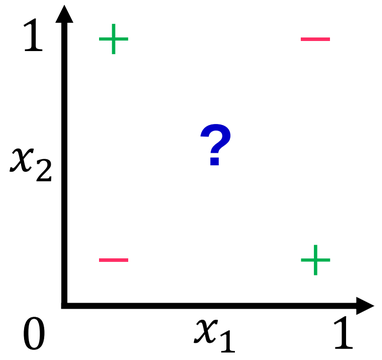
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

AND



x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR



x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

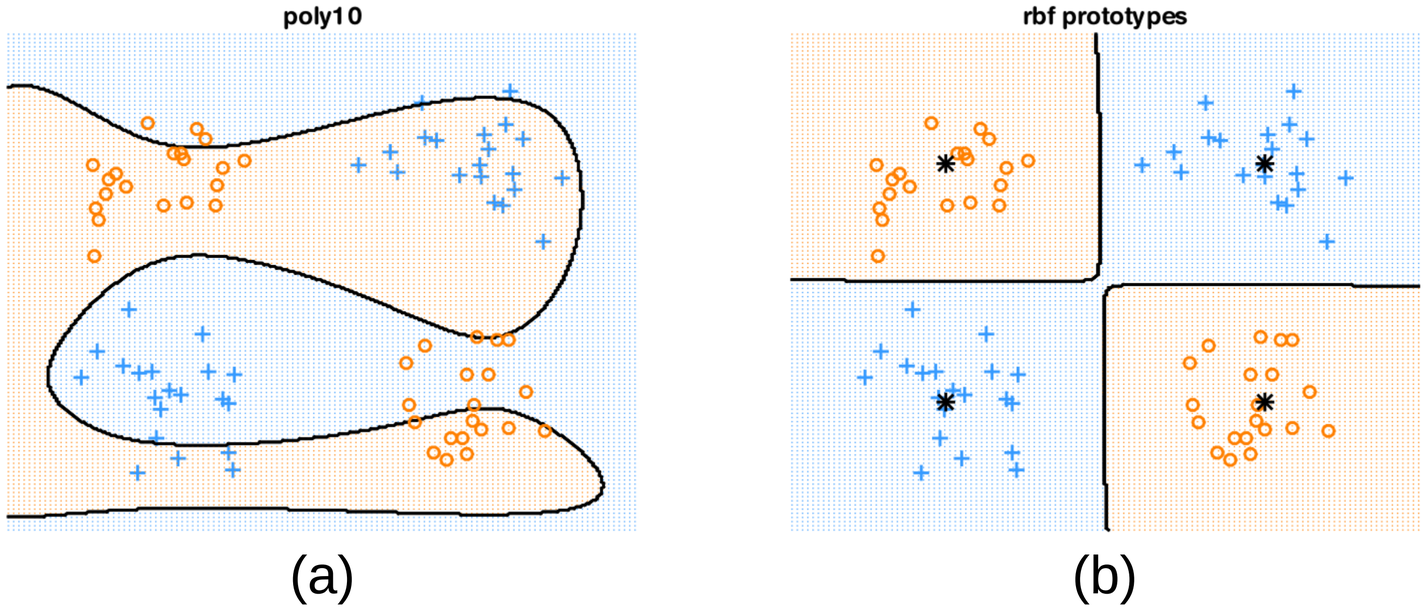
OR, AND: linearly separable

XOR: **not** linearly separable

Motivates the need for nonlinear transformation of inputs

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Noisy XOR problem



Fitting a linear logistic regression classifier using:

a) degree 10 polynomial expansion
   - ➢ cannot separate the classes

b) RBF kernel with centres specified by the 4 black points
   - ➢ easily solves the problem

Source: Murphy, K. (2022). Probabilistic Machine Learning - An Introduction. MIT Press

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Gaussian Processes

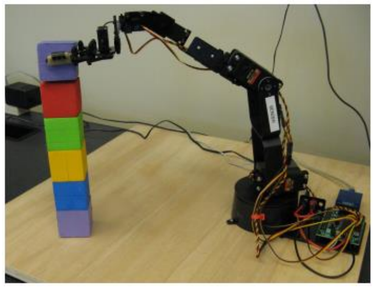
Co-financed by the European Union
Connecting Europe Facility

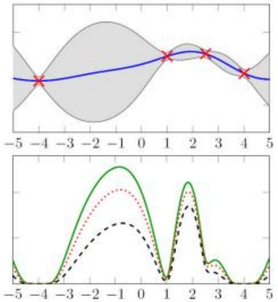
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

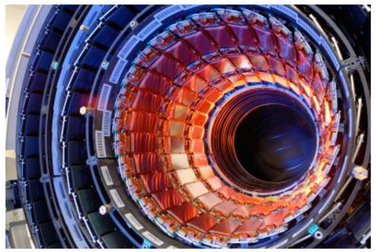
Gaussian Processes

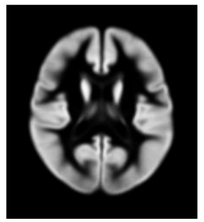
- **Gaussian process**: probabilistic method that gives a confidence for the predicted function
- Used mostly in regression tasks (however, it has also been used in classification and clustering tasks)

- **Some Application areas:**
  - Robotics
  - Experimental Design
  - Geostatistics
  - Time-series modeling and forecasting
  - High-energy physics
  - Medical applications





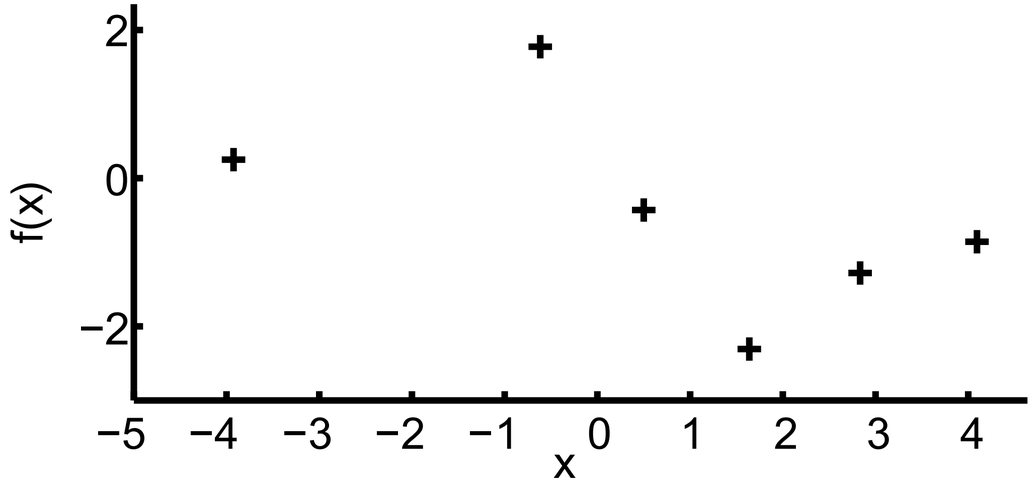


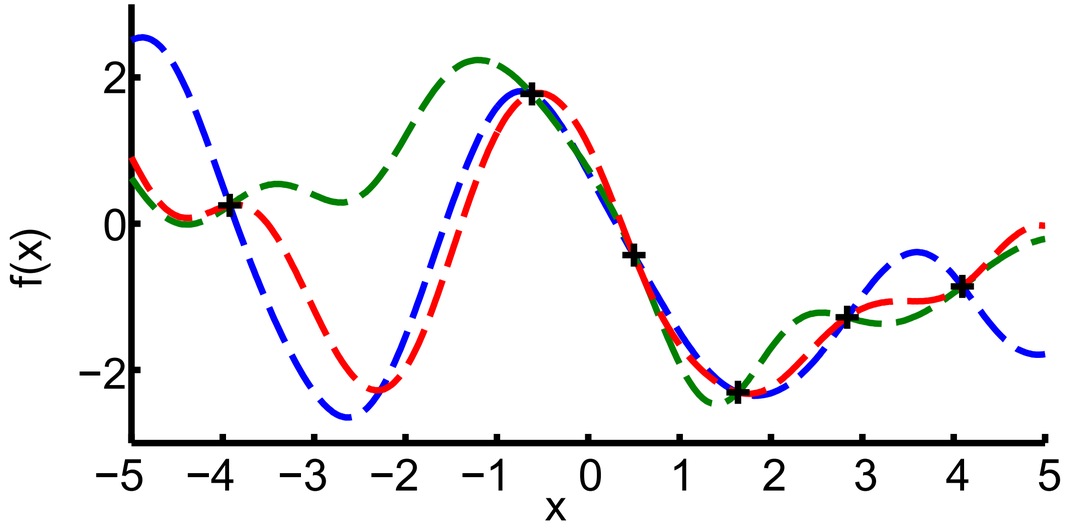


This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gaussian Processes Main Idea

- For a given set of training points, there are potentially infinitely many functions that fit the data





- How can we deal with this?

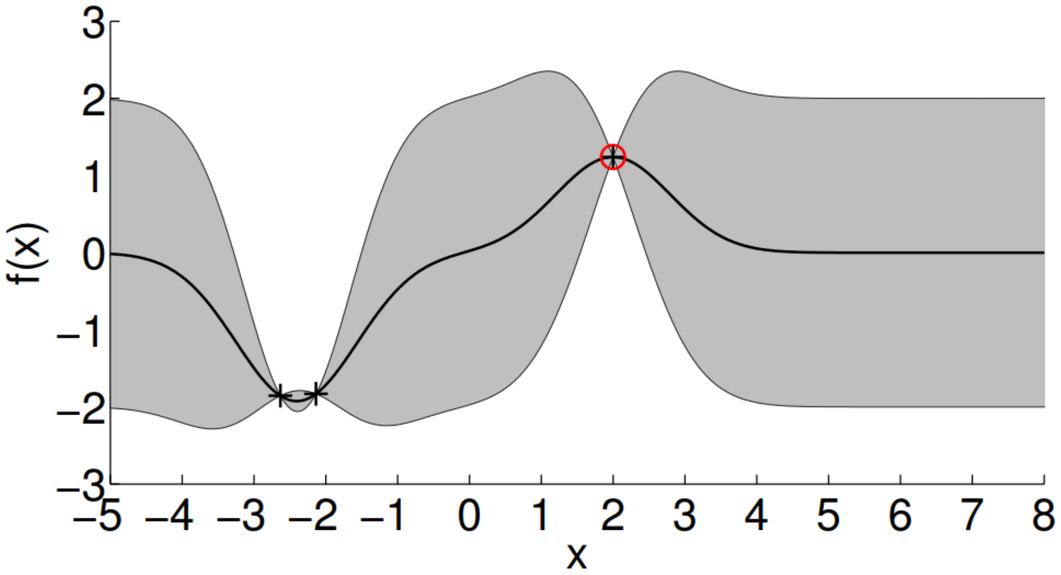
Figures adapted from: Deisenroth, M. P. & Rasmussen, C. E. (2011) PILCO: A Model-Based and Data-Efficient Approach to Policy Search

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gaussian Processes Main Idea

- GPs assign a probability to each of these functions
- The mean of this probability distribution (black line) represents the most probable characterization of the data
- Probabilistic approach allows us to incorporate the confidence of the prediction into the regression result (shown as shaded region)

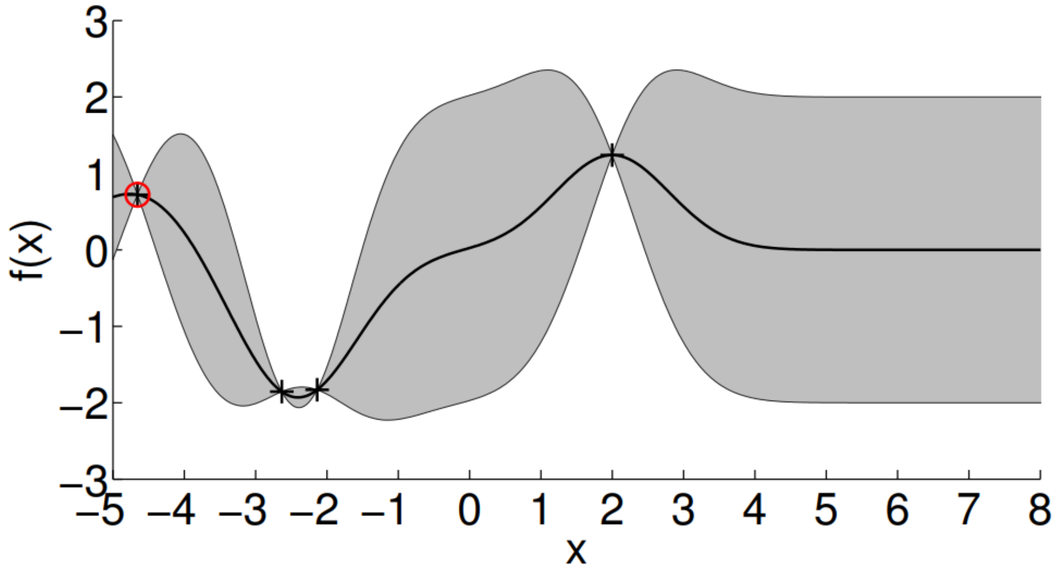


Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gaussian Processes Main Idea

- As we add more data points, we adapt the distribution

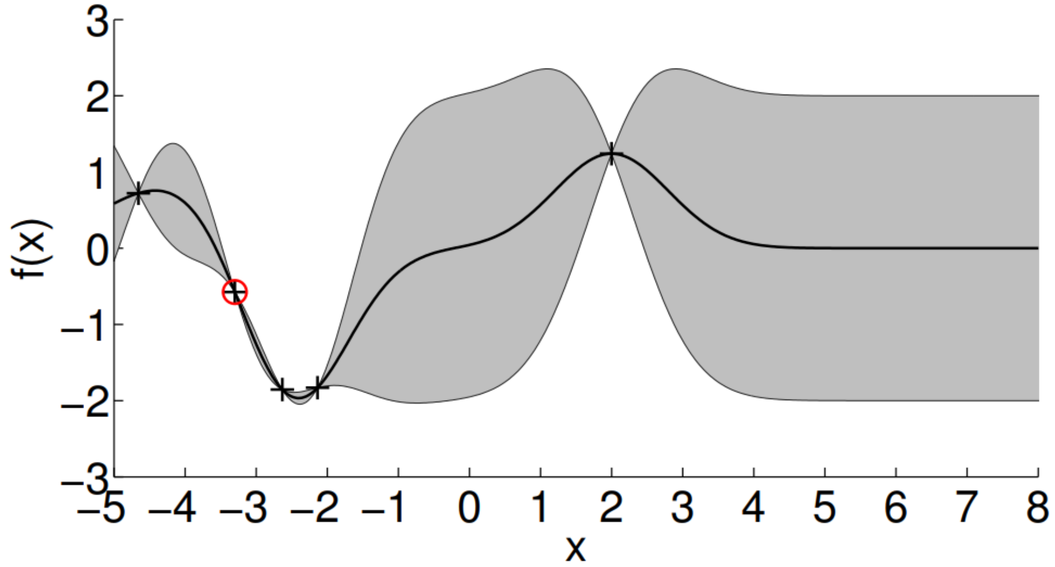


Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gaussian Processes Main Idea

- As we add more data points, we adapt the distribution

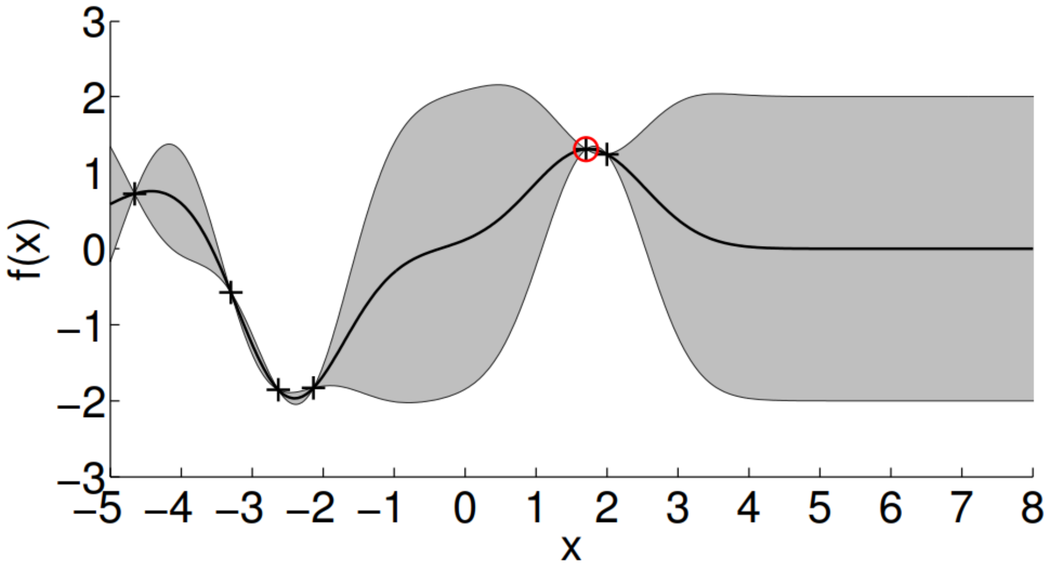


Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gaussian Processes Main Idea

- As we add more data points, we adapt the distribution

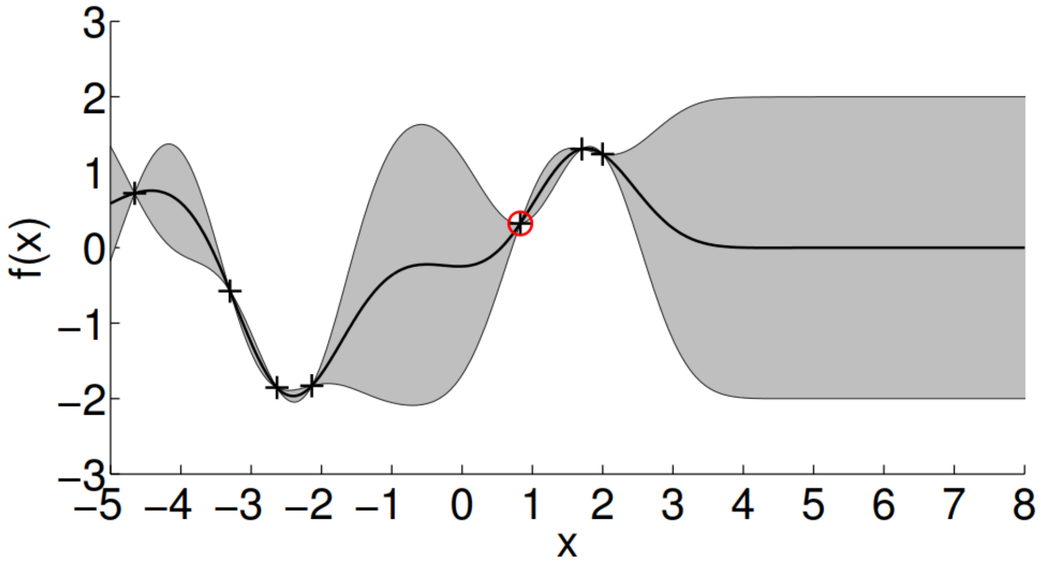


Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gaussian Processes Main Idea

- As we add more data points, we adapt the distribution

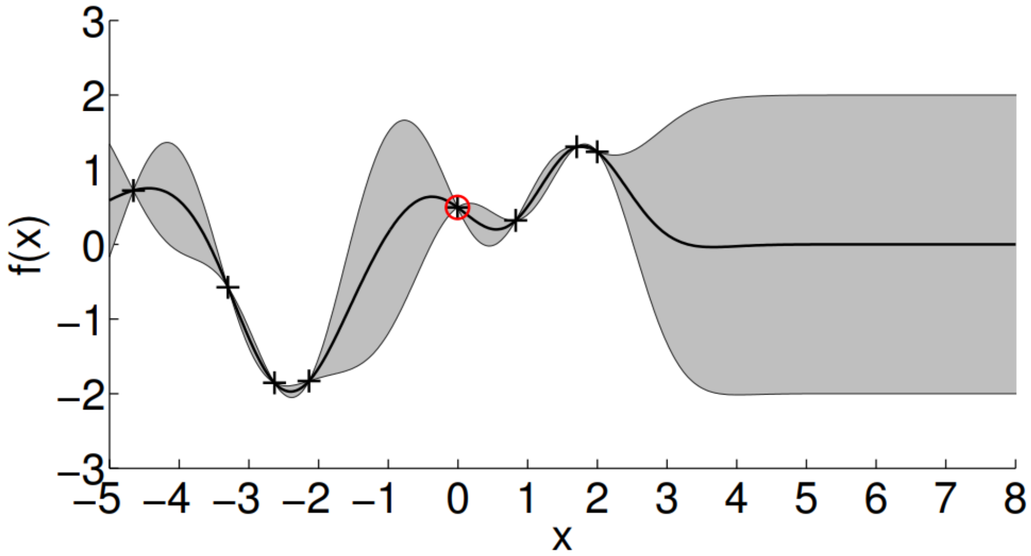


Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gaussian Processes Main Idea

- As we add more data points, we adapt the distribution

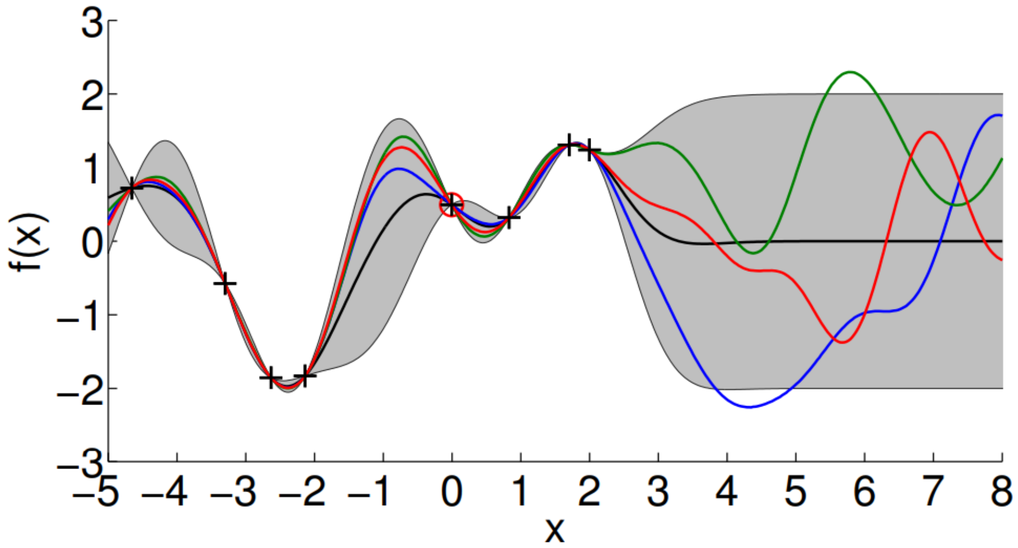


Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gaussian Processes Main Idea

- Since a GP is a probability distribution over functions, we can sample functions from it.



Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gaussian Process Definition

- We place a distribution $p(f)$ on functions f
- Informally, a function can be considered an infinitely long vector of function values $f = [f_1, f_2, f_3, \dots]$
- A GP is a generalization of a multivariate Gaussian distribution to infinitely many variables.

Definition (Rasmussen & Williams, 2006): A GP is a collection of random variables $f_1, f_2, \dots$, any finite number of which is Gaussian distributed.

- A Gaussian distribution is specified by a mean vector $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$
- A GP is specified by a **mean function** $m(.)$ and a **covariance function (kernel)** $k(.,.)$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gaussian Process Regression as Bayesian Inference

Objective: For a set of observations $y^{(i)} = f\left(x^{(i)}\right) + \varepsilon,\ \varepsilon \sim N(0, \sigma^2)$ find a (posterior) distribution over functions $p(f(.)|\boldsymbol{X}, \boldsymbol{y})$ that explains the data. Here: $\boldsymbol{X}$ training inputs, $\boldsymbol{y}$ training targets.

Training data: $\boldsymbol{X}$, $\boldsymbol{y}$. Bayes' theorem yields

$$p(f(.)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(.), \boldsymbol{X})p(f(.))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Prior: $p\left(f(.)\right) = GP(m, k)$ → Specify mean function m and kernel k

Likelihood (noise model): $p(\boldsymbol{y}|f(.), \boldsymbol{X}) = N(f(\boldsymbol{X}), \sigma^2 \boldsymbol{I})$

Marginal likelihood (evidence): $p(\boldsymbol{y}|\boldsymbol{X}) = \int p(\boldsymbol{y}|f(.), \boldsymbol{X})p(f(.)|\boldsymbol{X})\, df$

Posterior: $p(f(.)|\boldsymbol{X}, \boldsymbol{y}) = GP(m_{post}, k_{post})$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

GP prior

What characterizes the function we want to model?

- Mean function
  - Allows us to bias the model (can make sense in application-specific settings)
  - Can be a parametrized function. Example: $m_{\boldsymbol{\theta}}(x) = \boldsymbol{\theta}^T \boldsymbol{\varphi}(\boldsymbol{x})$
  - Can incorporate problem-specific prior knowledge (e.g., in robotics, natural sciences)
  - Can simplify the problem
  - Often: “Agnostic” mean function in the absence of data or prior knowledge: $m(.) = 0$ everywhere
- Covariance function (kernel)
  - Encodes high-level structural assumptions (e.g., smoothness, periodicity) of the function we want to model
  - Gaussian kernel: $k_{Gauss}\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right) = \sigma_f^2 \exp\left(-\left(\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)}\right)^T \left(\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)}\right)/\lambda^2\right)$
    - σ_f : Amplitude (vertical magnitude) of the function
    - λ : Length scale (standard deviation in a Gaussian distribution)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gaussian Processes Interactive diagrams

- https://distill.pub/2019/visual-exploration-gaussian-processes/
- http://chifeng.scripts.mit.edu/stuff/gp-demo/
- http://www.infinitecuriosity.org/vizgp/
- https://edward-rees.com/gp/
- http://smlbook.org/GP/

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Application of Gaussian Processes: Robot Damage Recovery

YouTube Video

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Next Lecture

- Neural Networks

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 9: Neural Networks 1: Modelling

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Revision

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel methods

- Classification problems often require nonlinear decision boundaries
  - These can be constructed using feature engineering, e.g., adding polynomial features
  - As we add features, we increase the dimensionality of the input which often makes the problem linearly separable, i.e., easier to solve in higher dimensions
  - However, this approach can exponentially increase the number of parameters to be learned, and has the disadvantage of needing to compute features explicitly
- We can alleviate these problems using the kernel trick, i.e., to use a kernel function that takes as input pairs of points in the original space, but computes their similarity in a higher dimensional feature space, without explicitly computing the feature transformation in the higher dimensional space
- Kernel examples: polynomial, Gaussian
- We can make valid kernels by combining valid kernels through addition, multiplication and scaling with a positive constant

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel methods

- Kernel methods are instance-based learners which compute the similarity of an input with stored training points and multiply this by some learned weight which is specific for that particular training point
- Support Vector Machines are binary classification models that learn the separating hyperplane with the maximum margin, and use kernels to deal with nonlinearly separable problems
- Maximum margin decision boundary:
  - minimizes the generalization error, as it ensures that points are classified far from the separating hyperplane
  - computed by finding the support vectors, i.e., the training points closest to the separating hyperplane
- SVMs optimize the hinge loss which penalizes incorrectly classified points by taking into account the margin (instead of only the decision boundary)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel methods

- SVMs' optimization objective can be solved using quadratic programming where data enters in the form of dot products and which returns certain coefficients that are greater than zero only for the support vectors
- This means that once the training is complete, SVMs only need to keep the support vectors to make a prediction, which makes the prediction fast
- SVMs can learn nonlinear decision boundaries by replacing the dot product of the original points in the optimization objective, with an appropriate kernel function
- We can do multi-class classification with SVMs using the one-vs-rest approach
- Support Vector Regression extends SVMs to regression problems
  - fits a line to the data with a margin (tube) around it
  - ignores points inside the tube because their combined error is small

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel methods

- Radial-basis function (RBF) networks are regression models for exact interpolation and approximation
- They compute a nonlinear transformation of the data using kernels and linearly combine them using weights
- The centre of the kernel can be at a training point (exact interpolation), however, it can be elsewhere
- Commonly used kernel: Gaussian
  - Value of 0 if distance between queried point and kernel centre is large
  - Value of 1 if distance between queried point and kernel centre is 0 (same point)
- If σ (Gaussian width) is large: smoother fit, thus, higher bias, lower variance
- We can use the normal equations to compute the RBF network weights for interpolation and approximation (fixed centres)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel methods

- We can adapt the weights, centre coordinates and Gaussian widths of an RBF network using gradient descent: results in better fit
- Unnormalized RBFs: more localized, need more (or wider) to cover the input space
- Normalized RBF networks may exhibit better generalization with fewer nodes
- RBF networks can be used for classification by feeding the output of regression RBF network into a sigmoid function (similarly to logistic regression)
- Gaussian processes: nonparametric probabilistic regression models that output not only the prediction but also the confidence in the prediction
  - Probability distribution over functions: can sample functions from it
  - Good for low-data problems
  - Can be provided with prior knowledge about the function we want to model

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 9: Neural Networks 1: Modelling

Learning Outcomes

You will understand:

1. What an artificial neuron is
2. The perceptron model, its learning algorithm and limitations
3. How to construct nonlinear decision boundaries by combining perceptrons
4. Different activation functions
5. What a feedforward neural network (NN) model is and how to represent it
6. How to implement a NN for fast computation
7. Why NNs compute their own features

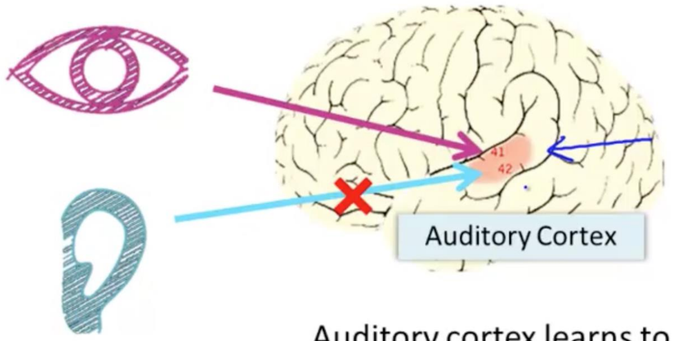
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Motivation: the human brain

- **Artificial Intelligence dream**: can we build truly intelligent machines?
- **Challenge**: mimic the human brain

- The “One Learning Algorithm” conjecture and plasticity
  - Brain rewiring experiments
  - Brainport: using tongue to “see”
  - Implanting sensors to the brain (e.g., a 3rd eye to a frog)
  - Brain learns how to use a new sensor



Auditory cortex learns to see

Images source: Andrew Ng’s Machine Learning course, Coursera

source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neural networks

- Algorithms that try to mimic the brain
- Networks of artificial neurons
  - Artificial neurons: crude approximation of biological neurons. They may be physical devices or mathematical constructs.
  - Artificial neural networks: They may be physical devices or simulated on conventional computers.
- A NN is just a parallel computational system consisting of many simple processing elements connected together in a specific way in order to perform a particular task.
- It acquires knowledge from its environment through a learning algorithm.
- Knowledge is stored as interneuron connection strengths, known as synaptic weights.

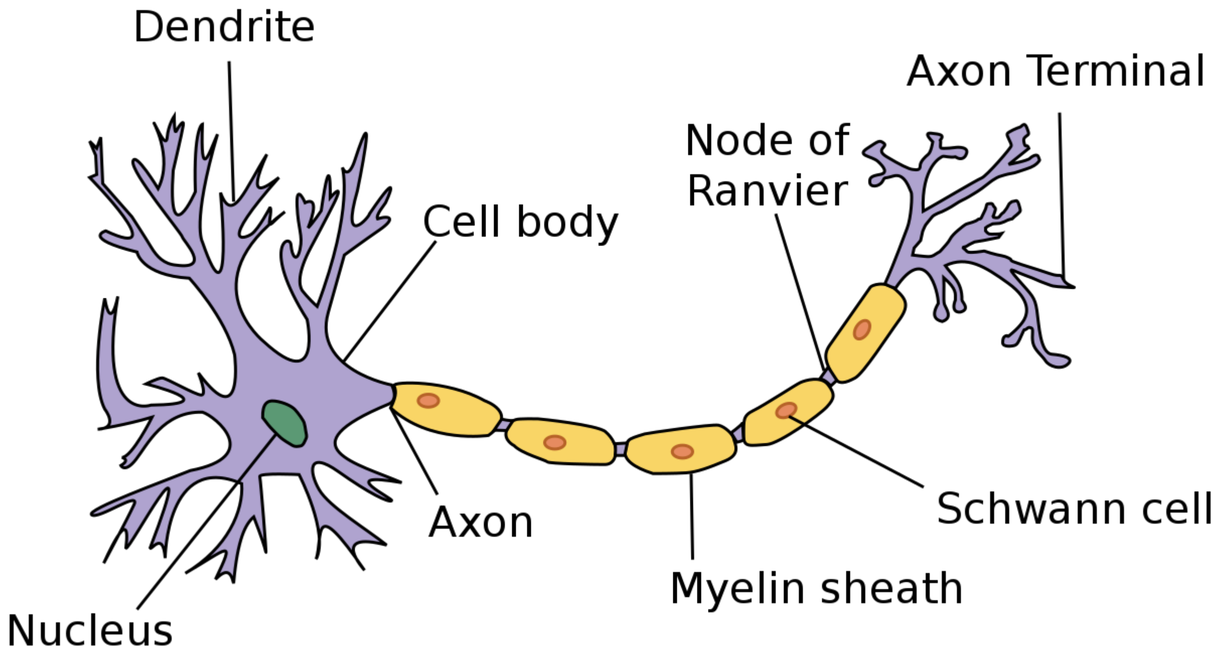
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neural networks

- Very widely used in 80s and early 90s; popularity diminished in late 90s
- Recent resurgence: state-of-the-art technique for many applications
- Gave birth to a subfield of ML known as **Deep Learning** (more in a later lecture)

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

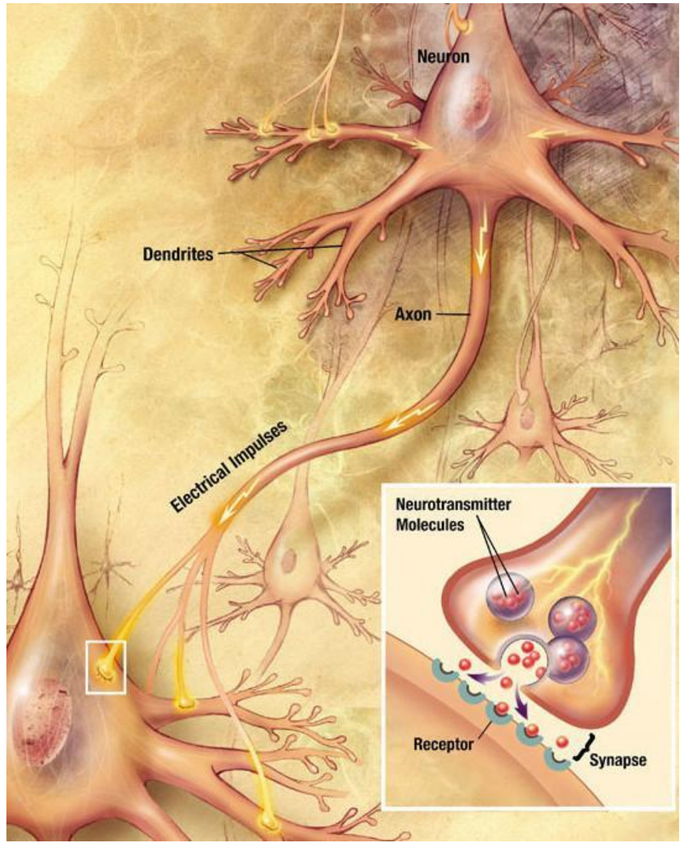
The Neuron – A high-level overview



source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neurons in the brain



source

100 billion neurons in the human brain connected to 10,000 others

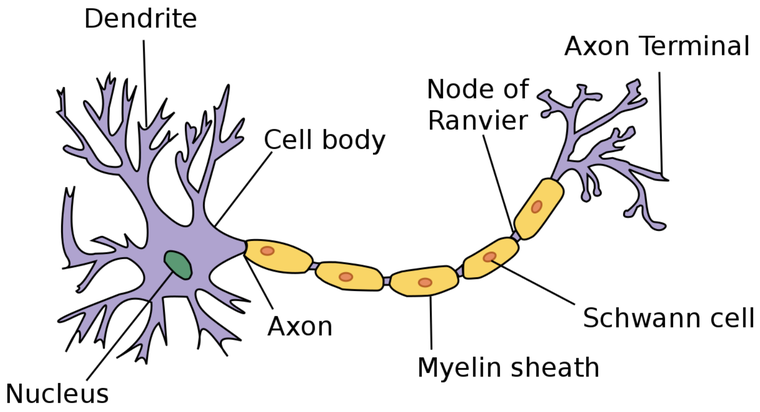
100-500 trillion synapses in the adult brain

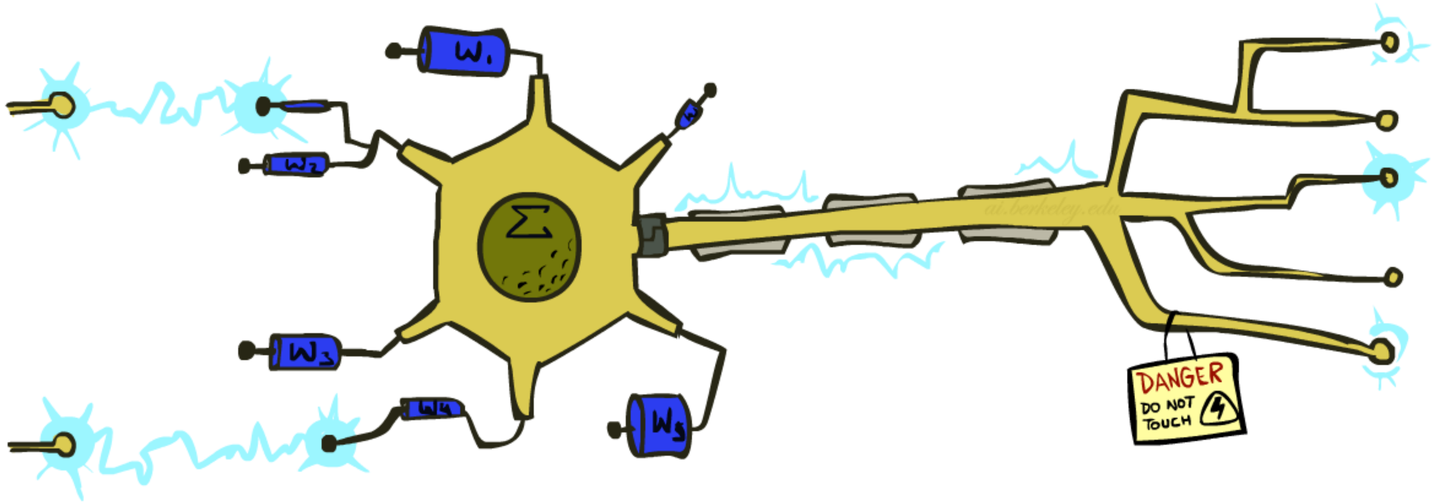
Power: 20W

Frontier Supercomputer: 21MW

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Artificial Neuron

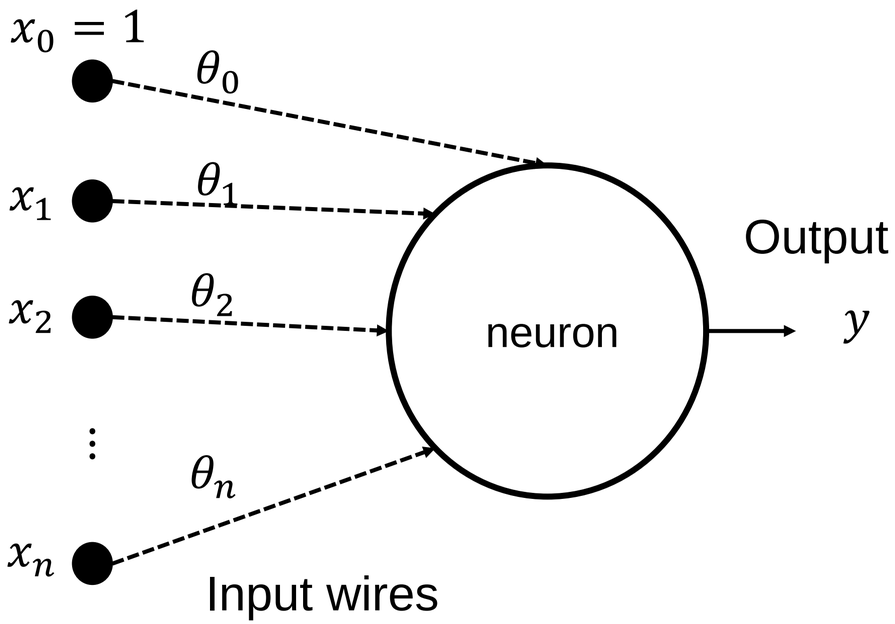




Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Artificial Neuron model



$\boldsymbol{x} = [x_0, x_1, x_2, \ldots, x_n]^T$

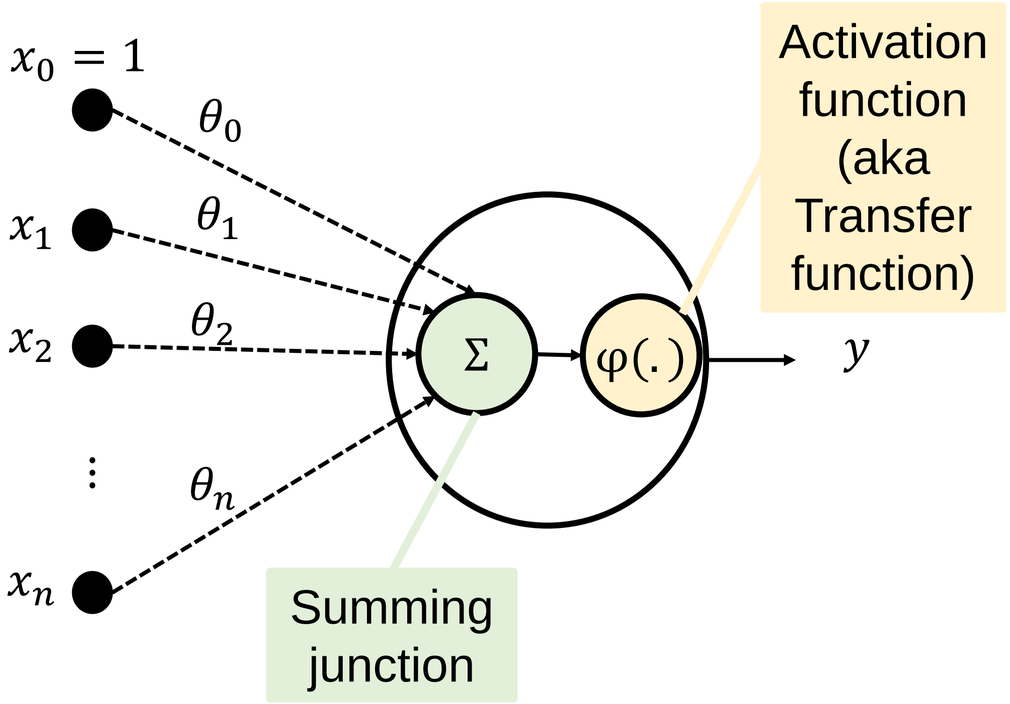
Input vector

$\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2, \ldots, \theta_n]^T$

Parameter vector (“weights”)

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Artificial Neuron model



$\boldsymbol{x} = [x_0, x_1, x_2, \ldots, x_n]^T$

Input vector

$\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2, \ldots, \theta_n]^T$

Parameter vector ("weights")

$y = \varphi(\sum_{i=0}^{n} \theta_i x_i) = \varphi(\boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{x})$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron

- A Neuron receives signals from other neurons and if these accumulated signals are strong enough, it fires
- Perceptron models this “firing” using as an activation function the Heaviside step function:

$$\varphi(x) = \begin{cases} 1 & if\ x > 0 \\ 0 & if\ x \leq 0 \end{cases}$$

- First proposed by McCulloch & Pitts (1943)
- Developed by Rosenblatt (1958) and became popular in the 60s
- Pattern classifier

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Learning

Algorithm

1. Initialize weights to 0 or small random values
2. For each example j in $D = \{(\boldsymbol{x}^{(j)}, d^{(j)})\}_{j=1:m}$
   1. Calculate actual output: $y^{(j)} = \varphi(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x})$
   2. Update weights: $\theta_i := \theta_i + \eta\,(d^{(j)} - y^{(j)})x_i^{(j)}$
      $(0 < \eta \leq 1$ learning rate)
3. Repeat for a number of iterations or until the error $|d^{(j)} - y^{(j)}|$ is below some threshold.

What it tries to achieve

Align the weight vector so as its dot product with input vectors that belong to the:

- Positive class, is positive
- Negative class, is negative

By doing so, the decision boundary (orthogonal to the weight vector) will correctly classify the two classes

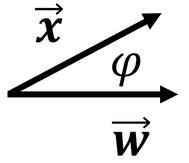
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Linear Algebra: dot product

Dot product between two vectors: $\vec{x} \cdot \vec{w} = ||\vec{x}|| \, ||\vec{w}|| \cos(\varphi)$

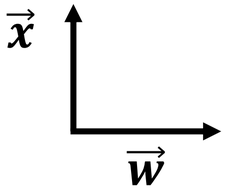
- Positive, when the vectors are pointing in the same direction

$$\vec{x} \cdot \vec{w} > 0$$



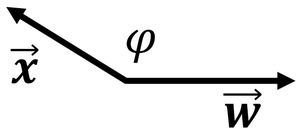
- Zero, if they are orthogonal

$$\vec{x} \cdot \vec{w} = 0$$



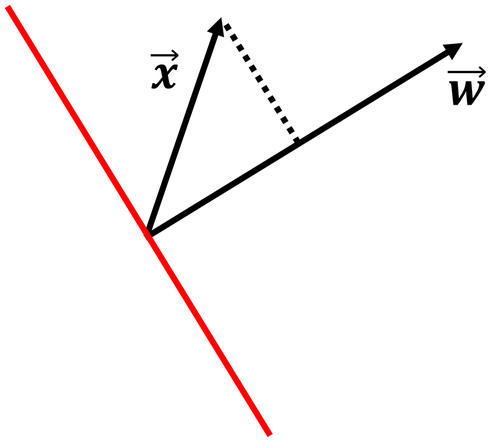
- Negative, when the vectors are pointing away from each other

$$\vec{x} \cdot \vec{w} < 0$$



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Learning



Decision line is orthogonal to the weight vector

In this example, $\vec{x} \cdot \vec{w} > 0$

If the desired output of x is:

- 1: no change will be done on the weight

$$w_i := w_i + \eta\,(0)x_i^{(j)} = w_i$$

- 0: the weight will be updated

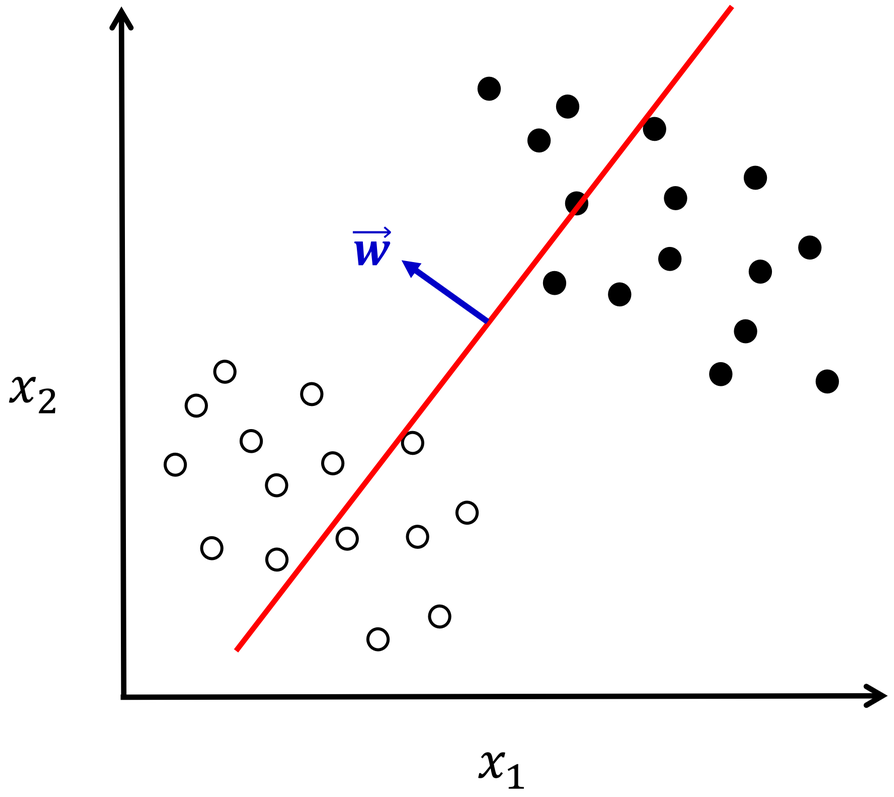
$$w_i := w_i + \eta\,(-1)x_i^{(j)}$$

Similar equations for an input of the negative class.

Co-financed by the European Union
Connecting Europe Facility

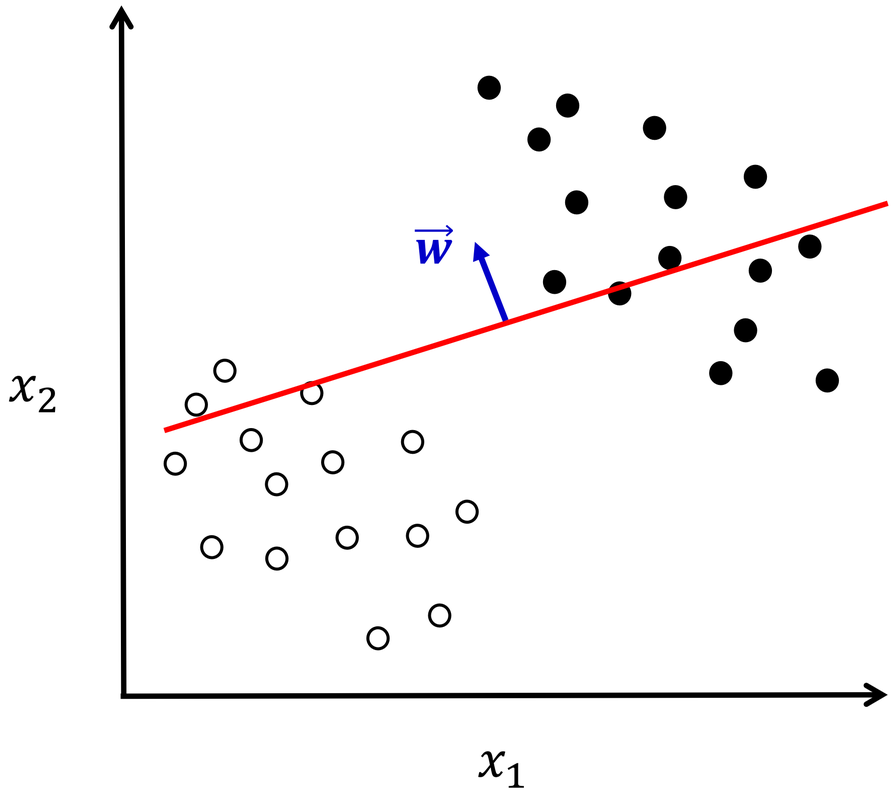
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Learning



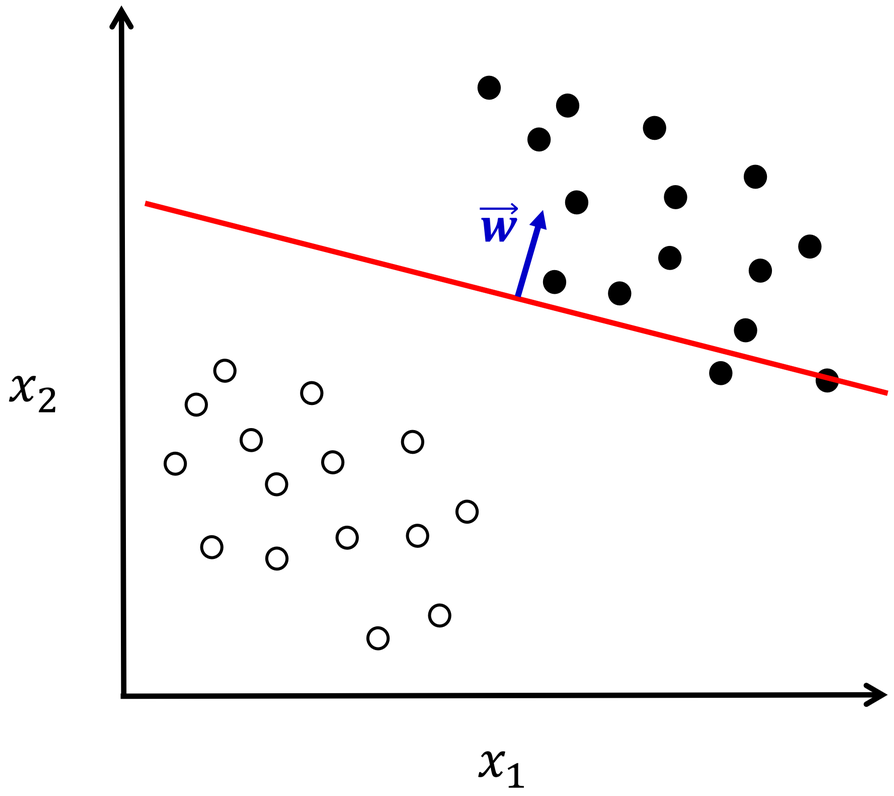
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Learning



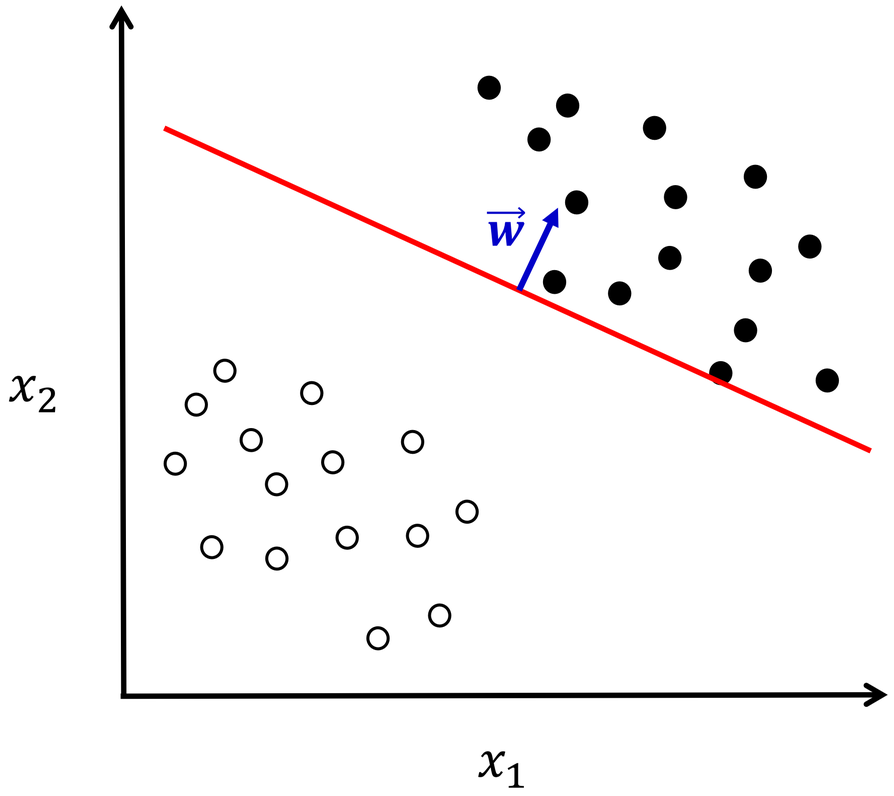
This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Learning



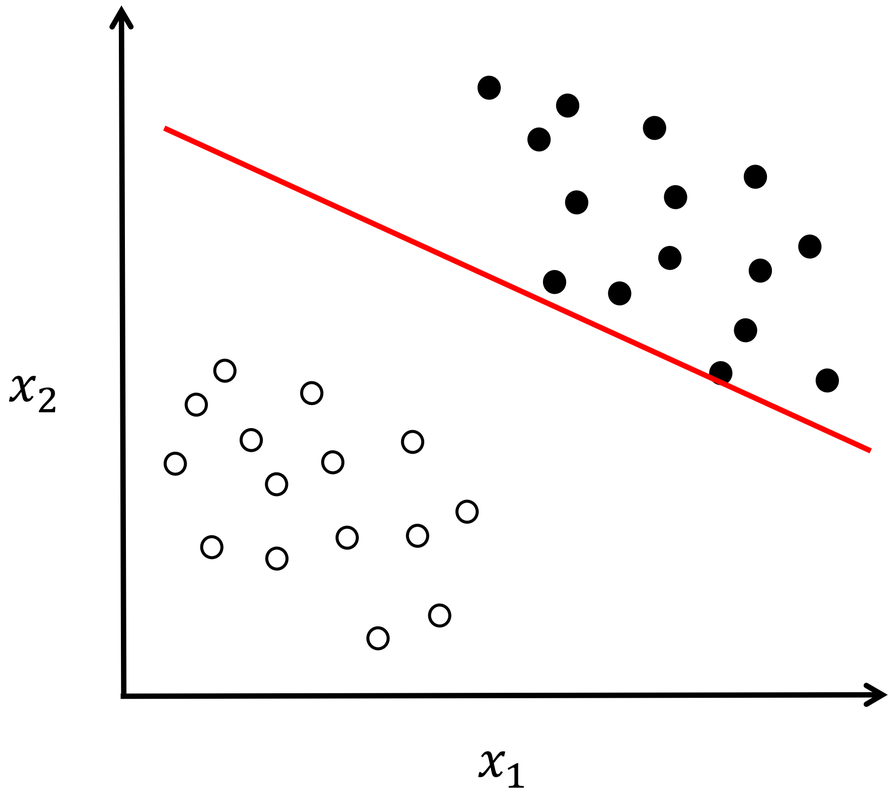
This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Learning



This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Learning



Once the decision surface separates the two classes, the perceptron learning algorithm stops

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Limitations

- Activation function
  - Output values are only 0 or 1
  - Non-differentiable: we cannot use gradient descent
- Can only classify linearly separable sets of vectors
- Lacks a concrete probabilistic framework

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Activation Functions

- Heaviside step function: $\varphi(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}) = \begin{cases} 1 & if \;\; \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x} > 0 \\ 0 & if \;\; \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x} \leq 0 \end{cases}$ — Perceptron

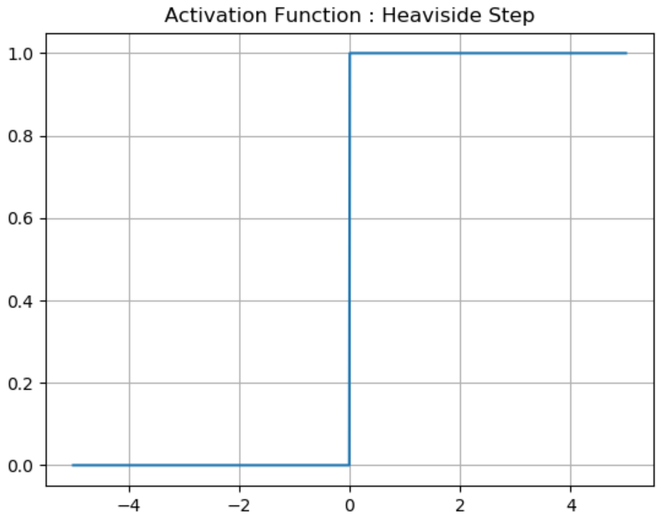
- Linear activation function: $\varphi(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}) = \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}$ — Linear regression

- Sigmoid function: $\varphi(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}) = \frac{1}{(1+e^{-\boldsymbol{a}*\boldsymbol{\theta}^{T}\boldsymbol{x}})}$ — Logistic regression

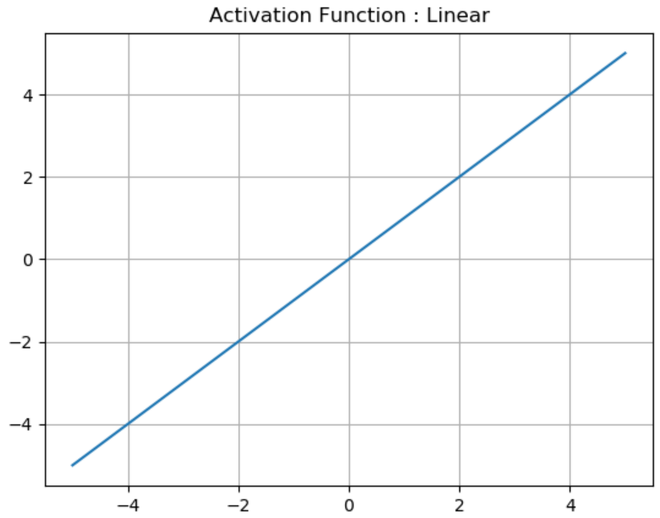
- Linear and logistic regression are equivalent to using a single neuron model with a linear and sigmoid activation function, respectively.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

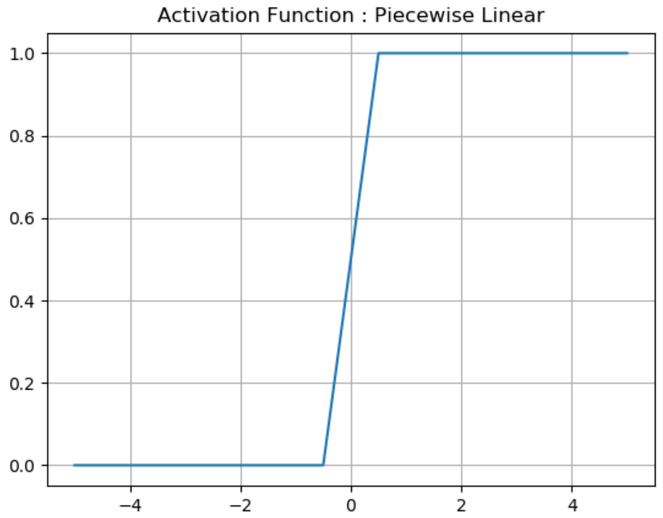
Activation Functions



$$\varphi(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases}$$



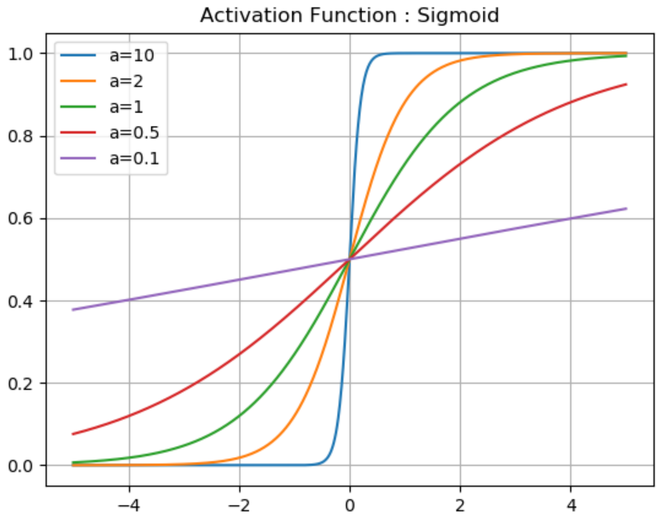
$$\varphi(x) = x$$



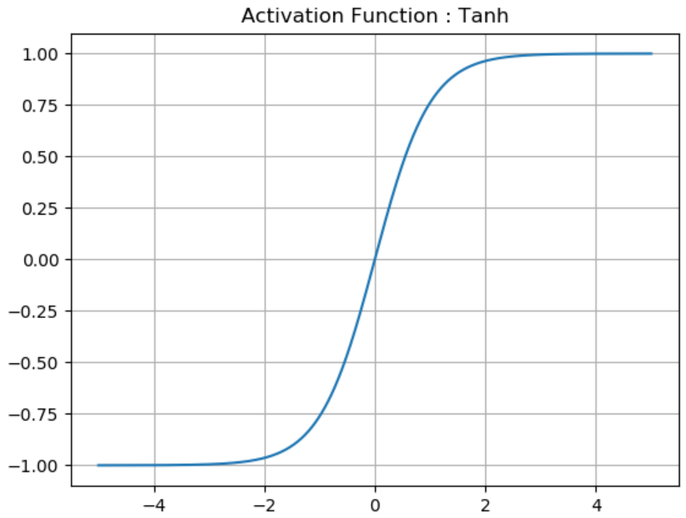
$$\varphi(x) = \begin{cases} 1 & x \geq 0.5 \\ x + 0.5 & -0.5 < x < 0.5 \\ 0 & x \leq 0.5 \end{cases}$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

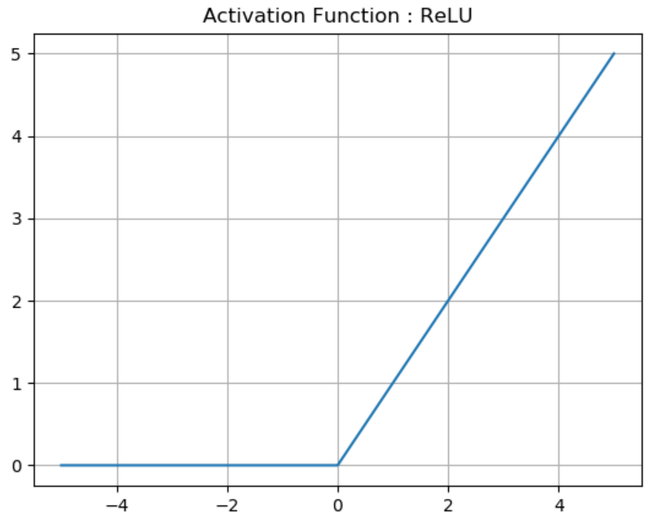
Activation Functions



$$\varphi(x) = \frac{1}{(1 + e^{-a*x})}$$



$$\varphi(x) = \tanh(x)$$



$$\varphi(x) = \max(0, x)$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Linear Separability example: NOT gate

Truth Table

x	y
0	1
1	0

Task:

Find the values of the weights in order to satisfy the truth table

$\theta_0 = ?$
$\theta_1 = ?$

$$y = \varphi(\theta_0 + \theta_1 x)$$

where

$$\varphi(z) = \begin{cases} 1 & if\ z > 0 \\ 0 & if\ z \leq 0 \end{cases}$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Linear Separability example: NOT gate

Truth Table

x	y
0	1
1	0

Task:

Find the values of the weights in order to satisfy the truth table

$$\theta_0 = 1$$
$$\theta_1 = -1.5$$

$$y = \varphi(\theta_0 + \theta_1 x)$$

where

$$\varphi(z) = \begin{cases} 1 & if\ z > 0 \\ 0 & if\ z \leq 0 \end{cases}$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Linear Separability example: OR gate

Truth Table

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

Task:

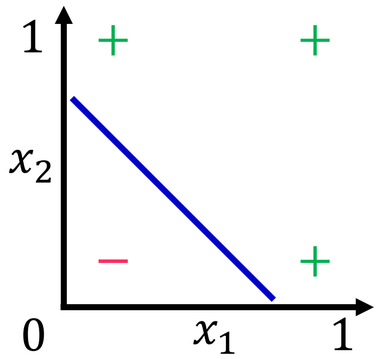
Find the values of the weights in order to satisfy the truth table

$$\theta_0 = ?$$
$$\theta_1 = ?$$
$$\theta_2 = ?$$

$$y = \varphi(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

where

$$\varphi(z) = \begin{cases} 1 & if\ z > 0 \\ 0 & if\ z \leq 0 \end{cases}$$



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Linear Separability example: OR gate

Truth Table

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

Task:

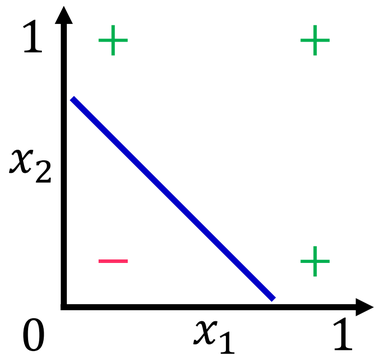
Find the values of the weights in order to satisfy the truth table

$$\theta_0 = -0.5$$
$$\theta_1 = +1$$
$$\theta_2 = +1$$

$$y = \varphi(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

where

$$\varphi(z) = \begin{cases} 1 & if\ z > 0 \\ 0 & if\ z \leq 0 \end{cases}$$



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Linear Separability example: AND gate

Truth Table

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

Task:

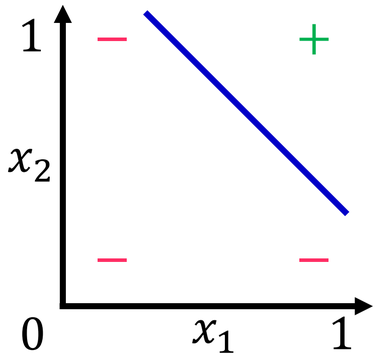
Find the values of the weights in order to satisfy the truth table

$$\theta_0 = ?$$
$$\theta_1 = ?$$
$$\theta_2 = ?$$

$$y = \varphi(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

where

$$\varphi(z) = \begin{cases} 1 & if\ z > 0 \\ 0 & if\ z \leq 0 \end{cases}$$



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Linear Separability example: AND gate

Truth Table

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

Task:

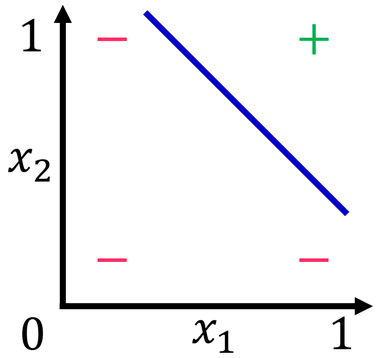
Find the values of the weights in order to satisfy the truth table

$$\theta_0 = -0.75$$
$$\theta_1 = +0.5$$
$$\theta_2 = +0.5$$

$$y = \varphi(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

where

$$\varphi(z) = \begin{cases} 1 & if\ z > 0 \\ 0 & if\ z \leq 0 \end{cases}$$



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Perceptron Linear Separability example: XOR gate

Truth Table

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

Task:

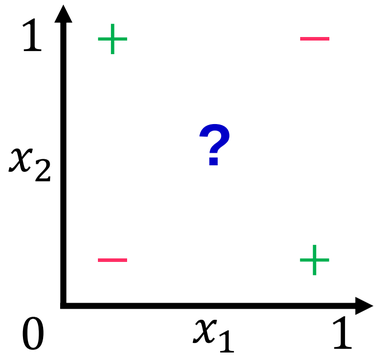
Find the values of the weights in order to satisfy the truth table

$$\theta_0 = ?$$
$$\theta_1 = ?$$
$$\theta_2 = ?$$

$$y = \varphi(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

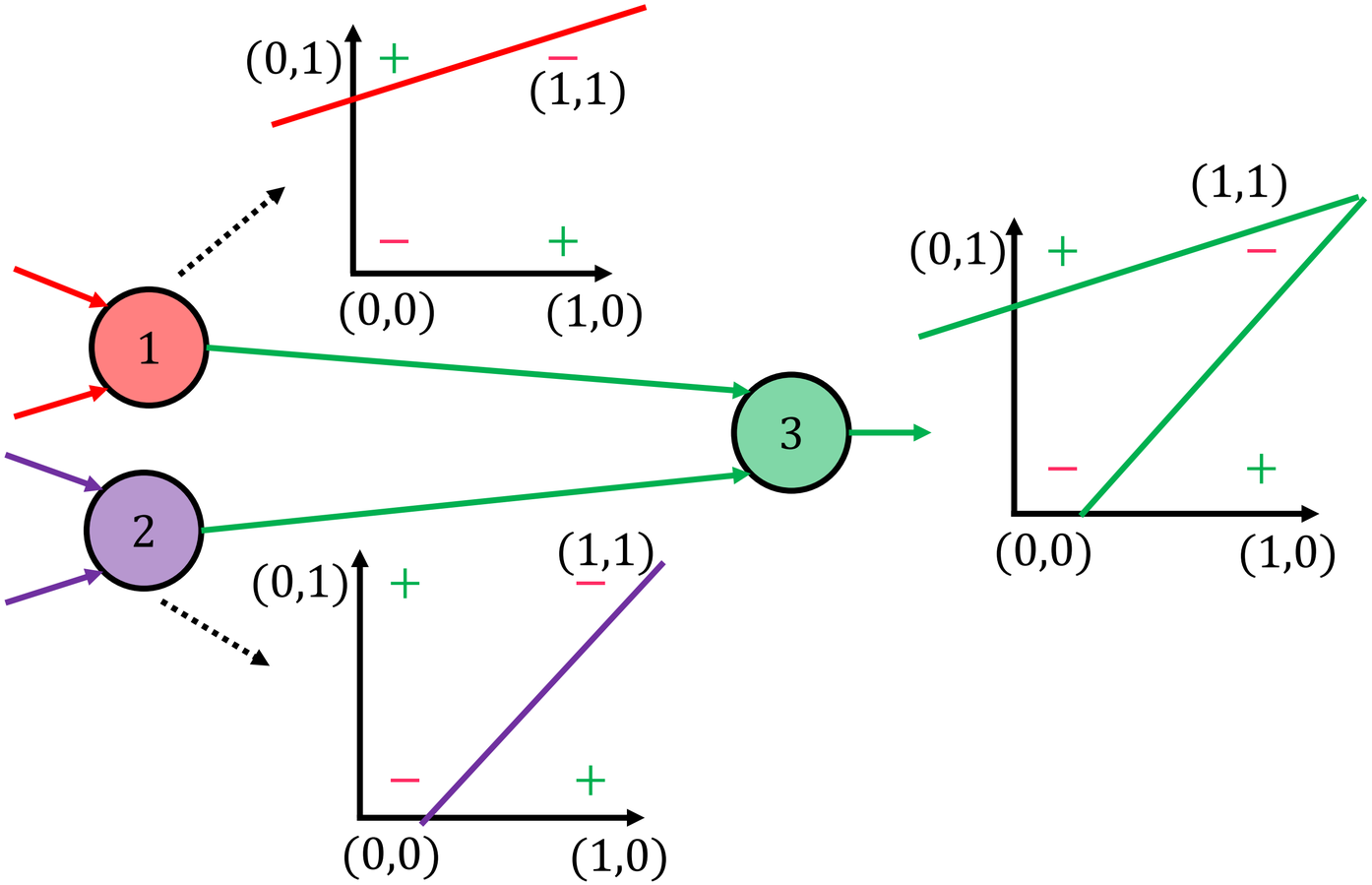
where

$$\varphi(z) = \begin{cases} 1 & if\ z > 0 \\ 0 & if\ z \leq 0 \end{cases}$$



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

How can we solve the XOR problem using perceptrons



Perceptron 1:

Detects pattern (0,1)

Perceptron 2:

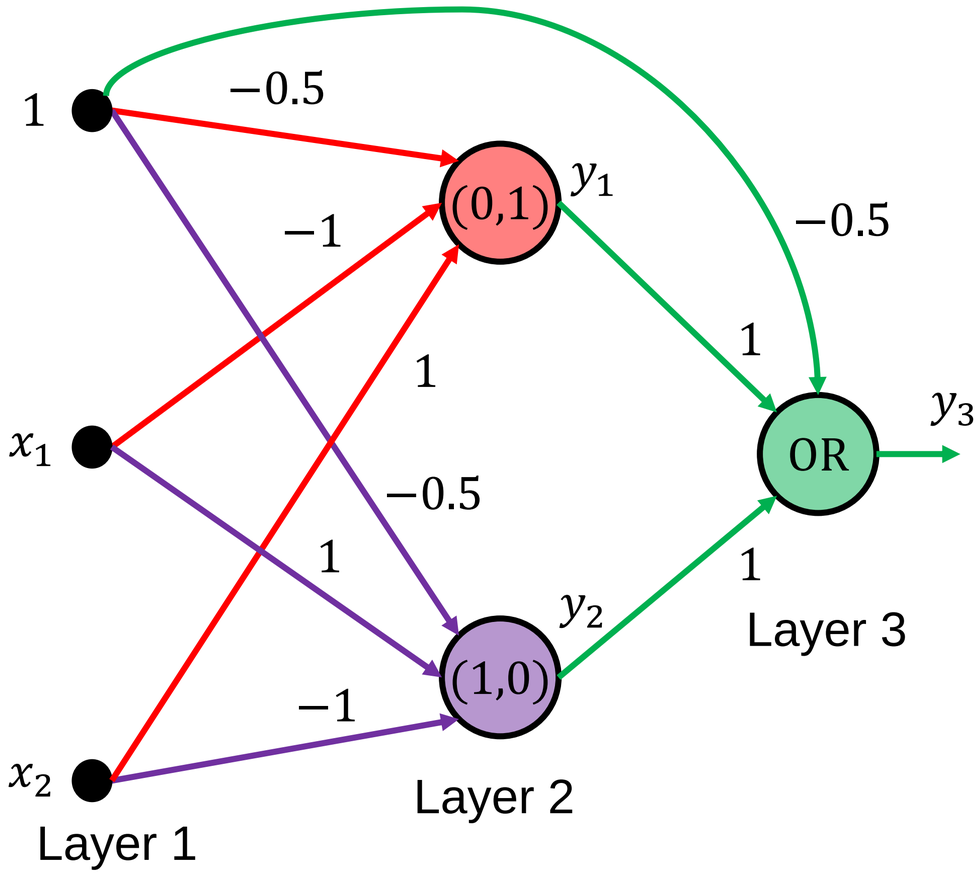
Detects pattern (1,0)

1,2 combined allow **Perceptron 3** to classify the input correctly

(as an OR gate)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

How can we solve the XOR problem using perceptrons



$$y_1 = \varphi(-0.5 - x_1 + x_2)$$
$$y_2 = \varphi(-0.5 + x_1 - x_2)$$
$$y_3 = \varphi(-0.5 + y_1 + y_2)$$

We can solve a **linearly inseparable** problem using **function composition**

Problem:

- Perceptrons in the 2nd layer do not know their desired outputs
- Cannot use information from inputs to adjust the weights, as the inputs are masked off from the output units by the hidden layer
- **The Heaviside step function needs to be changed**

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

How about the linear activation function?

It does not hide the inputs

Can we combine linear functions to solve a linearly inseparable problem?

➢ **No**

Combining linear functions gives us a linear function:

$y_1 = ax_1 + bx_2$

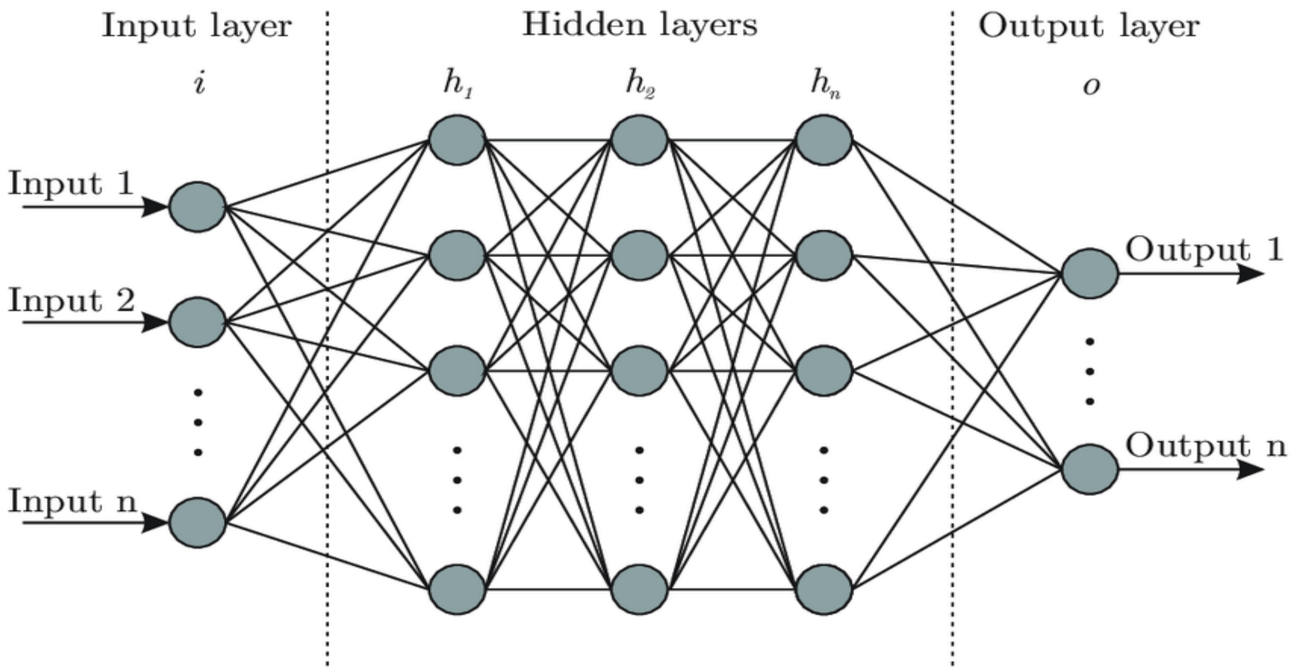
$y_2 = cx_1 + dx_2$

$$\begin{aligned} y_3 &= ey_1 + fy_2 \\ &= e(ax_1 + bx_2) + f(cx_1 + dx_2) \\ &= (eafc)x_1 + (ebfd)x_2 \\ &= \theta_1 x_1 + \theta_2 x_2 \end{aligned}$$

Need to use a nonlinear activation function!

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Multilayer Perceptron – Feedforward Neural Networks



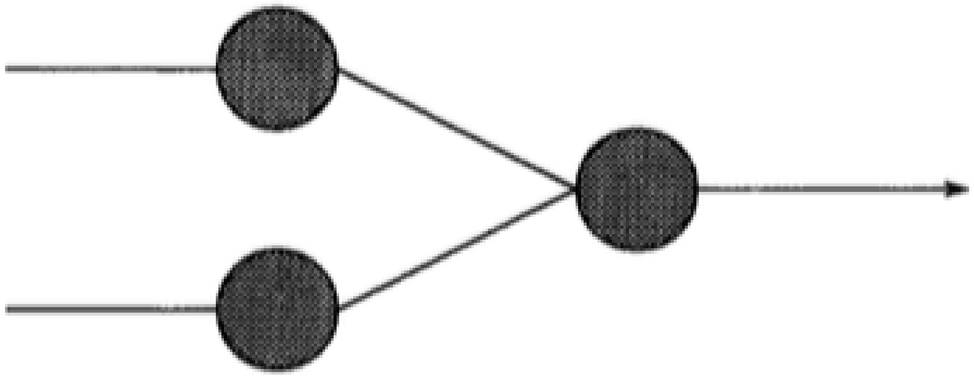
- Feedforward architecture
- Input layer:
  - Distributes the input to the first hidden layer
  - No processing (inactive)
- Hidden layers:
  - Can stack as many as we want
  - Need to have **nonlinear activation functions** (not necessarily the Heaviside step function)
- Output layer:
  - Calculates the output using an activation function appropriate for the task

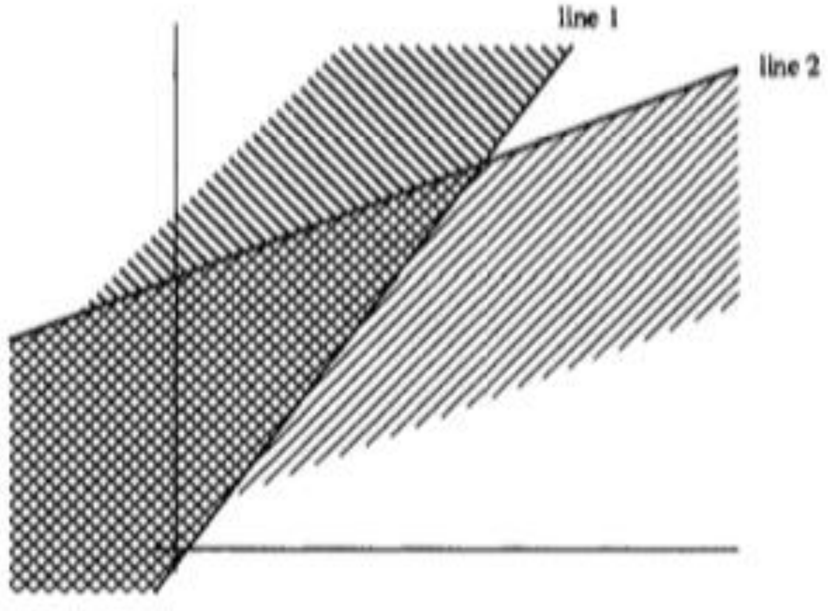
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MLPs as classifiers

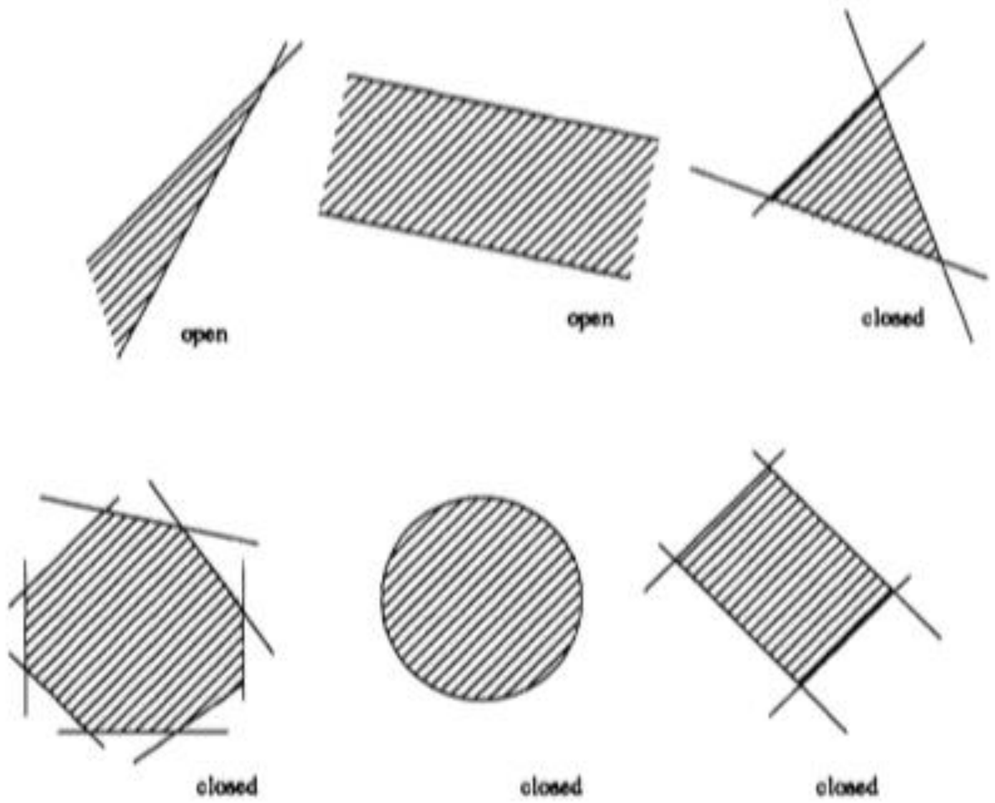
Each of the units in the first layer defines a line in pattern space and therefore the second unit produces a classification based on a combination of these lines.





This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MLPs as classifiers

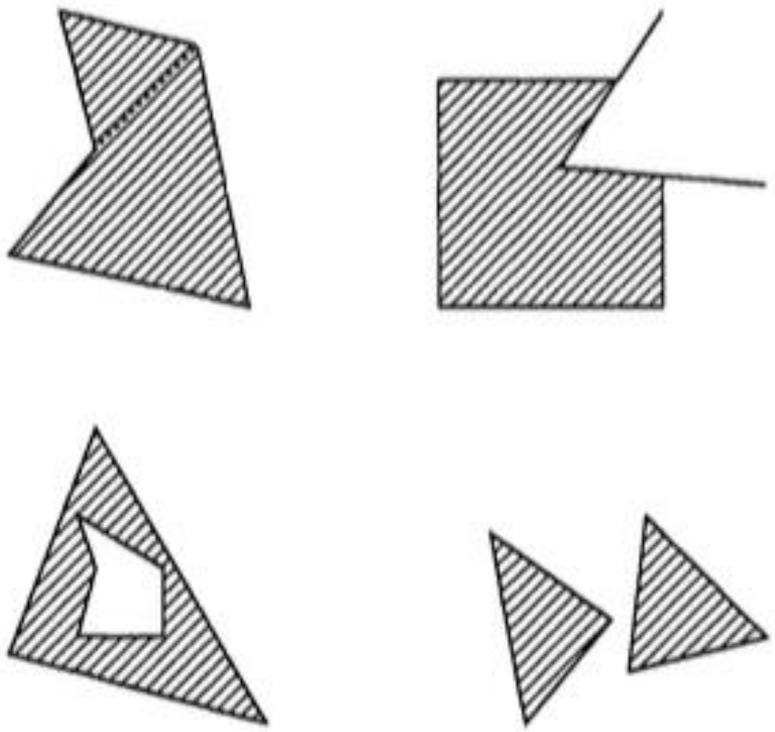


When the first layer consists of more than two units, it produces a pattern space which is a combination of more than two lines.

Regions produced in this way are known as convex regions or convex hulls.

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

MLPs as classifiers



Addition of one more layer of perceptrons allows the decision regions to take arbitrary shapes

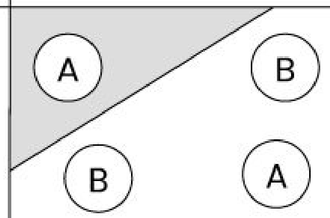
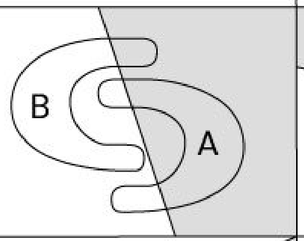
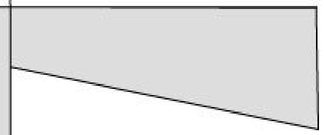
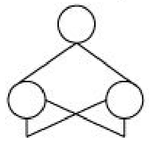
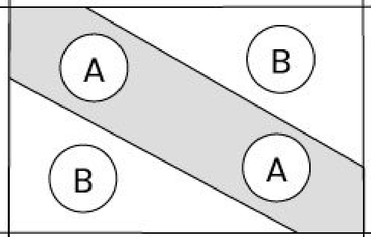
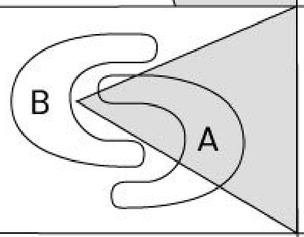
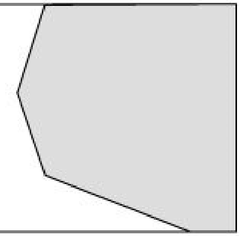
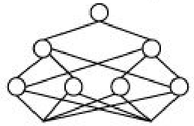
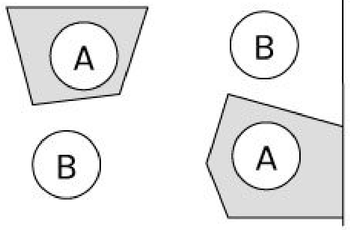
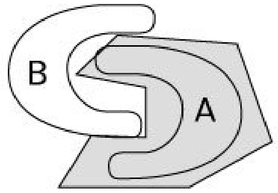
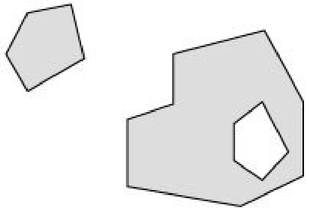
This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

MLPs as classifiers

- Three layer perceptron units (2 hidden layers) can form arbitrary complex shapes and are capable of separating any classes.
- The complexity of the shapes is limited by the number of nodes in the network.
- Thus, no more than three active layers are needed in a network

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

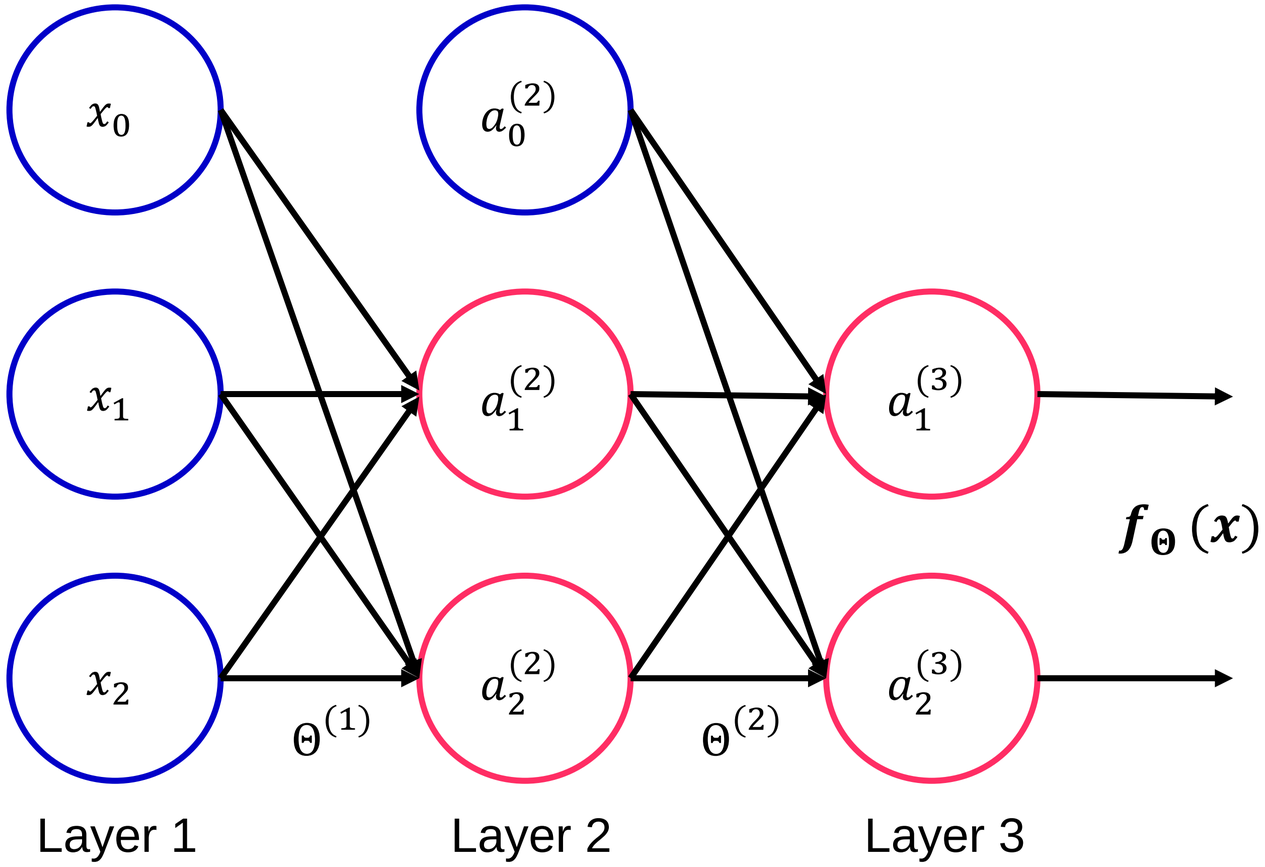
MLPs as classifiers

	Types of Decision Regions	Exclusive-OR Problem	Classes with Meshed Regions	Most General Region Shapes
Single-Layer 	Half Plane Bounded by Hyperplane			
Two-Layer 	Convex Open or Closed Regions			
Three-Layer 	Arbitrary (Complexity Limited by No. of Nodes)			

source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model Representation



$a_i^{(j)}$ = activation of unit i in layer j

$\Theta^{(j)}$ = matrix of weights between layer j and layer $j+1$

If network has n_j neurons in layer j, n_{j+1} neurons in layer $j+1$, then $\Theta^{(j)}$ will be of dimension $n_{j+1} \times (n_j + 1)$

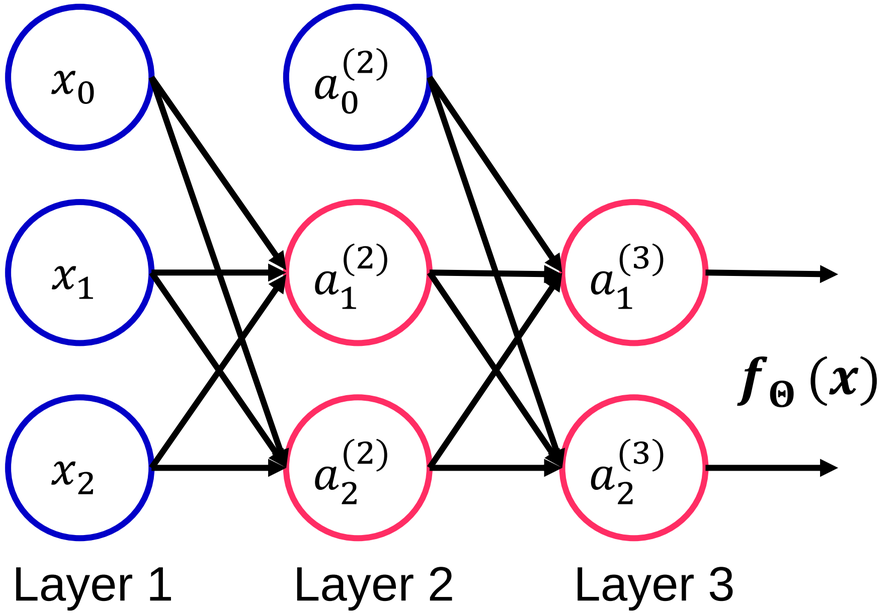
$\Theta^{(1)} \in \mathbb{R}^{2\times 3}$

$\Theta^{(2)} \in \mathbb{R}^{2\times 3}$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model Representation



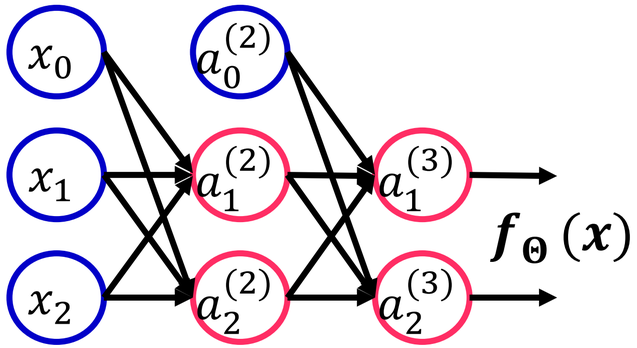
$$a_1^{(2)} = \varphi(\Theta_{1,0}^{(1)} x_0 + \Theta_{1,1}^{(1)} x_1 + \Theta_{1,2}^{(1)} x_2)$$
$$a_2^{(2)} = \varphi(\Theta_{2,0}^{(1)} x_0 + \Theta_{2,1}^{(1)} x_1 + \Theta_{2,2}^{(1)} x_2)$$

$$a_1^{(3)} = \varphi(\Theta_{1,0}^{(2)} a_0^{(2)} + \Theta_{1,1}^{(2)} a_1^{(2)} + \Theta_{1,2}^{(2)} a_2^{(2)})$$
$$a_2^{(3)} = \varphi(\Theta_{2,0}^{(2)} a_0^{(2)} + \Theta_{2,1}^{(2)} a_1^{(2)} + \Theta_{2,2}^{(2)} a_2^{(2)})$$

$$f_\Theta(x) = \begin{bmatrix} a_1^{(3)} \\ a_2^{(3)} \end{bmatrix}$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Forward propagation: vectorized implementation



$$a_1^{(2)} = \varphi(\Theta_{1,0}^{(1)} x_0 + \Theta_{1,1}^{(1)} x_1 + \Theta_{1,2}^{(1)} x_2)$$
$$a_2^{(2)} = \varphi(\Theta_{2,0}^{(1)} x_0 + \Theta_{2,1}^{(1)} x_1 + \Theta_{2,2}^{(1)} x_2)$$

$$a_1^{(3)} = \varphi(\Theta_{1,0}^{(2)} a_0^{(2)} + \Theta_{1,1}^{(2)} a_1^{(2)} + \Theta_{1,2}^{(2)} a_2^{(2)})$$
$$a_2^{(3)} = \varphi(\Theta_{2,0}^{(2)} a_0^{(2)} + \Theta_{2,1}^{(2)} a_1^{(2)} + \Theta_{2,2}^{(2)} a_2^{(2)})$$

$$\boldsymbol{x} = [x_0, x_1, x_2]^T = \boldsymbol{a}^{(1)}$$
$$\boldsymbol{z}^{(2)} = \left[z_1^{(2)}, z_2^{(2)}\right]^T$$

→ Input to hidden layer 1 (needs to be 2D)

$$\boldsymbol{z}^{(2)} = \Theta^{(1)} \boldsymbol{a}^{(1)}$$
$$\boldsymbol{a}^{(2)} = \boldsymbol{\varphi}(\boldsymbol{z}^{(2)})$$
Add $a_0^{(2)} = 1$

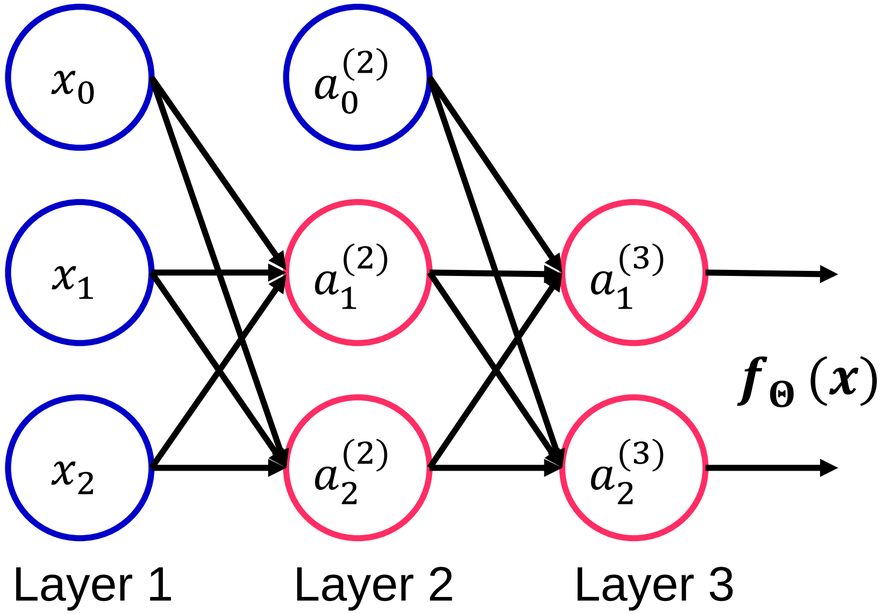
$$\boldsymbol{z}^{(3)} = \Theta^{(2)} \boldsymbol{a}^{(2)}$$
$$\boldsymbol{a}^{(3)} = \boldsymbol{\varphi}(\boldsymbol{z}^{(3)})$$

$$\boldsymbol{f}_\Theta(\boldsymbol{x}) = \left[a_1^{(3)}, a_2^{(3)}\right]^T$$

$$\boldsymbol{z}^{(j+1)} = \Theta^{(j)} \boldsymbol{a}^{(j)}$$
$$\boldsymbol{a}^{(j+1)} = \boldsymbol{\varphi}(\boldsymbol{z}^{(j+1)})$$
Add $a_0^{(j+1)} = 1$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression



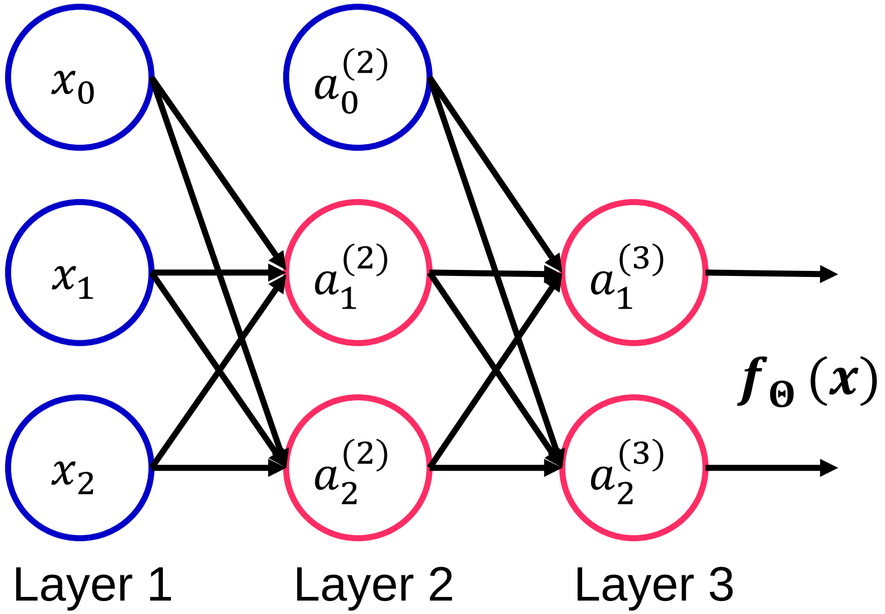
Use a **linear activation function** at the output layer:

$$\boldsymbol{z^{(3)} = \Theta^{(2)} a^{(2)}}$$

$$\boldsymbol{a^{(3)} = \varphi\left(z^{(3)}\right) = z^{(3)}}$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Classification



Binary classification: 1 output neuron with the sigmoid activation function

y = 0 or 1

Multiclass classification (K classes): K output neurons with the softmax activation function:

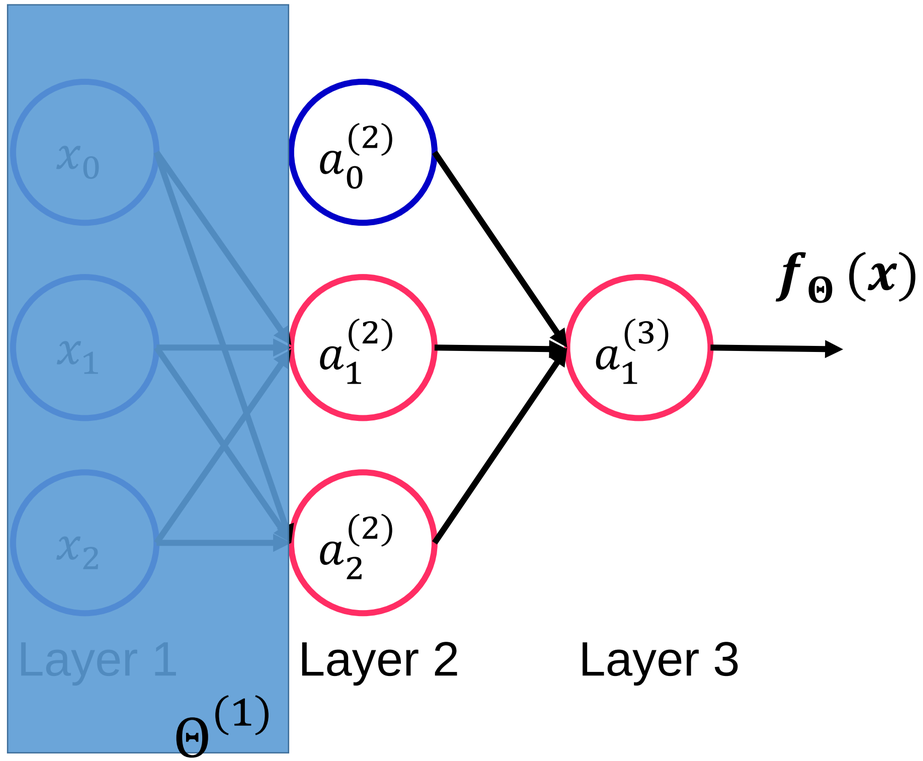
$$a_i^{(3)} = \varphi\left(z_i^{(3)}\right) = \frac{exp(z_i^{(3)})}{\sum_{k=1}^{n} exp(z_j^{(3)})}$$

e.g. $\begin{bmatrix}1\\0\\0\end{bmatrix}$, $\begin{bmatrix}0\\1\\0\end{bmatrix}$, $\begin{bmatrix}0\\0\\1\end{bmatrix}$

dog cat horse

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neural Networks learn their own features



Example: binary classification

Nodes from Layer 2 are like input **features** in *logistic regression*, where

$$x' = \left[a_0^{(2)}, a_1^{(2)}, a_2^{(2)}\right]^T$$

In a NN the features are:

- **Learned** as functions of the input (using another set of parameters $\Theta^{(1)}$)

- Not constrained to manually designed ones (e.g., polynomial terms)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neural Networks learn their own features

- Stacking more hidden layers enables the model to learn more complex features as functions of existing features, tuned for the task at hand

- This can be **dramatically more efficient** than other feature engineering approaches. For example, consider building a *car image classifier* based on 50×50 pixel images.
  - Input dimensionality: $n = 2500$ (7500 if RGB)
  - Quadratic features ($x_i \times x_j$) $\approx$ 3 million features/parameters
  - NN (100 units in Layer 2, 50 units in L3) $\approx$ 255K parameters

$$\boldsymbol{x} = \begin{bmatrix} pixel\ 1\ intensity \\ \vdots \\ pixel\ 2500\ intensity \end{bmatrix}$$

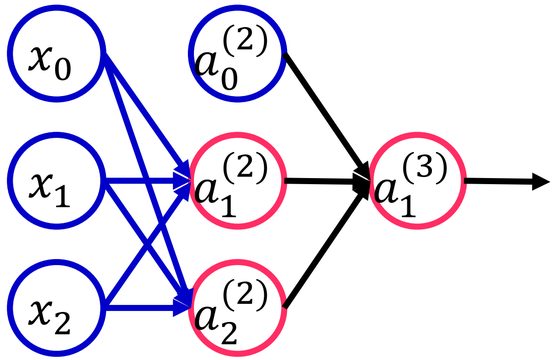
- **Universal function approximators**: neural networks with 1 hidden layer with an arbitrary width and a smooth activation function can approximate any continuous function

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Quiz

- A neural network with no hidden layers and a linear activation function at its output unit is equivalent to the linear regression model. True or False?

- The following NN has one hidden layer, where the parameters between the input layer and the hidden layer are frozen, and all active nodes have the sigmoid activation function. This is equivalent to a logistic regression model. True or False?



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 10: Neural Networks 2: Training

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 10: Neural Networks 1: Training

Learning Outcomes

You will learn about:

1. The cost function of neural network models for regression and classification
2. How to train neural networks using backpropagation
3. How to implement backpropagation efficiently
4. Stochastic gradient descent with momentum
5. Advanced optimization methods for training neural networks
6. How to improve the performance of neural networks using early stopping, hyperparameter tuning and ensembles
7. Evolving neural networks

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression cost function

Linear regression + Regularization:

$$L(\boldsymbol{\theta}) = \frac{1}{2m}\left[\sum_{i=1}^{m}(f_{\boldsymbol{\theta}}(x^{(i)}) - y^{(i)})^2\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

Neural network + Regularization:

$$L(\Theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K}\left(\left(f_{\Theta}(x^{(i)})\right)_k - y_k^{(i)}\right)^2\right] + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{n_l}\sum_{j=1}^{n_{l+1}}\left(\Theta_{j,i}^{(l)}\right)^2$$

K output units ($\boldsymbol{y} \in \mathbb{R}^K$)

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Multiclass classification cost function

Multinomial logistic regression + Regularization:

$$L(\boldsymbol{\theta}) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[y_k^{(i)}\, log\left(f_{\boldsymbol{\theta}}(x^{(i)})\right)_k\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

Neural network + Regularization:

$$L(\Theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[y_k^{(i)}\, log\left(f_{\Theta}(x^{(i)})\right)_k\right] + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{n_l}\sum_{j=1}^{n_{l+1}}\left(\Theta_{j,i}^{(l)}\right)^2$$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gradient Computation

Objective: minimize $L(\Theta)$
Θ

Using **Gradient Descent**

Need to compute: $L(\Theta)$

$$\frac{\partial}{\partial \Theta_{j,i}^{(l)}} L(\Theta)$$

Observations:

- A NN has more than 1 parameter to optimize
- A NN is a composition of functions

We need to know how to compute partial derivatives of a composite function

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Partial derivatives

- For a function of two or more variables $f(x_1, x_2)$ there is no single derivative but partial derivatives with respect to each variable.
- These are just the ordinary derivatives of the function with respect to one variable while holding the other variables constant.

- Example: for $f(x_1, x_2) = x_1^2 + 3x_2 - x_1 x_2$

$$\frac{\partial f}{\partial x_1} = 2x_1 - x_2$$

$$\frac{\partial f}{\partial x_2} = 3 - x_1$$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Composite functions

- For a composite function $y = f[g(x)]$ we can apply the **chain rule** to get $f'(x)$:

$$f'(x) = \frac{dy}{dx} = \frac{dy}{dg}\frac{dg}{dx} = f'[g(x)]\, g'(x)$$

- Although the argument of f is another function $g(x)$, we treat $g(x)$ as if it were just a regular variable and differentiate f using ordinary differentiation with respect to that variable.
- We then multiply the derivative of g with respect to x, i.e., $g'(x)$, to get the complete derivative.

- Similarly if $y = f(u)$ where $u = g(v)$ and $v = h(x)$ then:

$$\frac{dy}{dx} = \frac{dy}{du}\frac{du}{dv}\frac{dv}{dx}$$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Composite functions

Example: $y = f[g(x)]$

$g(x) = x^2$
$f(z) = 3z$

$z = g(x)$
$\Rightarrow f\big(g(x)\big) = f(x^2) = 3x^2$
$\Rightarrow f'(x) = \frac{df}{dx} = 6x$

Chain rule:

$$\frac{dy}{dx} = \frac{dy}{dg}\,\frac{dg}{dx}$$

$$= 3 \cdot 2x$$
$$= 6x$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

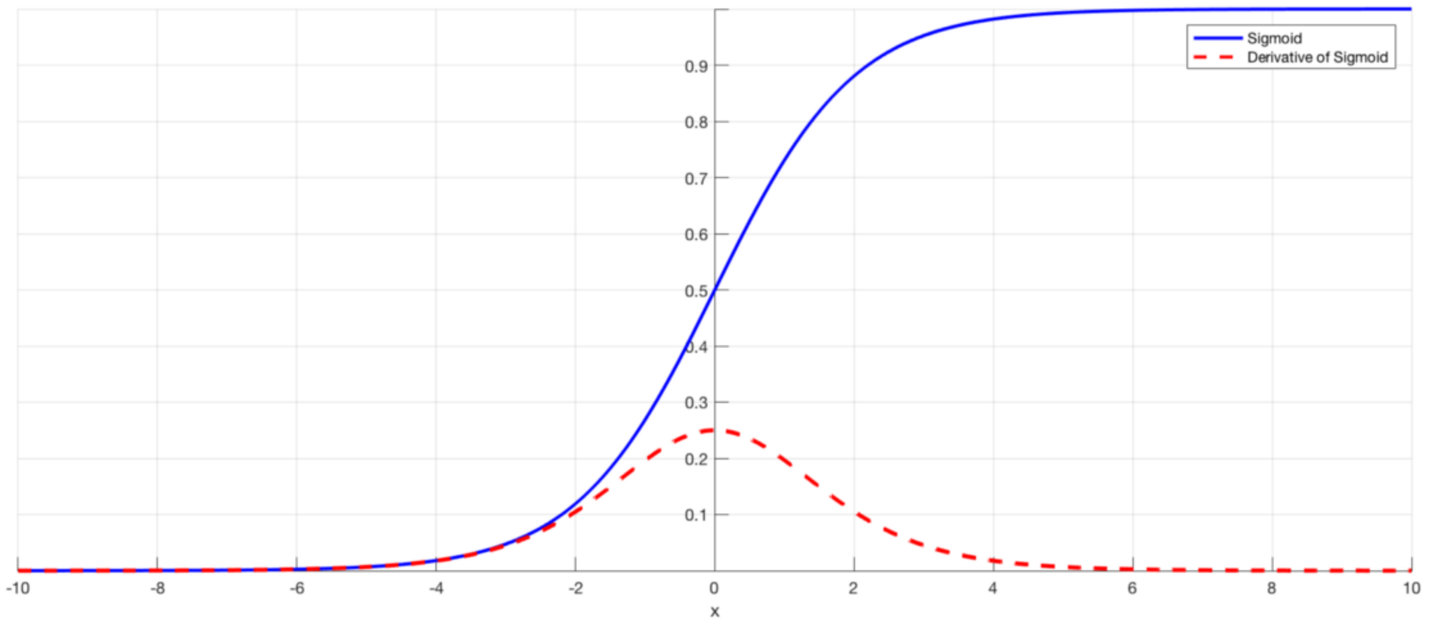
Derivative of the sigmoid function

$$y = f(x) = \frac{1}{1 + \exp(-ax)}$$

$$f(x) = \frac{d}{dx}\left(\frac{1}{1 + \exp(-ax)}\right)$$

$$= \frac{a \exp(-ax)}{(1 + \exp(-ax))^2}$$

$$= a\, f(x)\left(1 - f(x)\right)$$



Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Backpropagation algorithm

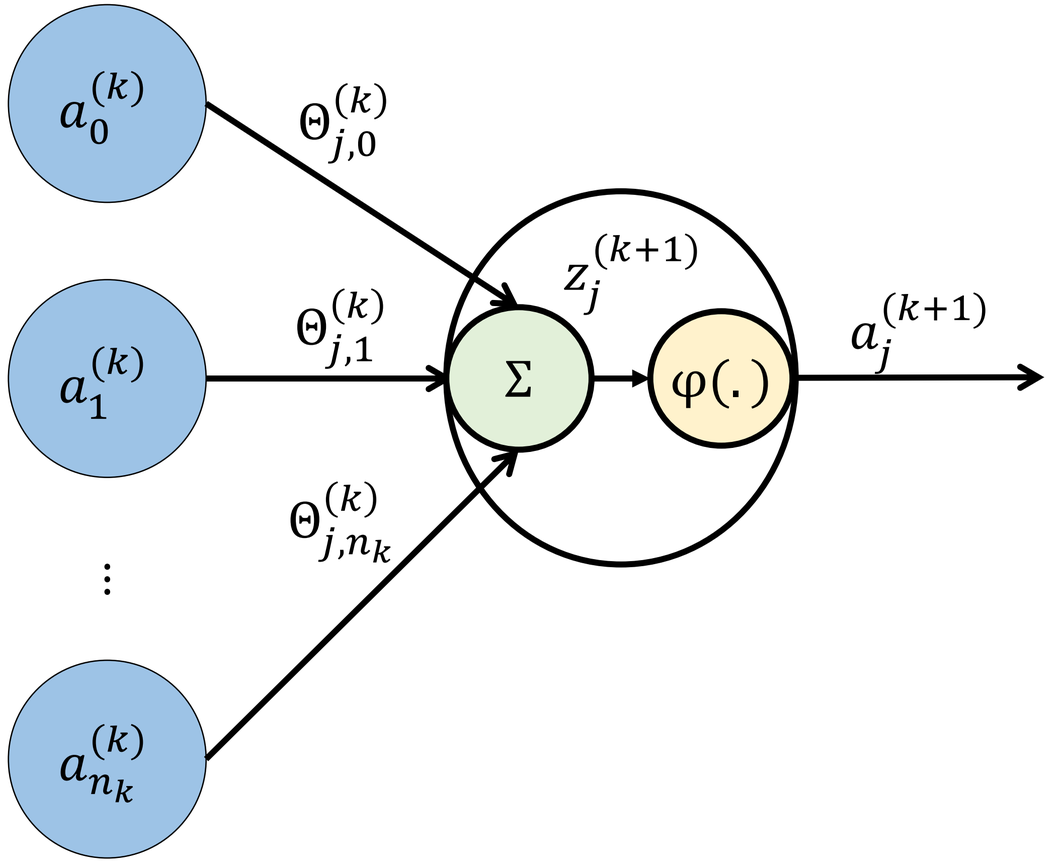
- **Forward propagation**:
  - Takes an input pattern and transforms it by propagating it forward into the network
  - Computes the activation of each node $a_i^{(k)}$ in order to compute the error

- **Backpropagation**:
  - Computes the gradient at a specific input pattern
  - "Reverses the flow of information" by propagating errors backwards in the network (uses the chain rule)
  - Intuitively: each node needs to compute its own error: $\delta_i^{(k)}$

- Backpropagation is sometimes referred to as an algorithm that trains a neural network.
  - It is actually an efficient way of computing the gradient
  - Various gradient-based optimization methods can be used (typically gradient descent)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Forward propagation



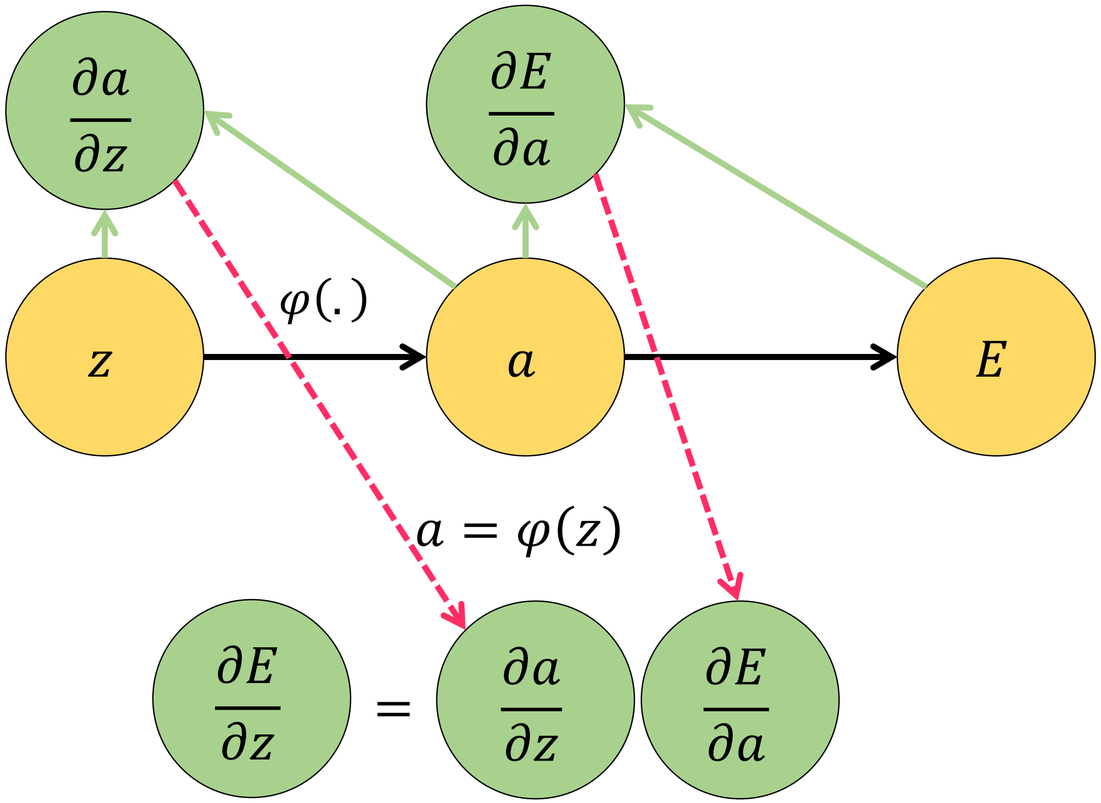
$$z_j^{(k+1)} = \sum_{i=0}^{n_k} \Theta_{j,i}^{(k)} a_i^{(k)}$$

$$a_j^{(k+1)} = \varphi\left(z_j^{(k+1)}\right)$$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Backpropagation

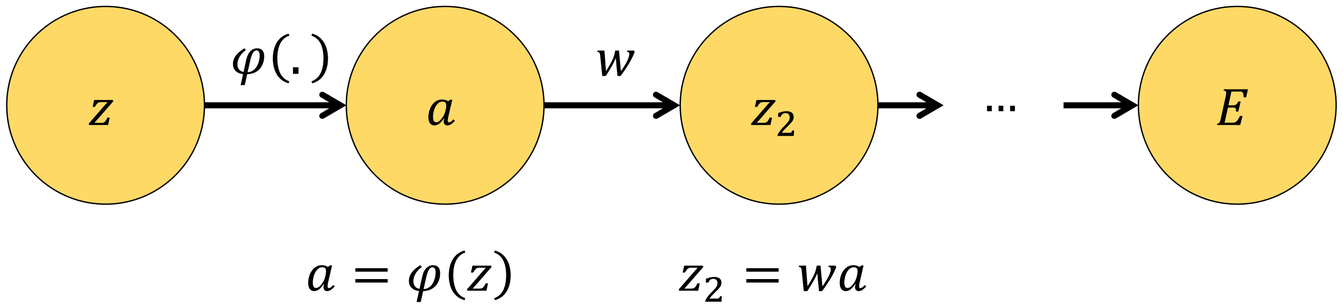


$$\frac{\partial E}{\partial z} = \frac{\partial E}{\partial a}\frac{\partial a}{\partial z}$$

$$\frac{\partial E}{\partial z} = \frac{\partial E}{\partial a}\varphi'(z)$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Backpropagation



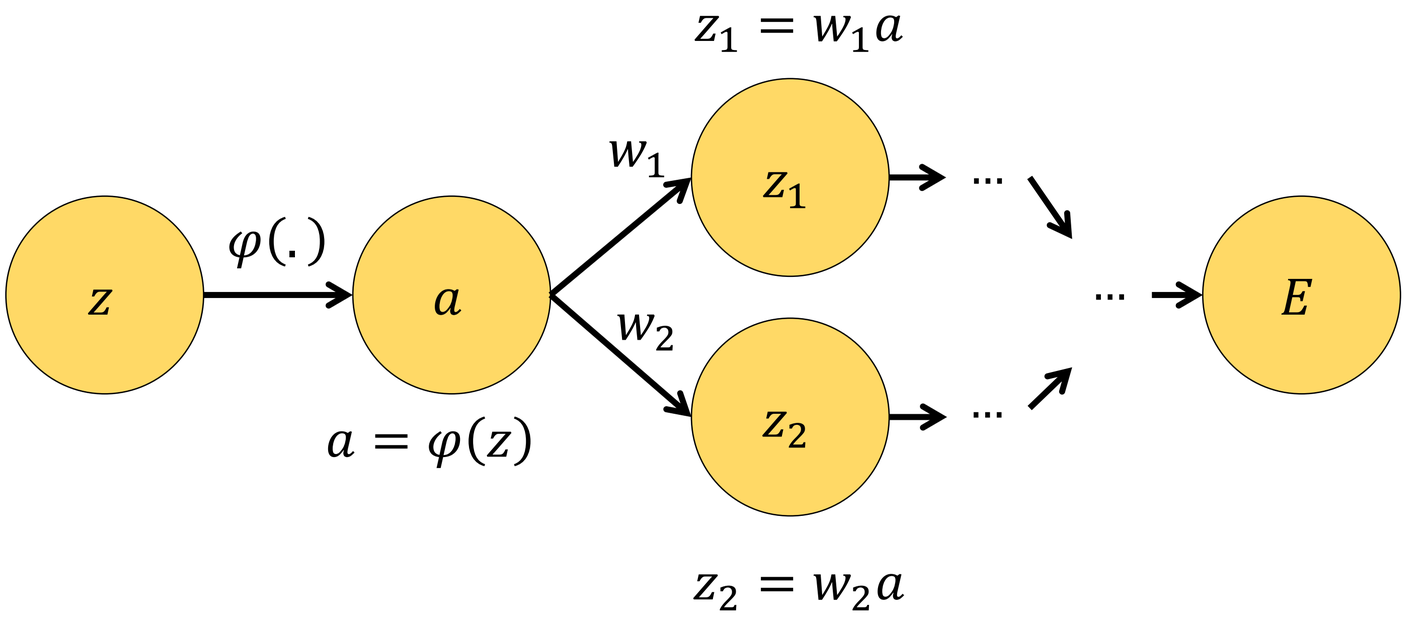
$$\frac{\partial E}{\partial z} = \frac{\partial E}{\partial z_2}\frac{\partial z_2}{\partial a}\frac{\partial a}{\partial z}$$

$$\frac{\partial E}{\partial z} = \frac{\partial E}{\partial z_2}\, w\, \varphi'(z)$$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Backpropagation



$$\frac{\partial E}{\partial z} = \left(\frac{\partial E}{\partial z_1}\frac{\partial z_1}{\partial a} + \frac{\partial E}{\partial z_2}\frac{\partial z_2}{\partial a}\right)\frac{\partial a}{\partial z}$$

$$\frac{\partial E}{\partial z} = \varphi'(z)\sum_{k=1}^{\#out} w_k \frac{\partial E}{\partial z_k}$$

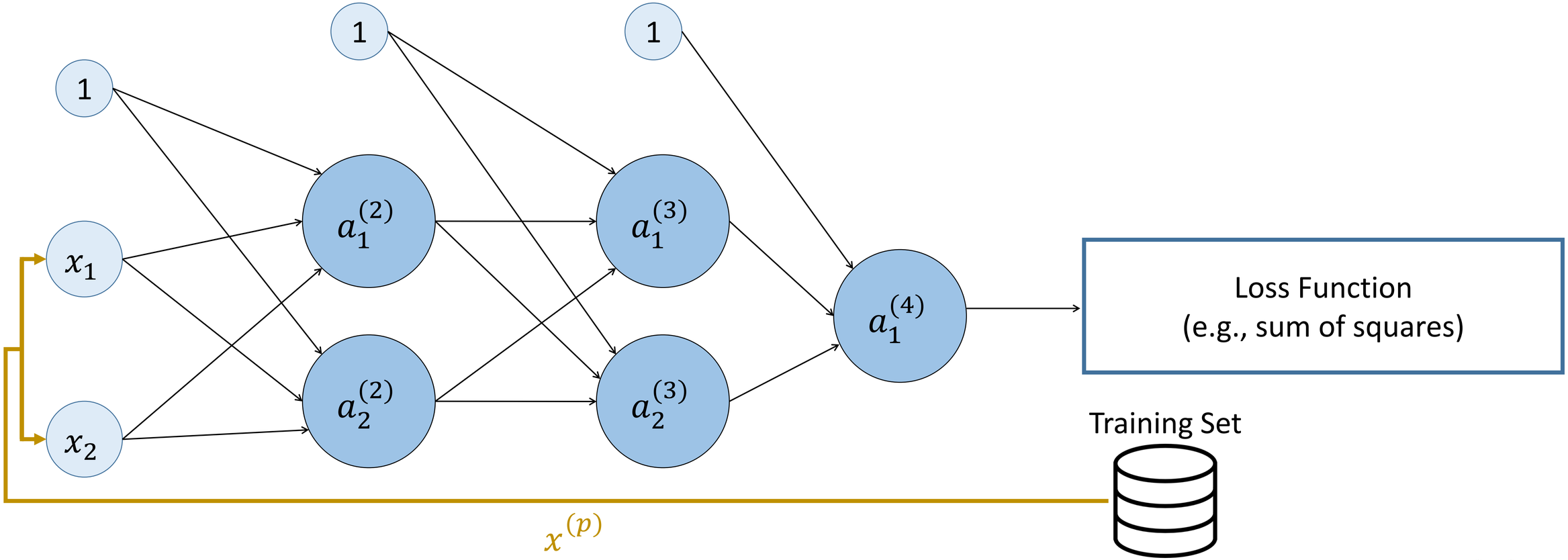
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Backpropagation Algorithm Online Updating

Step by Step

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

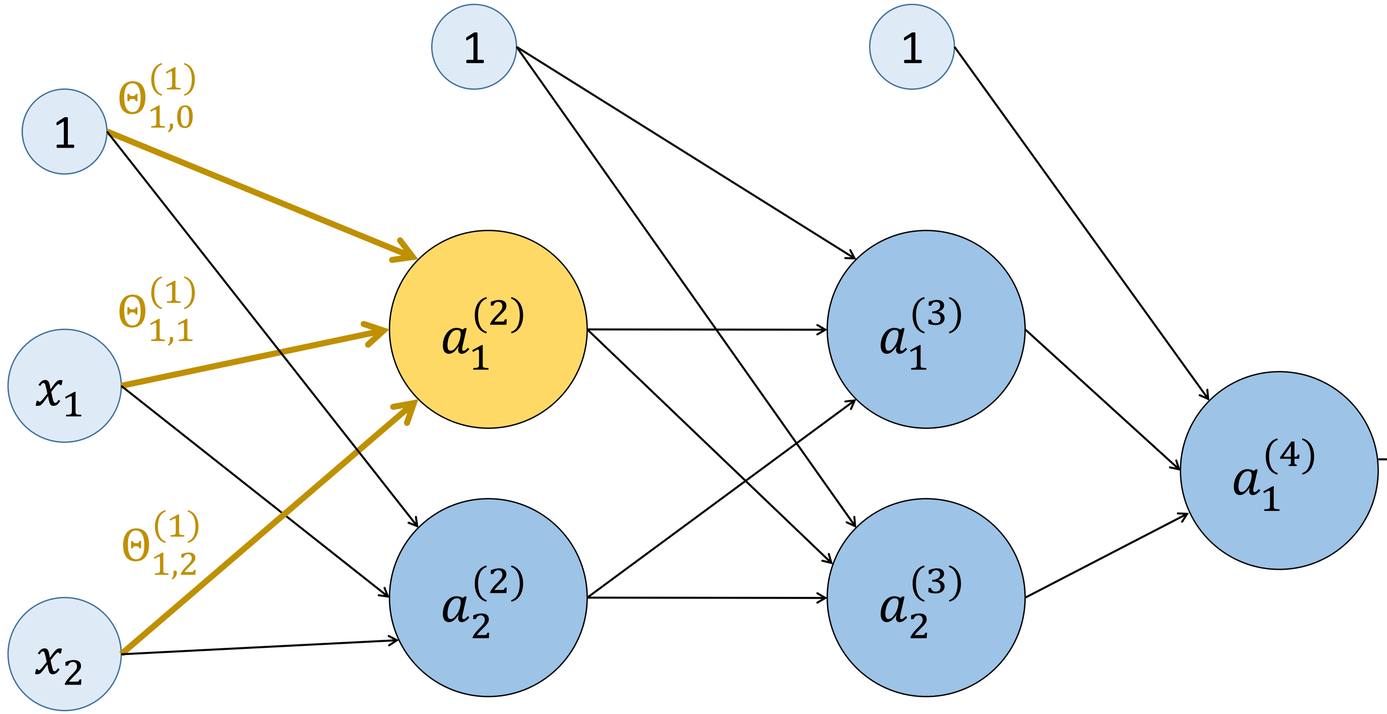
Pattern p is presented



Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Forward propagation: Compute node outputs



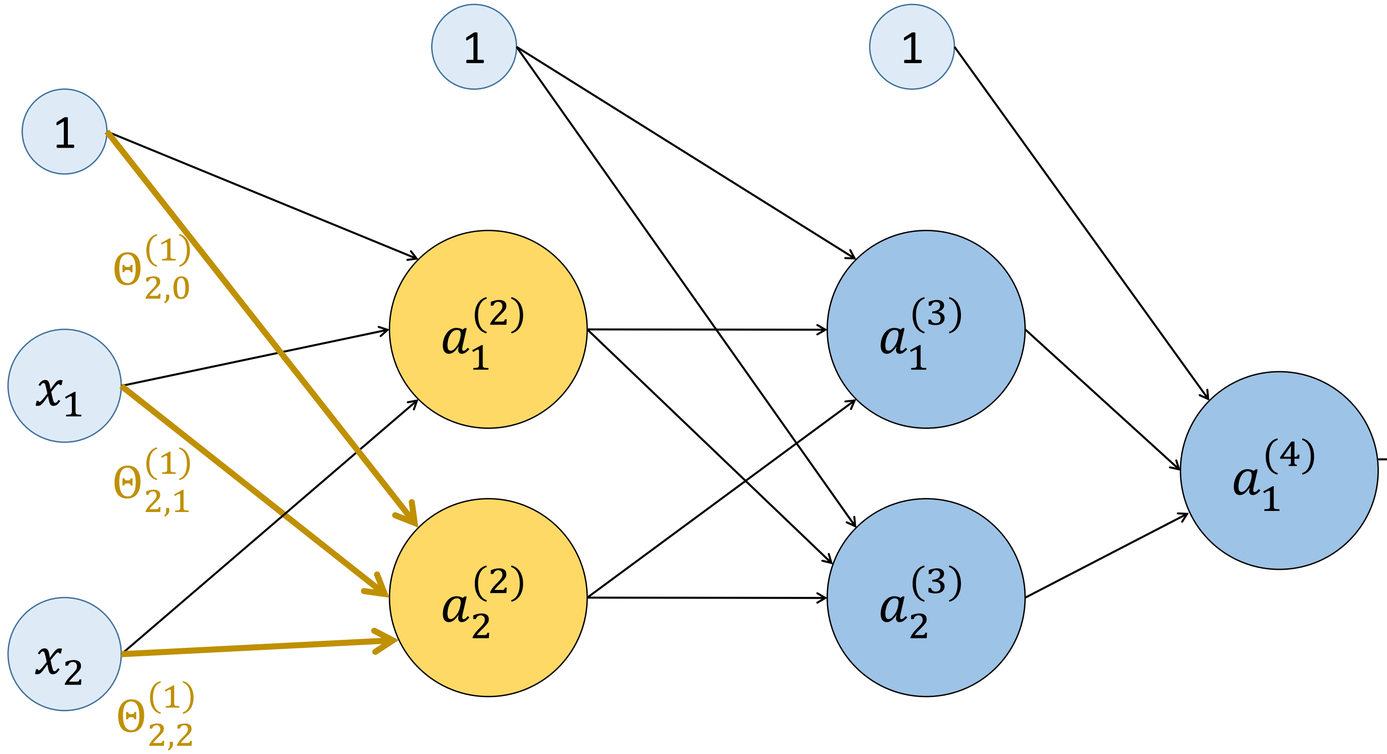
$$a_1^{(2)} = \varphi(\Theta_{1,0}^{(1)} + \Theta_{1,1}^{(1)} x_1 + \Theta_{1,2}^{(1)} x_2)$$

Loss Function
(e.g., sum of squares)

Training Set

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Forward propagation: Compute node outputs



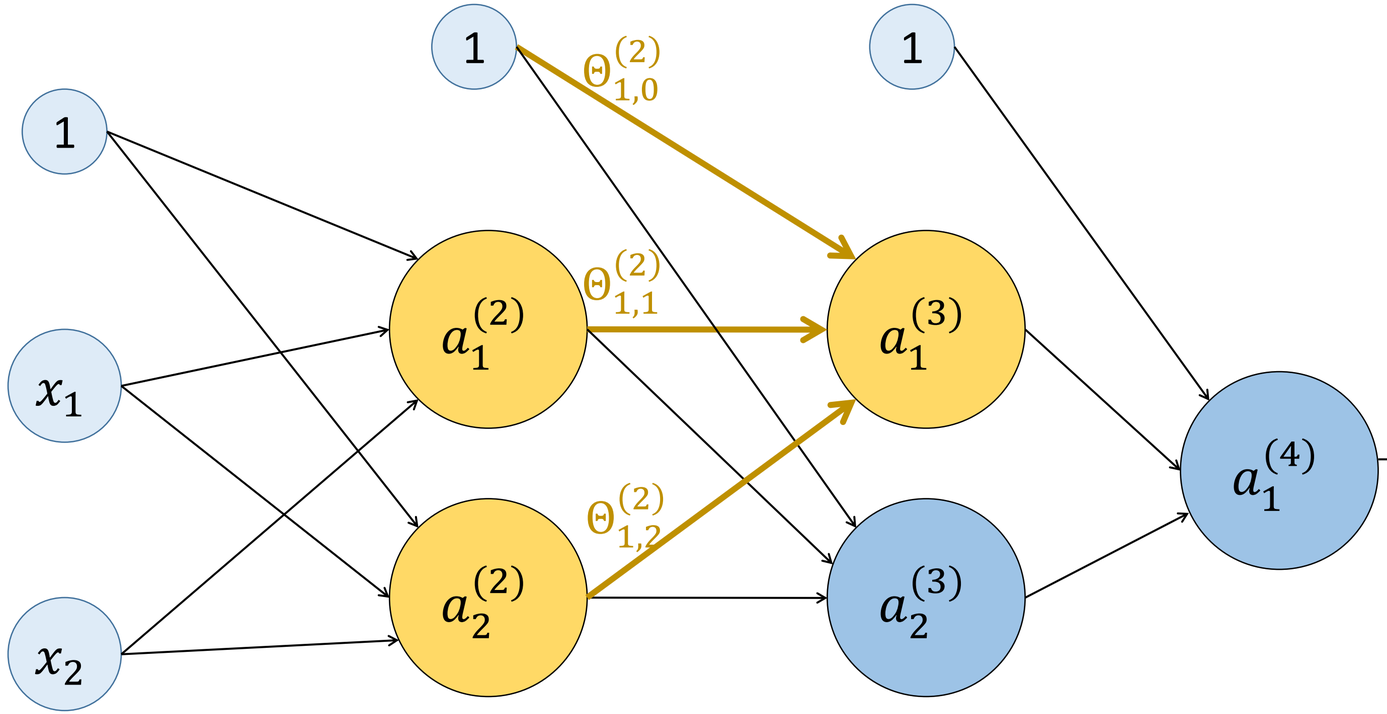
$$a_2^{(2)} = \varphi(\Theta_{2,0}^{(1)} + \Theta_{2,1}^{(1)} x_1 + \Theta_{2,2}^{(1)} x_2)$$

Loss Function
(e.g., sum of squares)

Training Set

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Forward propagation: Compute node outputs



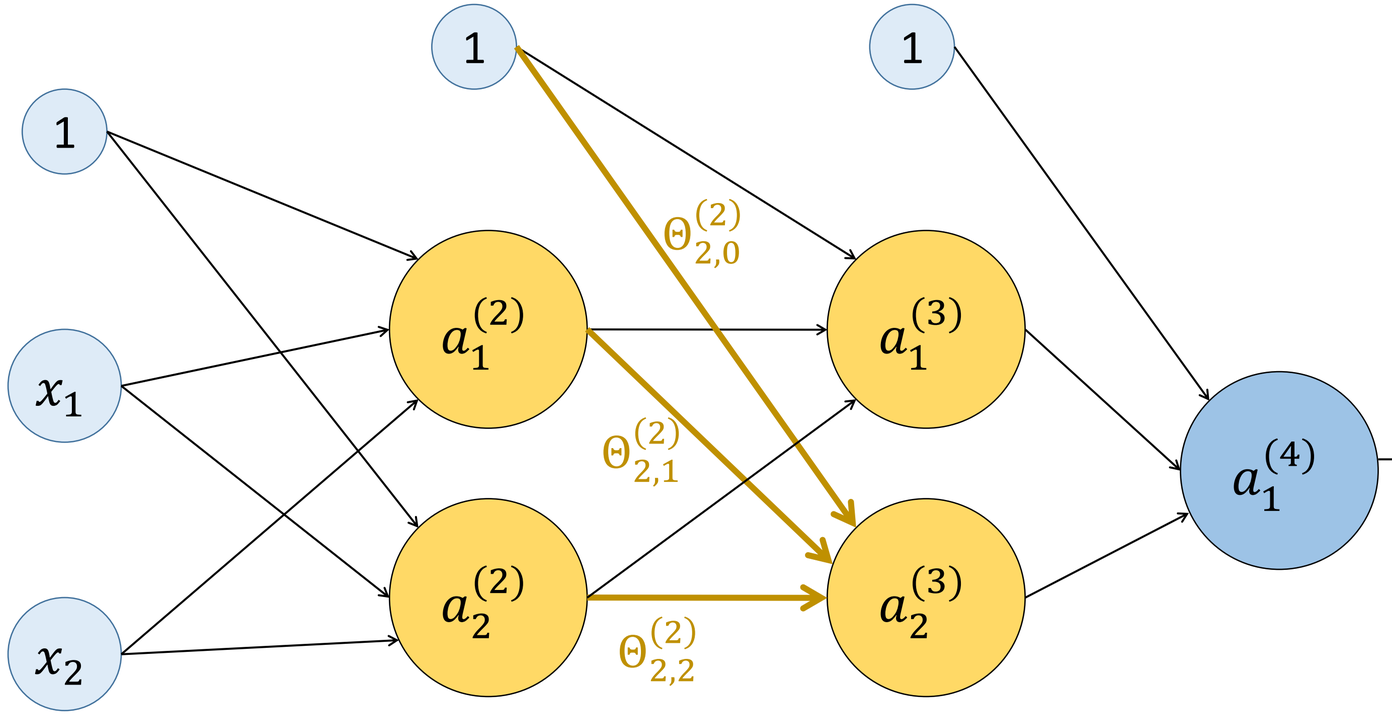
$$a_1^{(3)} = \varphi(\Theta_{1,0}^{(2)} + \Theta_{1,1}^{(2)} a_1^{(2)} + \Theta_{1,2}^{(2)} a_2^{(2)})$$

Loss Function
(e.g., sum of squares)

Training Set

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Forward propagation: Compute node outputs



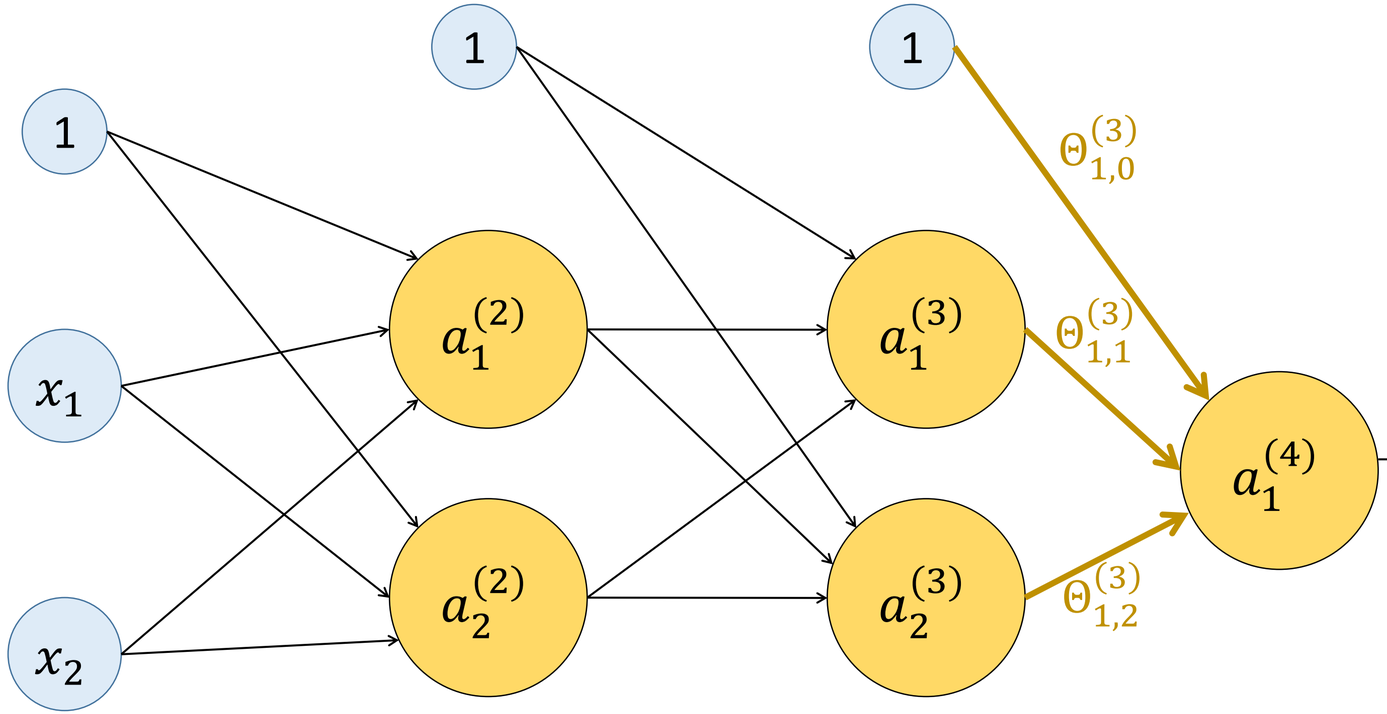
$$a_2^{(3)} = \varphi(\Theta_{2,0}^{(2)} + \Theta_{2,1}^{(2)} a_1^{(2)} + \Theta_{2,2}^{(2)} a_2^{(2)})$$

Loss Function
(e.g., sum of squares)

Training Set

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Forward propagation: Compute node outputs



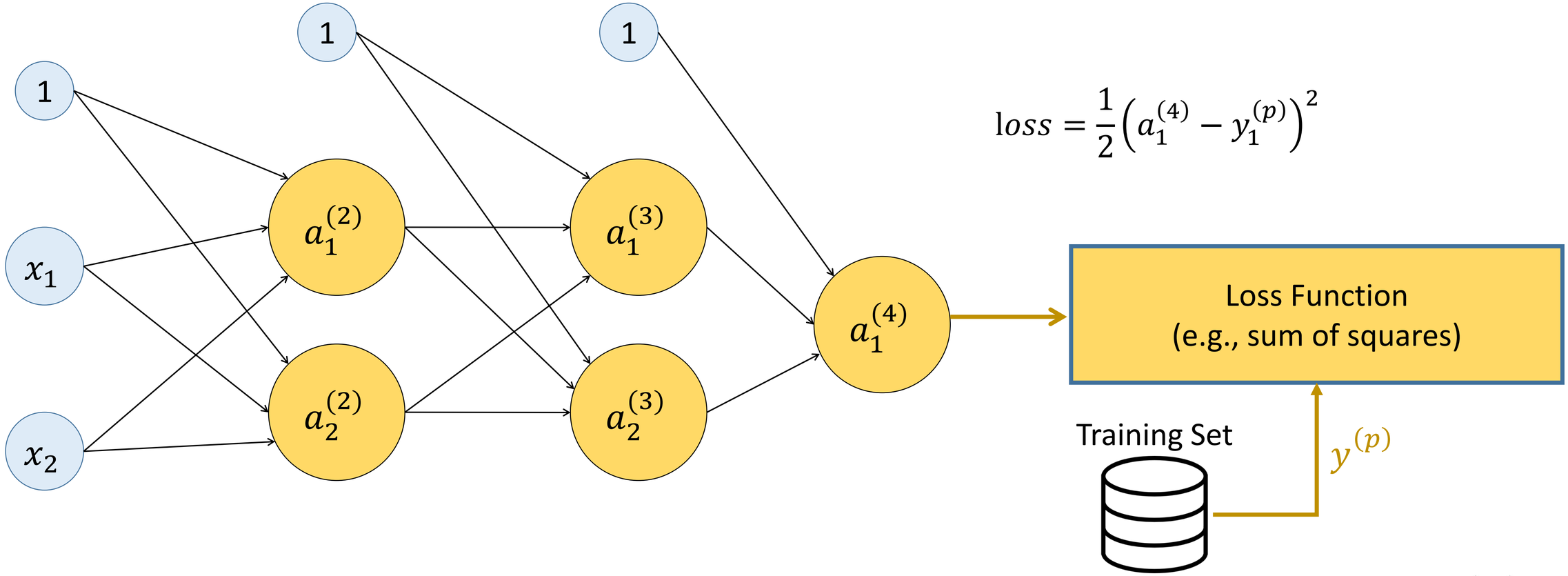
$$a_1^{(4)} = \varphi(\Theta_{1,0}^{(3)} + \Theta_{1,1}^{(3)} a_1^{(3)} + \Theta_{1,2}^{(3)} a_2^{(3)})$$

Loss Function
(e.g., sum of squares)

Training Set

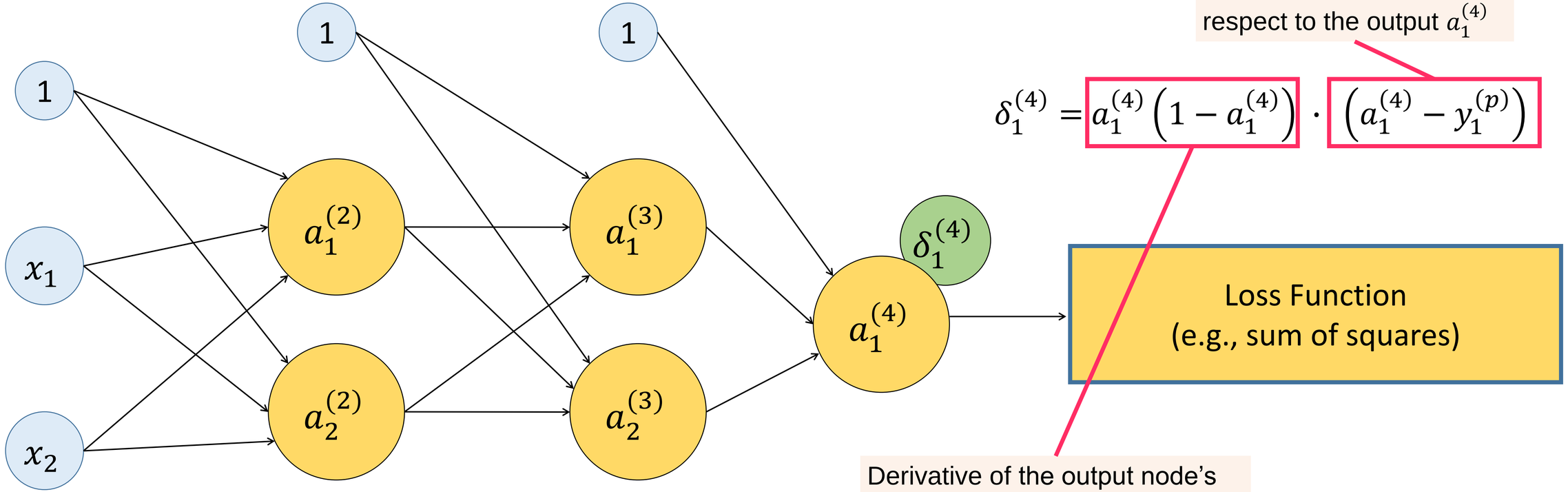
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Forward propagation: Compute the loss



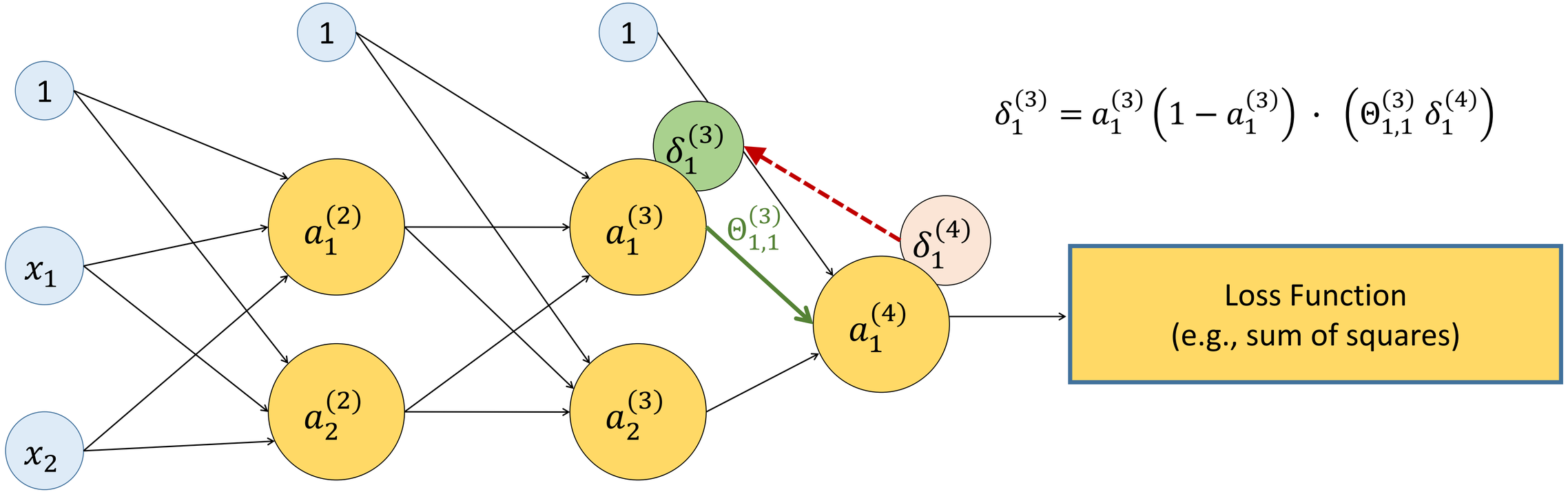
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Backpropagation: Compute partial derivatives



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

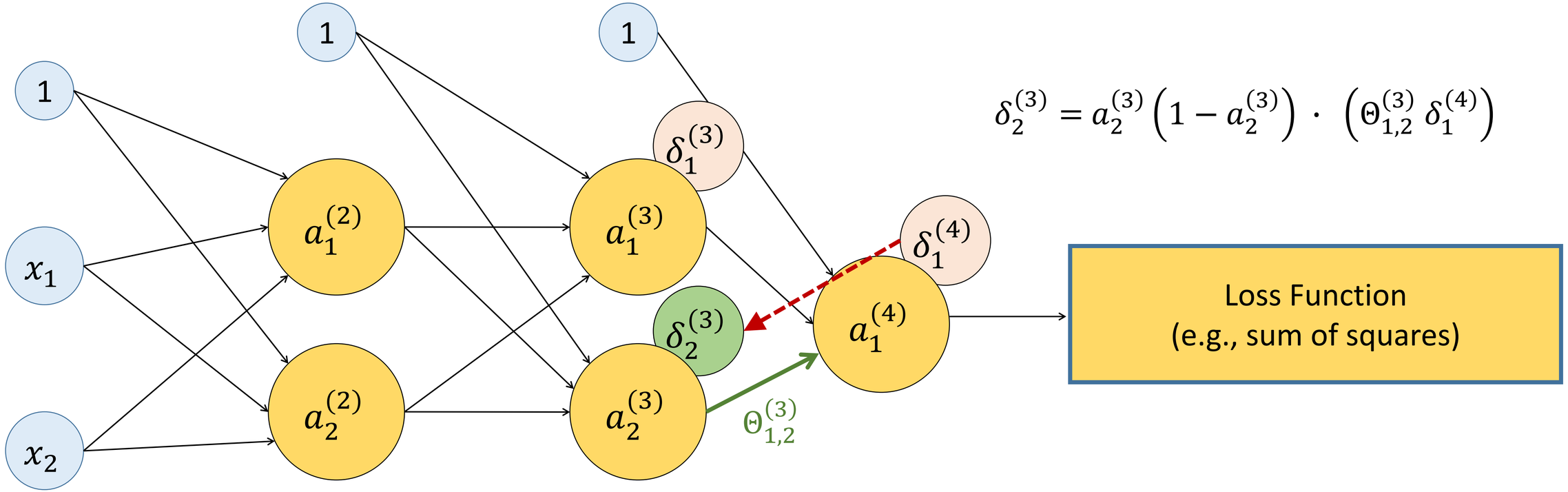
Backpropagation: Compute partial derivatives



$$\delta_1^{(3)} = a_1^{(3)}\left(1 - a_1^{(3)}\right) \cdot \left(\Theta_{1,1}^{(3)}\,\delta_1^{(4)}\right)$$

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

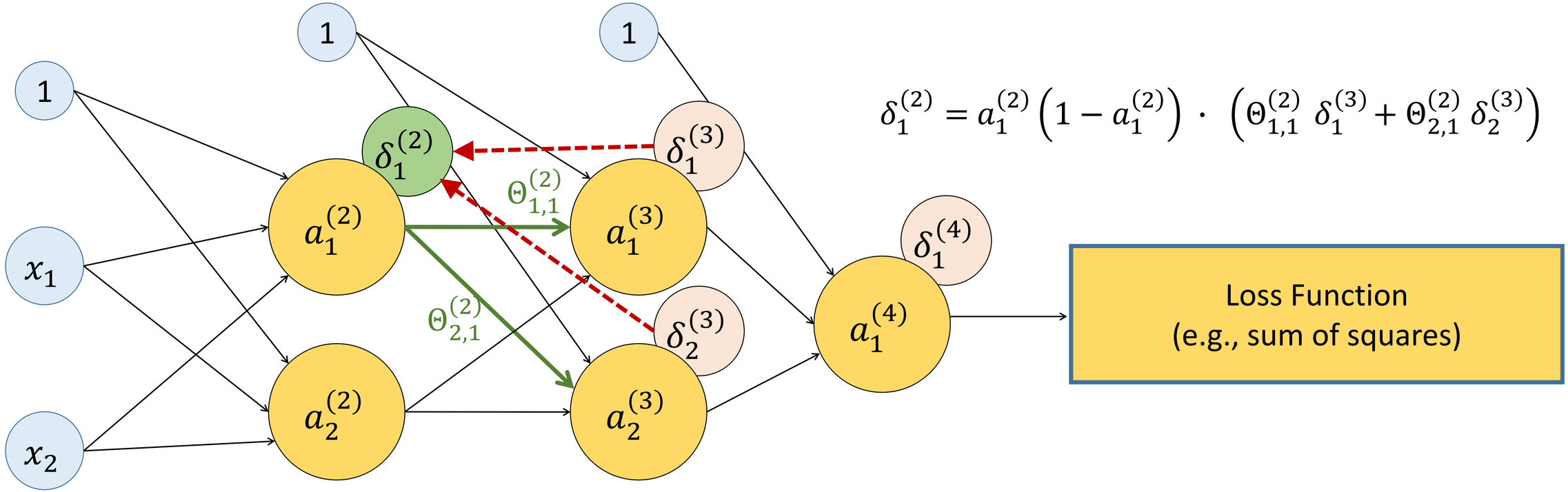
Backpropagation: Compute partial derivatives



$$\delta_2^{(3)} = a_2^{(3)}\left(1 - a_2^{(3)}\right) \cdot \left(\Theta_{1,2}^{(3)}\,\delta_1^{(4)}\right)$$

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

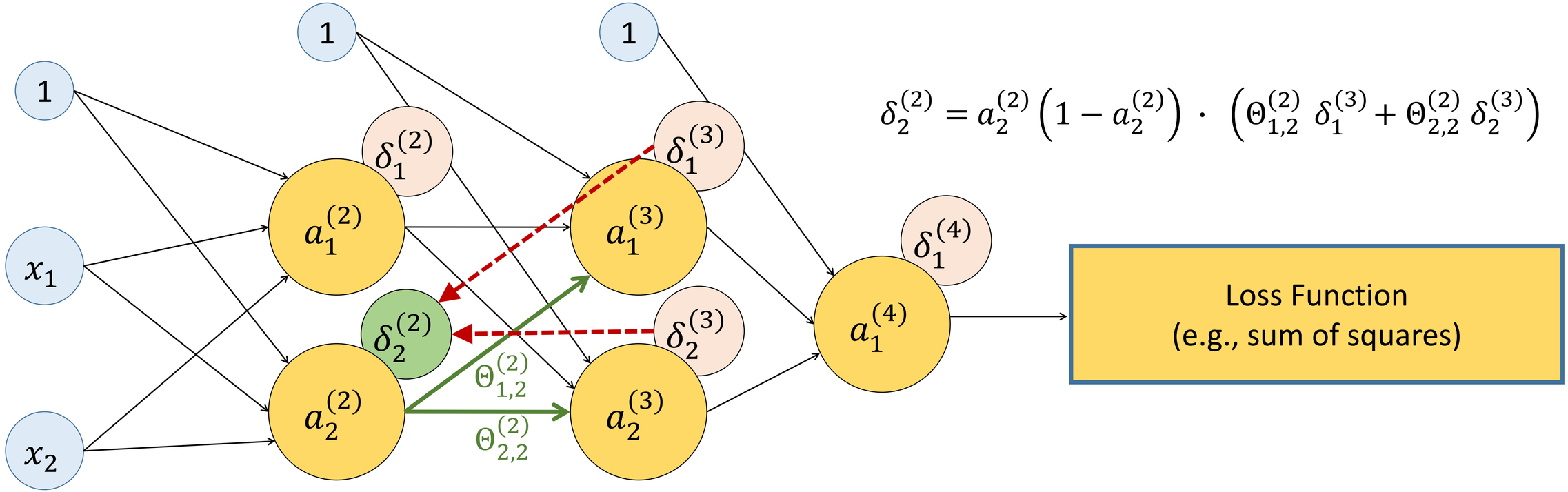
Backpropagation: Compute partial derivatives



$$\delta_1^{(2)} = a_1^{(2)}\left(1 - a_1^{(2)}\right) \cdot \left(\Theta_{1,1}^{(2)}\ \delta_1^{(3)} + \Theta_{2,1}^{(2)}\ \delta_2^{(3)}\right)$$

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

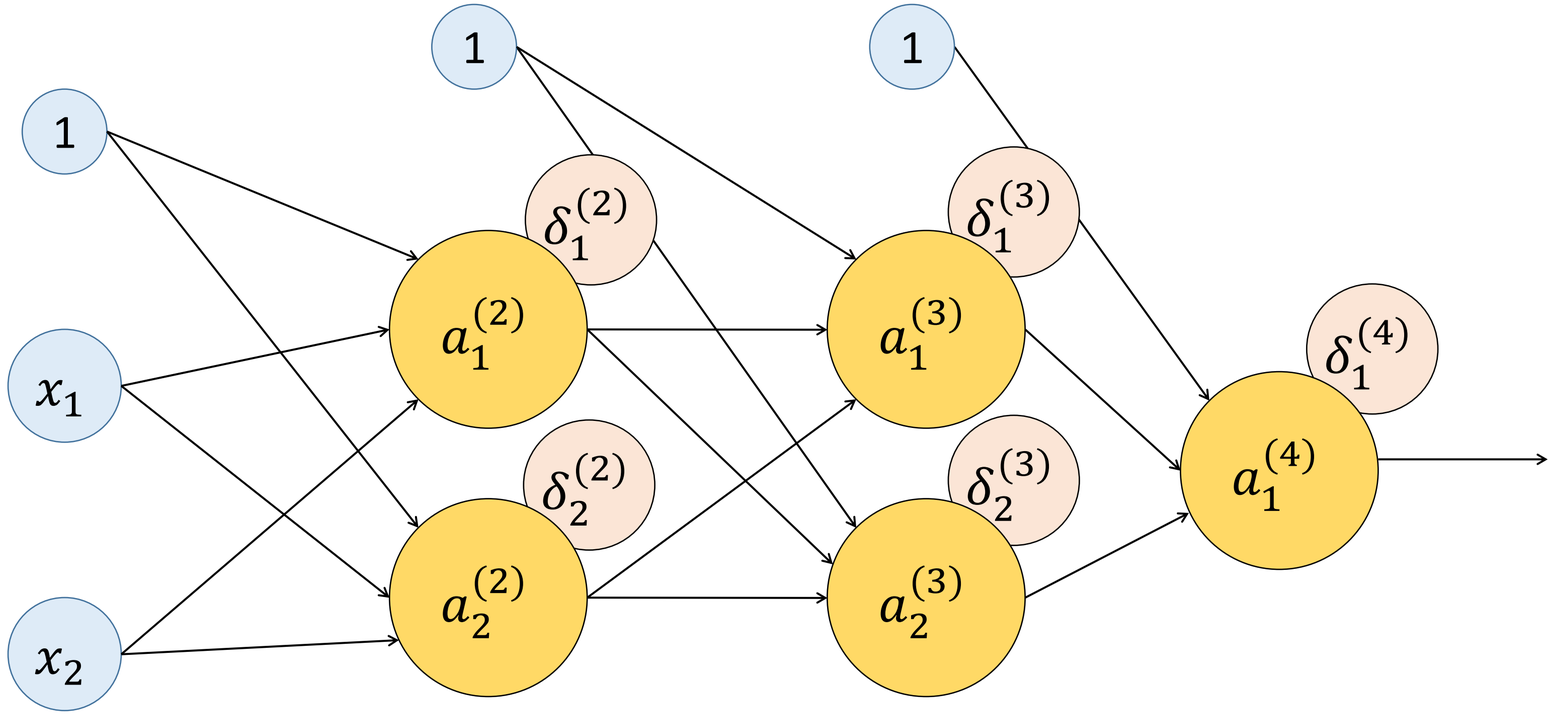
Backpropagation: Compute partial derivatives



$$\delta_2^{(2)} = a_2^{(2)}\left(1 - a_2^{(2)}\right) \cdot \left(\Theta_{1,2}^{(2)}\ \delta_1^{(3)} + \Theta_{2,2}^{(2)}\ \delta_2^{(3)}\right)$$

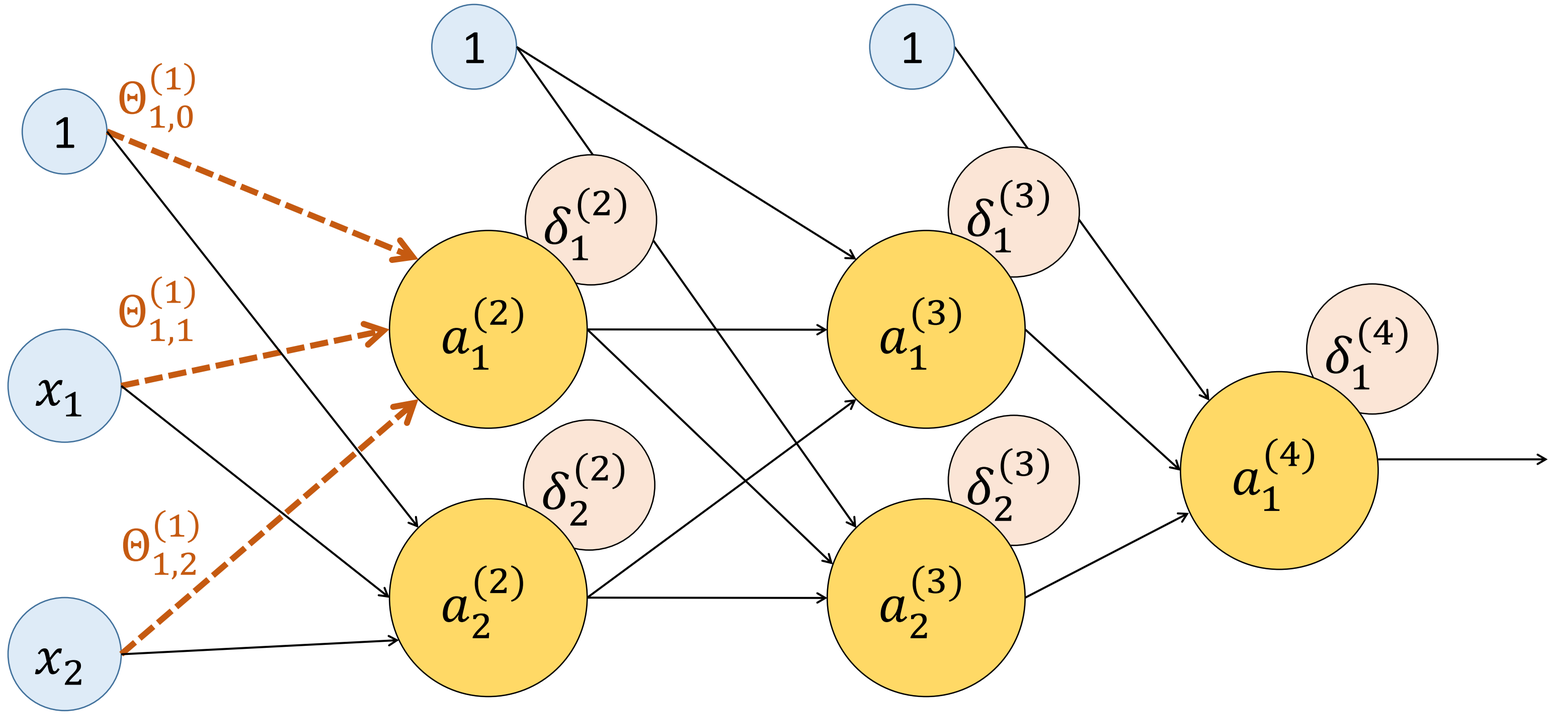
This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Update the parameters with stochastic gradient descent



This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Update the parameters with stochastic gradient descent



Derivative of the loss with respect to $\Theta_{1,0}^{(1)}$

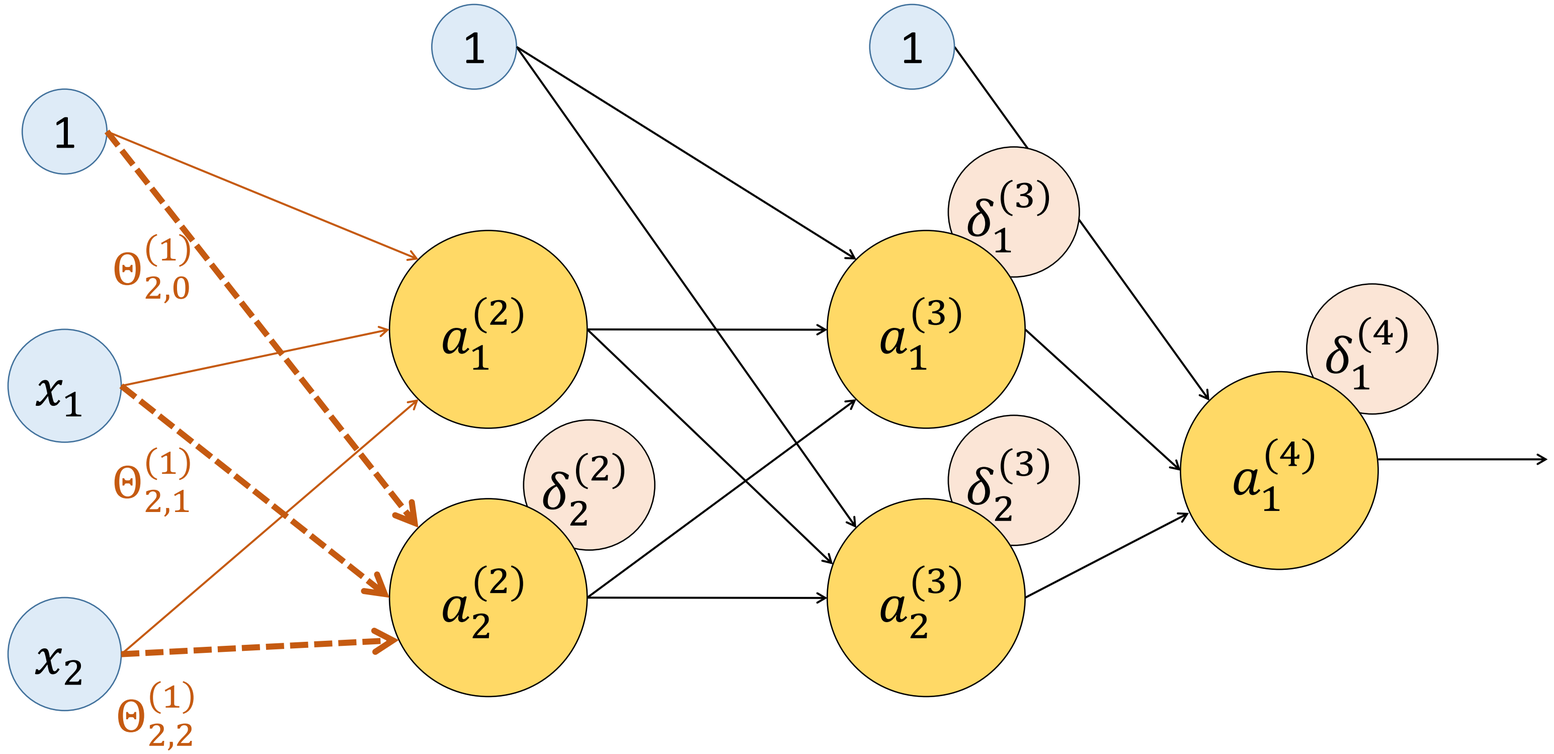
$$\Theta_{1,0}^{(1)} := \Theta_{1,0}^{(1)} - \eta \delta_1^{(2)} x_0$$

$$\Theta_{1,1}^{(1)} := \Theta_{1,1}^{(1)} - \eta \, \delta_1^{(2)} x_1$$

$$\Theta_{1,2}^{(1)} := \Theta_{1,2}^{(1)} - \eta \, \delta_1^{(2)} x_2$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Update the parameters with stochastic gradient descent



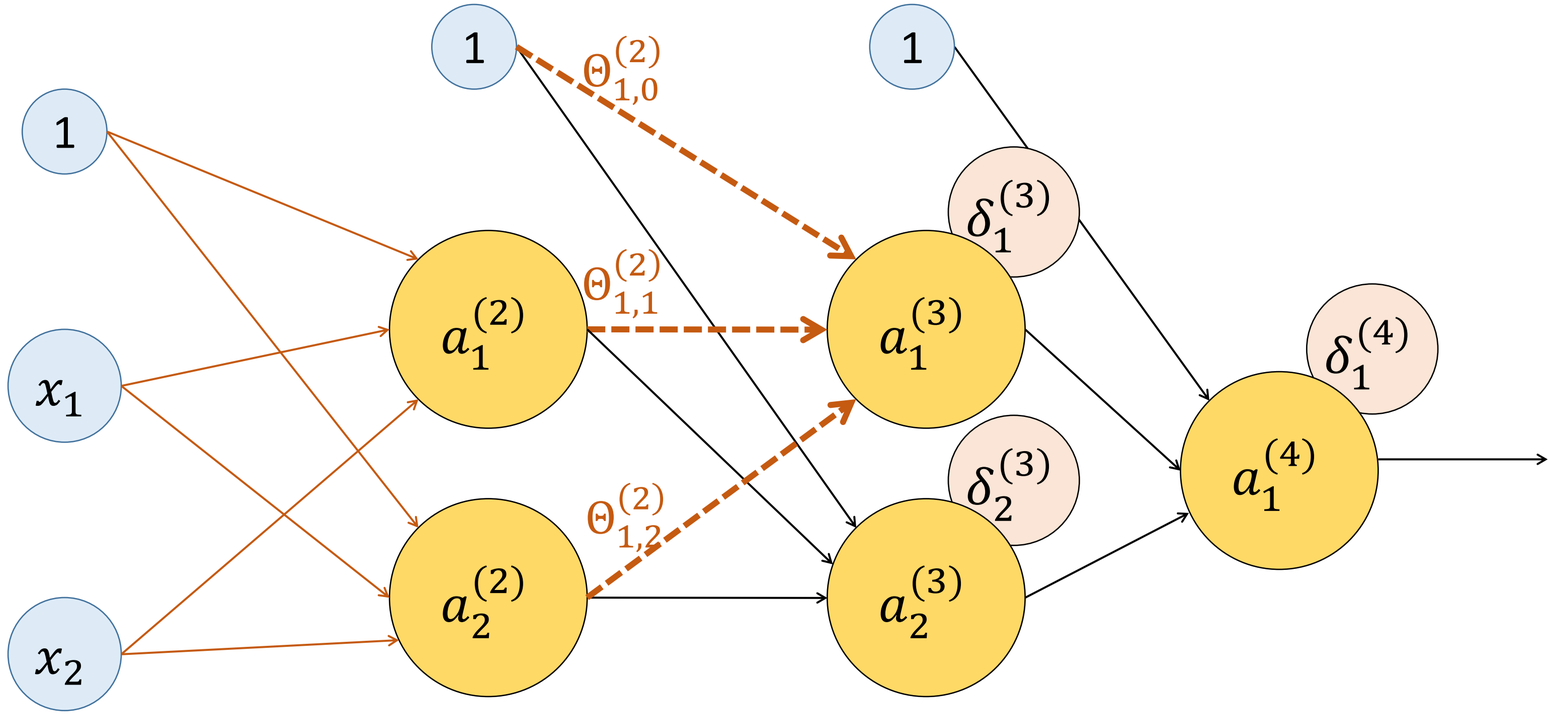
$$\Theta_{2,0}^{(1)} := \Theta_{2,0}^{(1)} - \eta\, \delta_2^{(2)} x_0$$

$$\Theta_{2,1}^{(1)} := \Theta_{2,1}^{(1)} - \eta\, \delta_2^{(2)} x_1$$

$$\Theta_{2,2}^{(1)} := \Theta_{2,2}^{(1)} - \eta\, \delta_2^{(2)} x_2$$

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Update the parameters with stochastic gradient descent



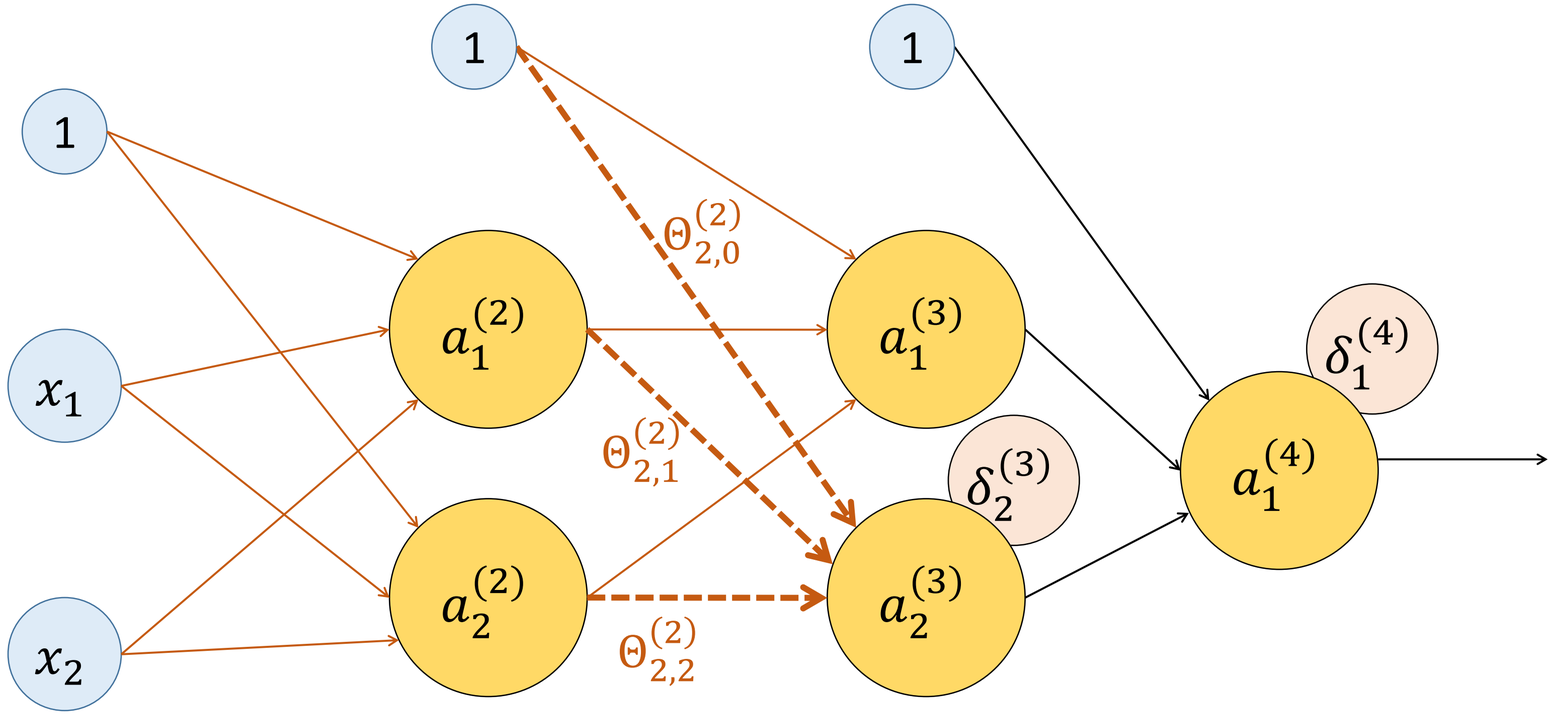
$$\Theta_{1,0}^{(2)} := \Theta_{1,0}^{(2)} - \eta\, \delta_1^{(3)} a_0^{(2)}$$

$$\Theta_{1,1}^{(2)} := \Theta_{1,1}^{(2)} - \eta\, \delta_1^{(3)} a_1^{(2)}$$

$$\Theta_{1,2}^{(2)} := \Theta_{1,2}^{(2)} - \eta\, \delta_1^{(3)} a_2^{(2)}$$

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Update the parameters with stochastic gradient descent



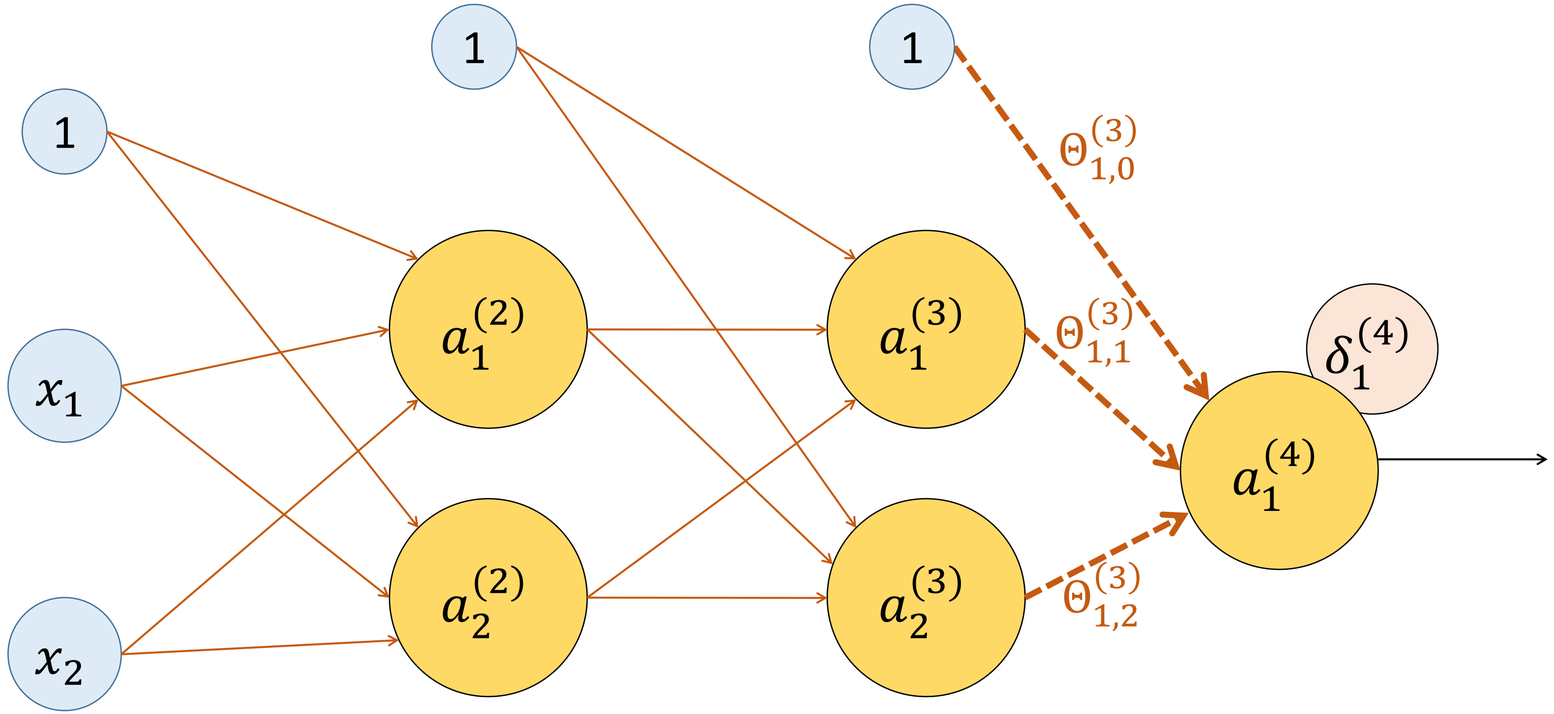
$$\Theta_{2,0}^{(2)} := \Theta_{2,0}^{(2)} - \eta\, \delta_2^{(3)} a_0^{(2)}$$

$$\Theta_{2,1}^{(2)} := \Theta_{2,1}^{(2)} - \eta\, \delta_2^{(3)} a_1^{(2)}$$

$$\Theta_{2,2}^{(2)} := \Theta_{2,2}^{(2)} - \eta\, \delta_2^{(3)} a_2^{(2)}$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Update the parameters with stochastic gradient descent



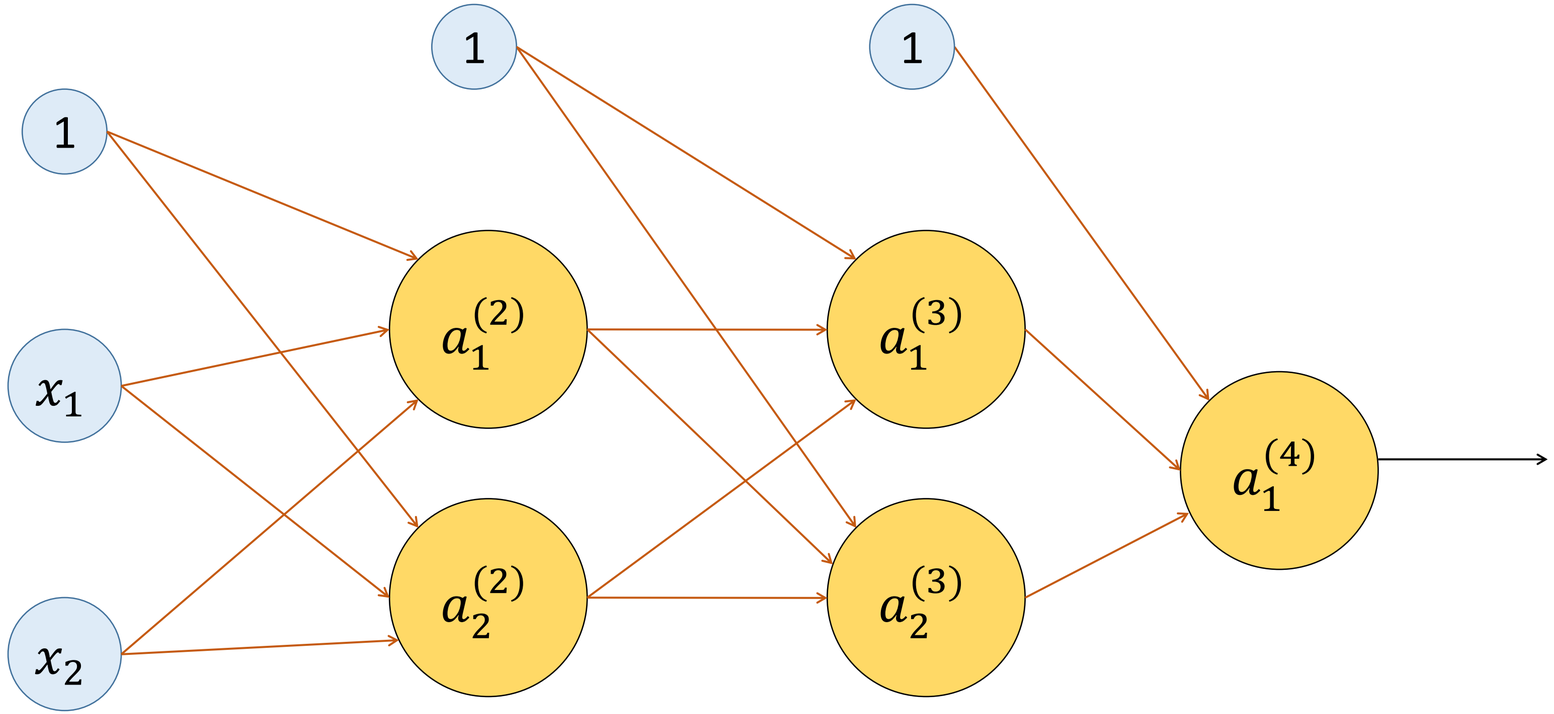
$$\Theta_{1,0}^{(3)} := \Theta_{1,0}^{(3)} - \eta\, \delta_1^{(4)} a_0^{(3)}$$

$$\Theta_{1,1}^{(3)} := \Theta_{1,1}^{(3)} - \eta\, \delta_1^{(4)} a_1^{(3)}$$

$$\Theta_{1,2}^{(3)} := \Theta_{1,2}^{(3)} - \eta\, \delta_1^{(4)} a_2^{(3)}$$

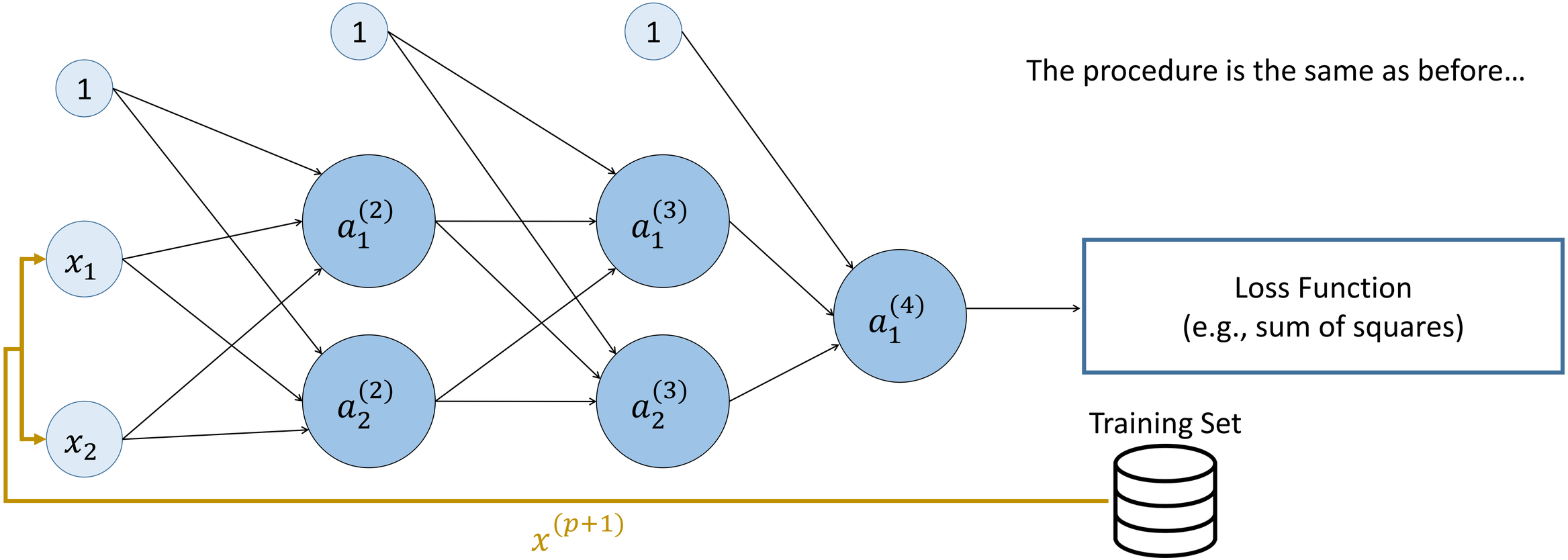
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

All parameters have been updated



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Pattern p+1 is presented



The procedure is the same as before…

Loss Function
(e.g., sum of squares)

Training Set

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Backpropagation algorithm

Weight update equation:

$$\Delta\Theta_{j,i}^{(k)} = -\eta \cdot \frac{\partial}{\partial\Theta_{j,i}^{(l)}} L(\Theta)$$

Online updating (stochastic gradient descent)

where:

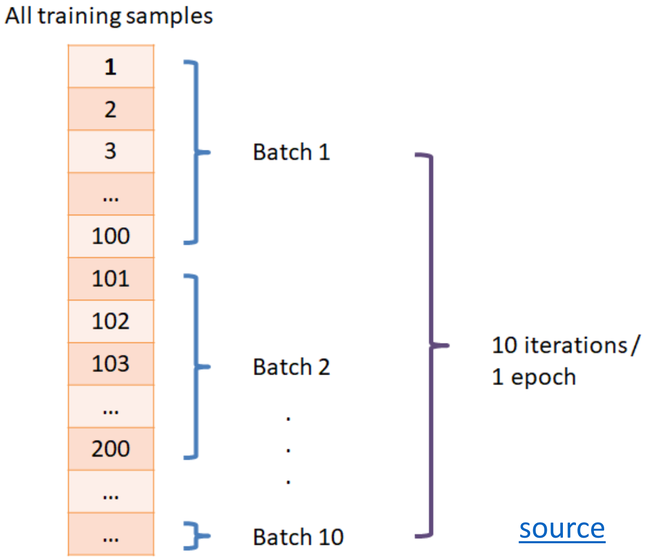
$$\frac{\partial}{\partial\Theta_{j,i}^{(l)}} L(\Theta) = \delta_j^{(k+1)} a_i^{(k)}$$

$$\frac{\partial}{\partial\Theta_{j,i}^{(l)}} L(\Theta) = \frac{1}{m}\sum_{p=1}^{m} \delta_j^{(k+1)}(p)\, a_i^{(k)}(p)$$

Batch updating (batch gradient descent)

Mini-batch sampling/updating:

- Instead of using the whole training set, use nonoverlapping subsets of it (mini-batches)
- Use the batch updating procedure where m is the size of the mini-batch



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Training feedforward NNs using backpropagation

For each epoch (or until total error or gradient norm is small enough)

Set $\Delta_{ij}^{(l)} = 0$ (for all l, i, j)

for $i = 1\ to\ m$

Set $a^{(1)} = x^{(i)}$

Forward propagation: Compute $a^{(l)}$ for $l = 2,3,..,L$

Backpropagation: Compute $\delta^{(L)} = a^{(L)} - y^{(i)}, \delta^{(L-1)}, \delta^{(L-2)}, \ldots, \delta^{(2)}$

$\Delta_{ij}^{(l)} := \Delta_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$

$D_{ij}^{(l)} := \frac{1}{m}\Delta_{ij}^{(l)} + \lambda\Theta_{ij}^{(l)}$ if $j \neq 0$

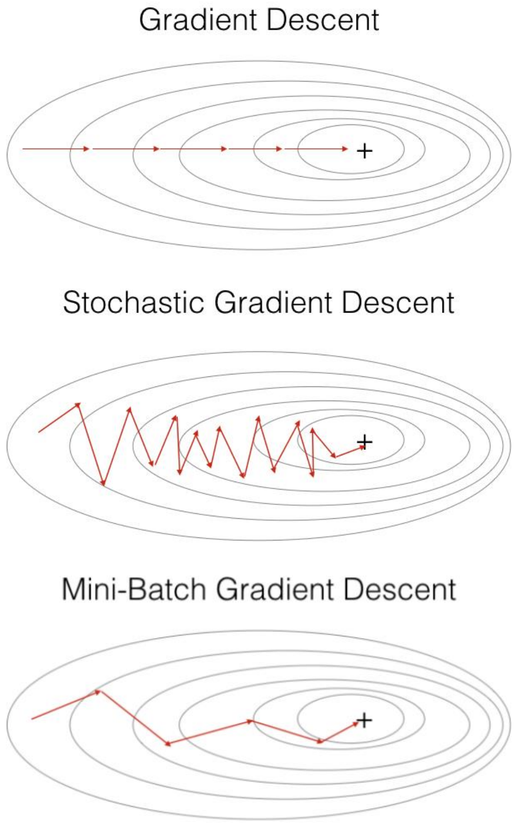
$D_{ij}^{(l)} := \frac{1}{m}\Delta_{ij}^{(l)}$ if $j = 0$

Epoch: 1 pass over all training data

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gradient descent

- Use batch GD if the training set can fit in memory
  - Smooth training curve
  - Slow (1 update requires the whole training set)
  - Can easily get stuck in local minima
- Use stochastic GD for big training sets:
  - Noisy training curve
  - Fast (1 update requires a single instance)
  - Still prone to getting stuck in local minima, but less than batch GD
- In practice:
  - Use mini-batch GD by setting the batch size depending on the problem (e.g., instance size) and the specs of the computer so that it can fit in memory
  - Batch size: $2^0, 2^1, 2^2, \ldots, m$



source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Efficiency of Backpropagation

Backpropagation:

- Needs $O(W)$ operations where W = number of parameters

Numerical differentiation:

- Finite differences method:

$$\frac{\partial L(\theta_0, \dots, \theta_i, \dots, \theta_k)}{\partial \theta_i} \approx \frac{L(\theta_0, \dots, \theta_i + \varepsilon, \dots, \theta_k) - L(\theta_0, \dots, \theta_i - \varepsilon, \dots, \theta_k)}{2\varepsilon} \qquad \varepsilon = 10^{-4}$$

- Need to perturb every parameter and calculate the loss: $O(W^2)$ operations

- Can be used to check if the implementation of backpropagation is correct

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

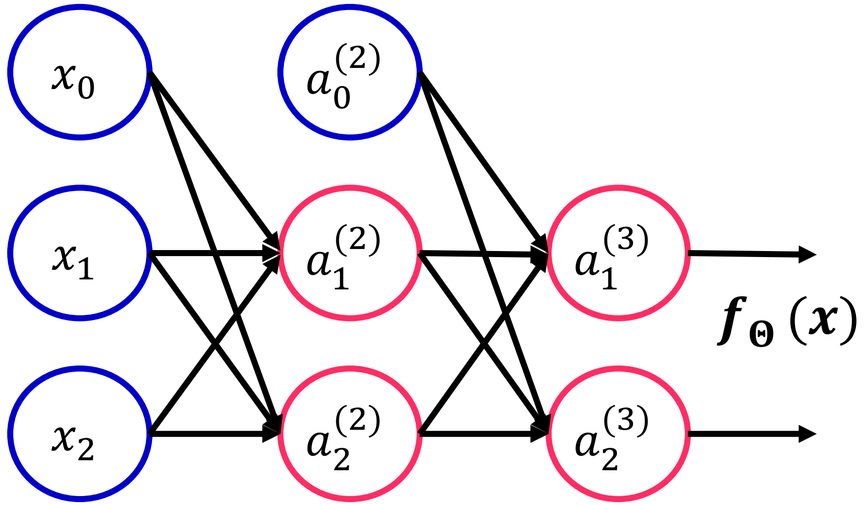
Vectorized Implementation

Forward propagation:

$$\boldsymbol{a^{(1)} = x}$$
$$\boldsymbol{z^{(2)} = \Theta^{(1)} a^{(1)}}$$
$$\boldsymbol{a^{(2)} = \varphi(z^{(2)})}$$

Add $a_0^{(2)} = 1$

$$\boldsymbol{z^{(3)} = \Theta^{(2)} a^{(2)}}$$
$$\boldsymbol{a^{(3)} = \varphi\left(z^{(3)}\right) = f_\Theta(x)}$$



This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Vectorized Implementation

Backpropagation:

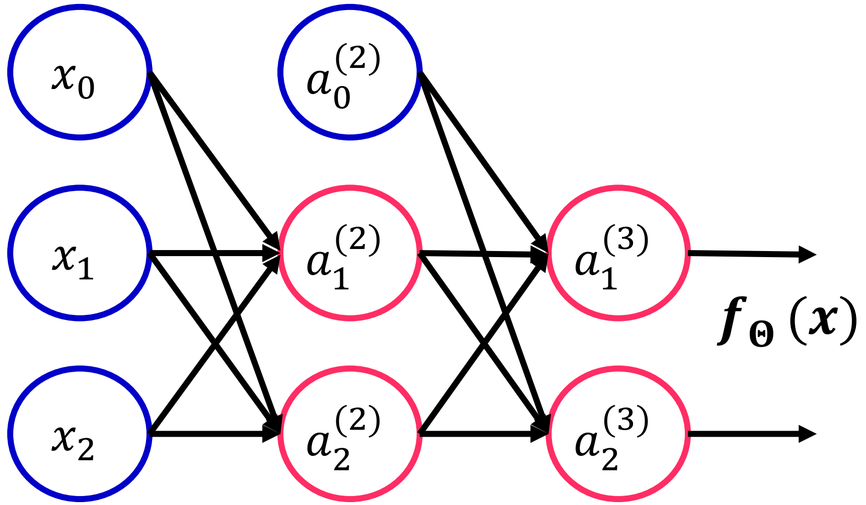
$$\boldsymbol{\delta}^{(3)} = \boldsymbol{a}^{(3)} - \boldsymbol{y}$$

$$\boldsymbol{\delta}^{(2)} = \left(\Theta^{(2)}\right)^{T} \boldsymbol{\delta}^{(3)} .* \boldsymbol{\varphi}'(\boldsymbol{z}^{(2)})$$

$.*$: element-wise multiplication

$$\boldsymbol{\varphi}'\left(\boldsymbol{z}^{(2)}\right) = \boldsymbol{a}^{(2)} .* \left(1 - \boldsymbol{a}^{(2)}\right)$$

No $\boldsymbol{\delta}^{(1)}$



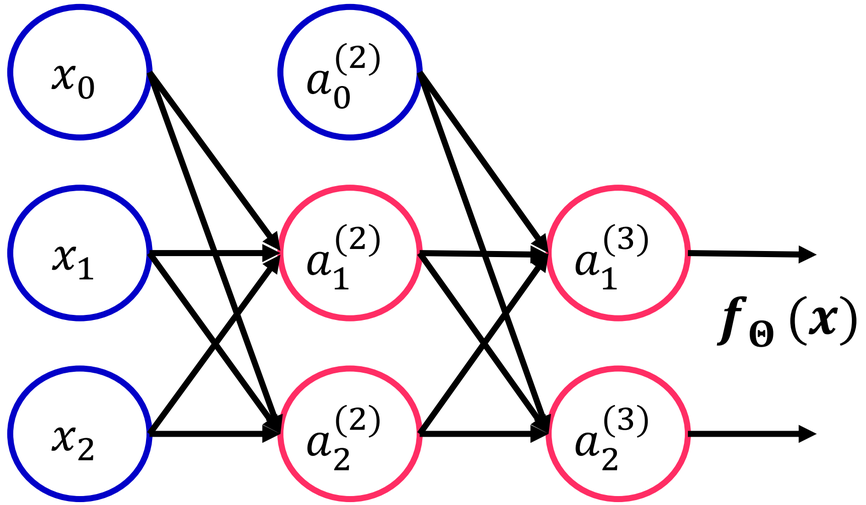
$$\frac{\partial}{\partial \Theta_{j,i}^{(l)}} L(\Theta) = a_j^{(l)} \delta_i^{(l+1)}$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Random Initialization

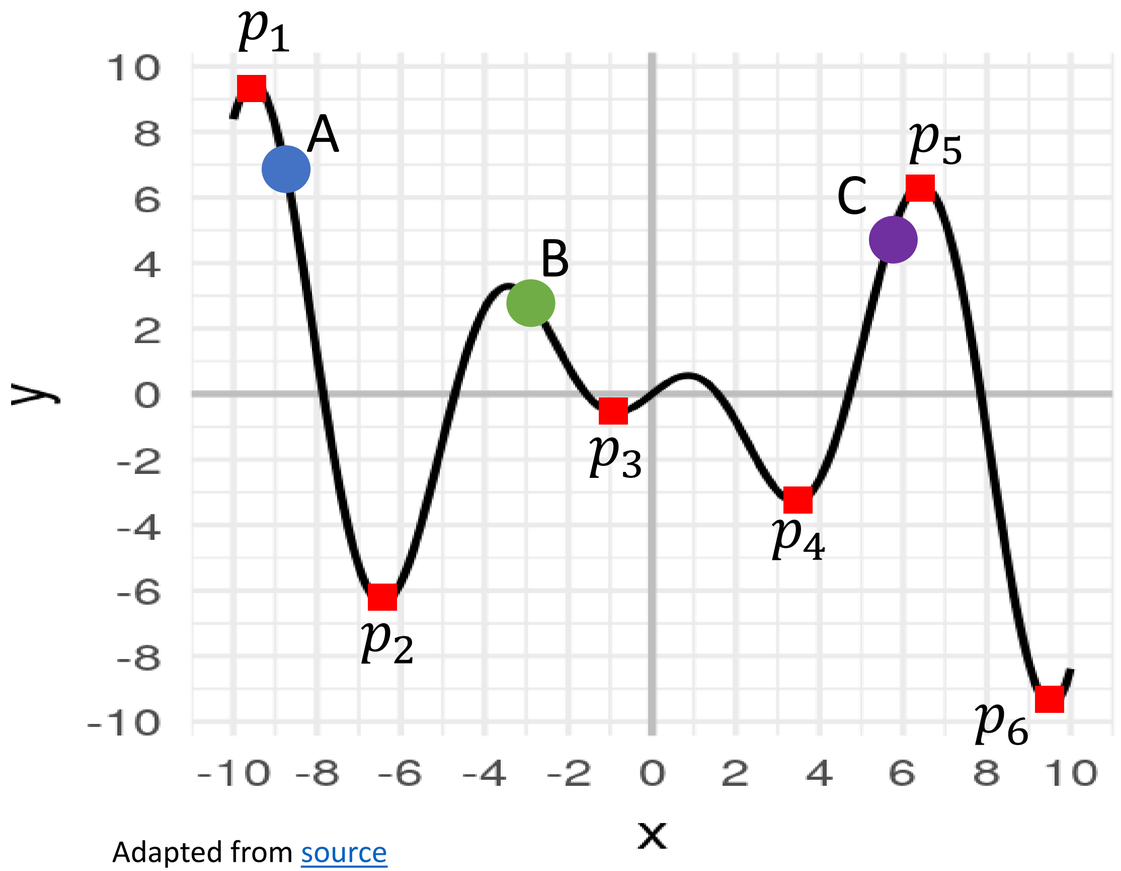
How to initialize $\Theta_{j,i}^{(l)}$?

- Zero initialization : $\Theta_{j,i}^{(l)} = 0$ for all i, j, l
  - After each update, weights that have the same source neuron are identical
  - All hidden units compute the same function
  - Network does not learn well
- Random initialization : $\Theta_{j,i}^{(l)} \sim U(-\varepsilon, \varepsilon)$
  - E.g. $\varepsilon = 1$
  - Each node computes a different function
  - Network learns more interesting functions



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Quiz



Adapted from source

Where will gradient descent end up if it uses a small learning rate and starting from:

- A:

  p2
- B:

  p3
- C:

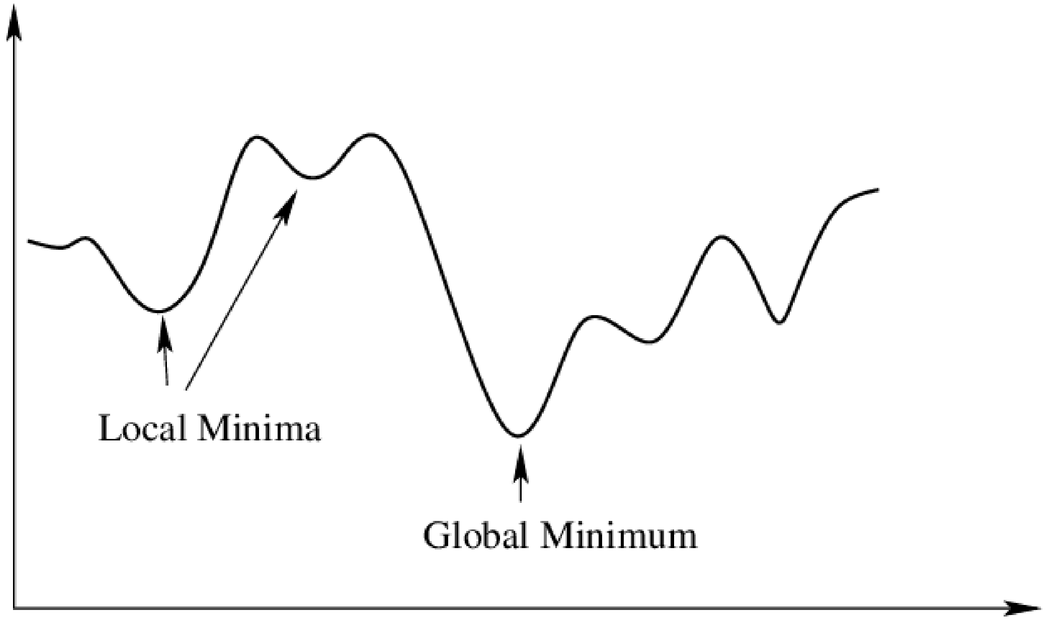
  p4

p6 will never be reached

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Problems with SGD

- Getting stuck in local minima
- Slow convergence or divergence



source

Let’s revise the approach…

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Stochastic gradient descent with momentum

Gradient Descent

learning rate

$$\theta(t+1) = \theta(t) - \eta \nabla \mathrm{L}(\theta(t))$$

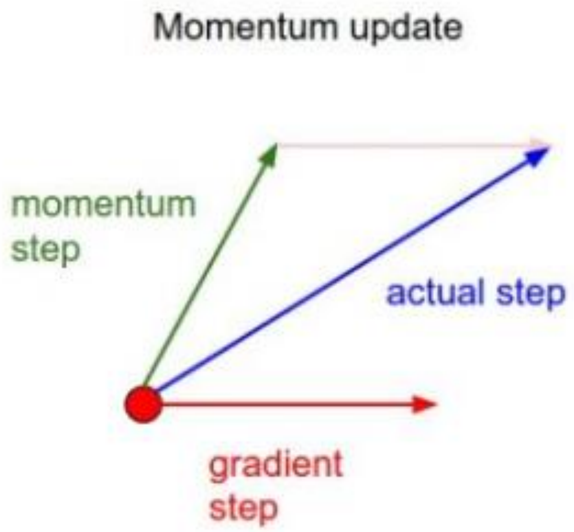
Gradient descent with momentum

momentum factor

$$z(t+1) = \mu z(t) + \nabla \mathrm{L}(\theta(t))$$
$$\theta(t+1) = \theta(t) - \eta\, z(t+1)$$

$$z(0) = 0$$

If $\mu = 0$ we get SGD



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

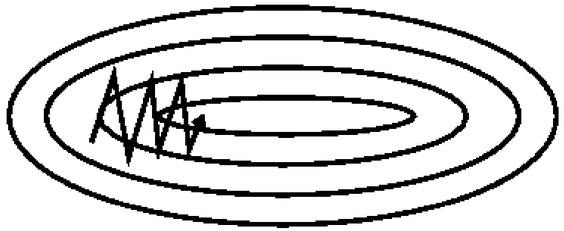
Stochastic gradient descent with momentum

- Momentum term: type of memory of the previous direction we were following

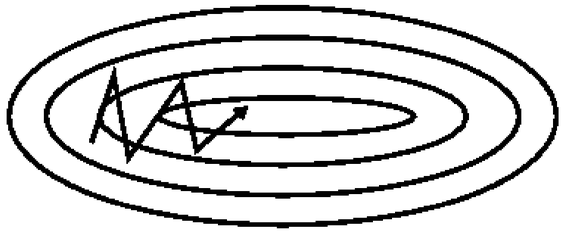
- Can speed up learning
  - by dampening oscillations
  - by increasing the effective learning rate in low curvature regions

- Can escape local optima

- Typical value: $\mu = 0.9$



SGD

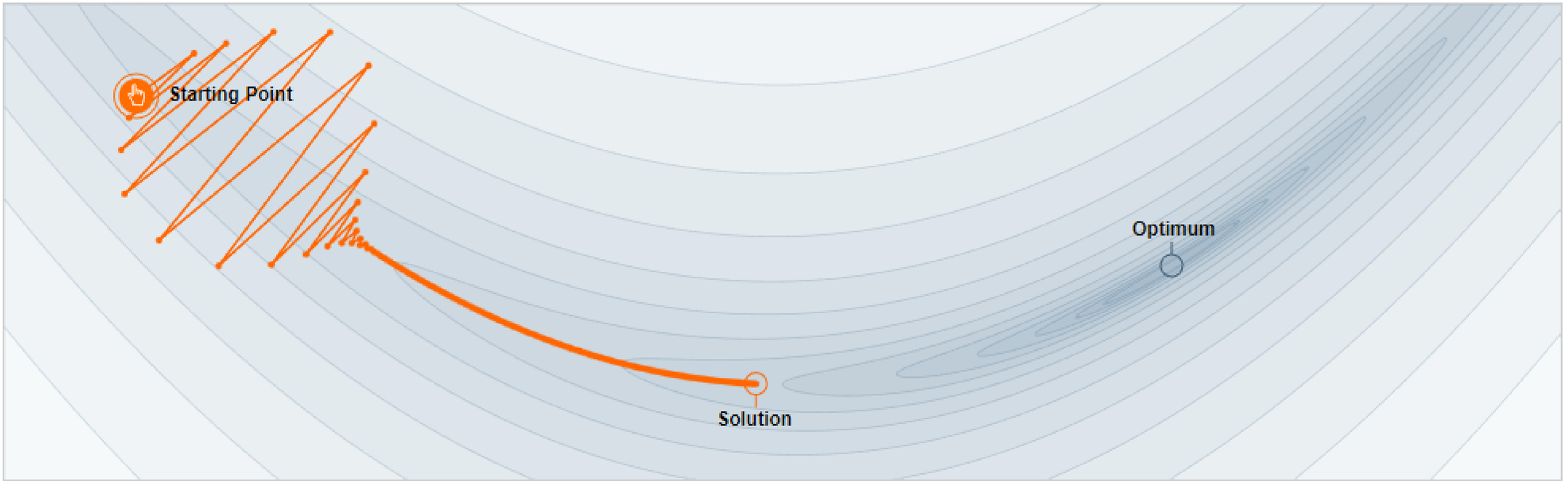


SGD with momentum

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

SQD with Momentum

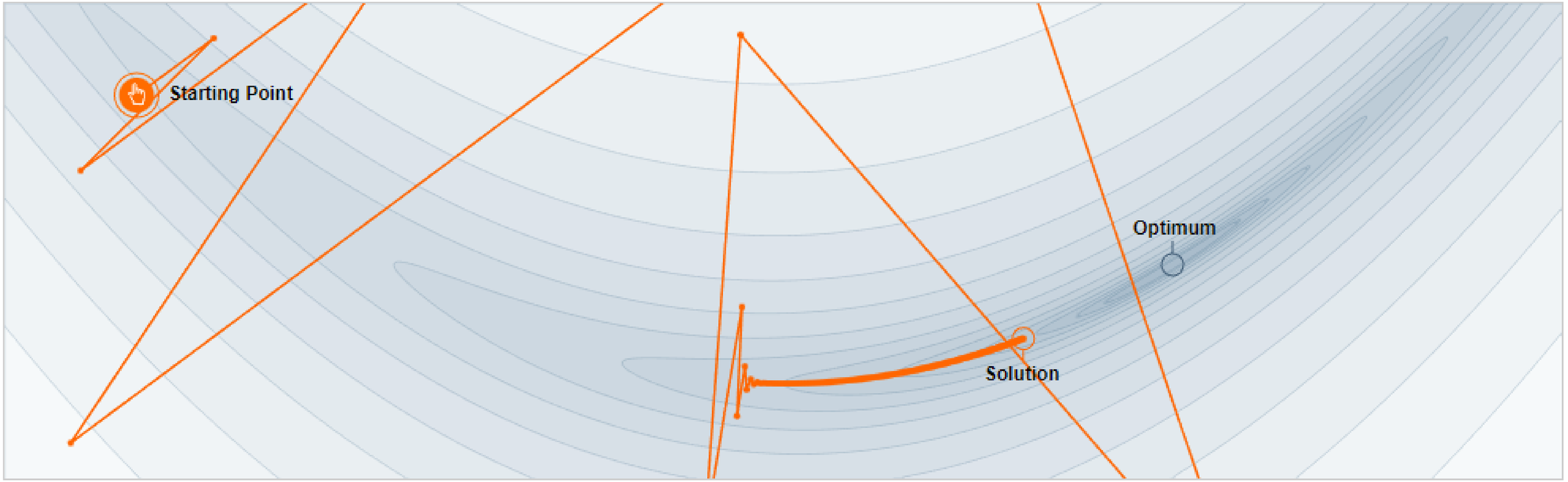


$$\eta = 0.003$$
$$\mu = 0.0$$

source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

SQD with Momentum

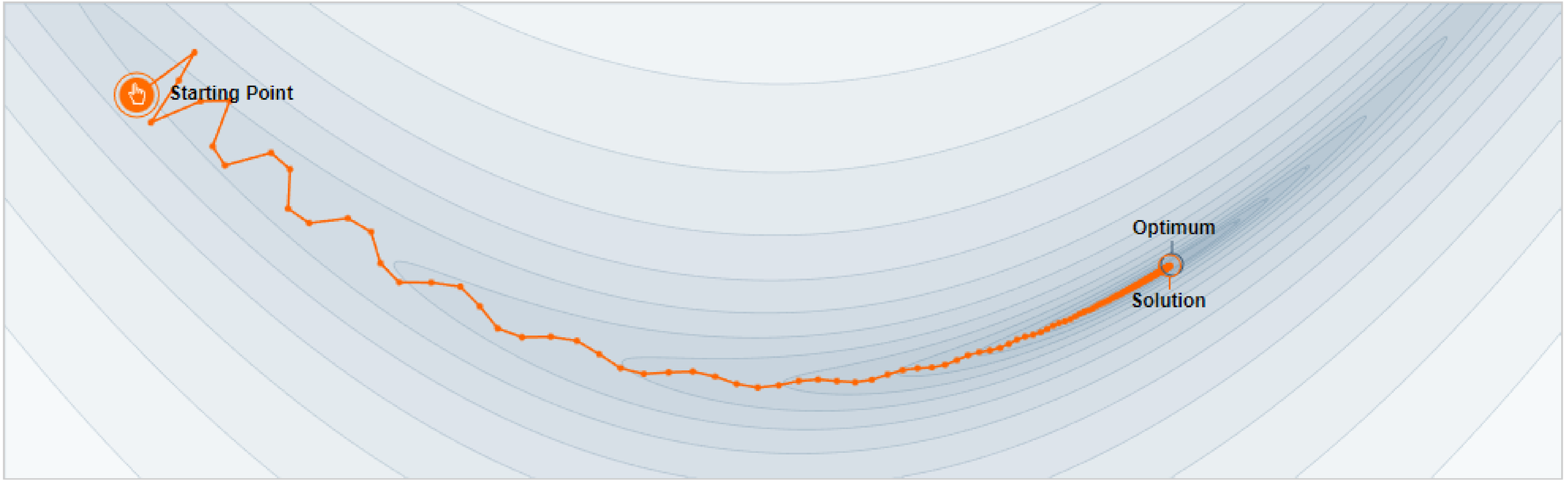


$$\eta = 0.004$$
$$\mu = 0.0$$

source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

SQD with Momentum

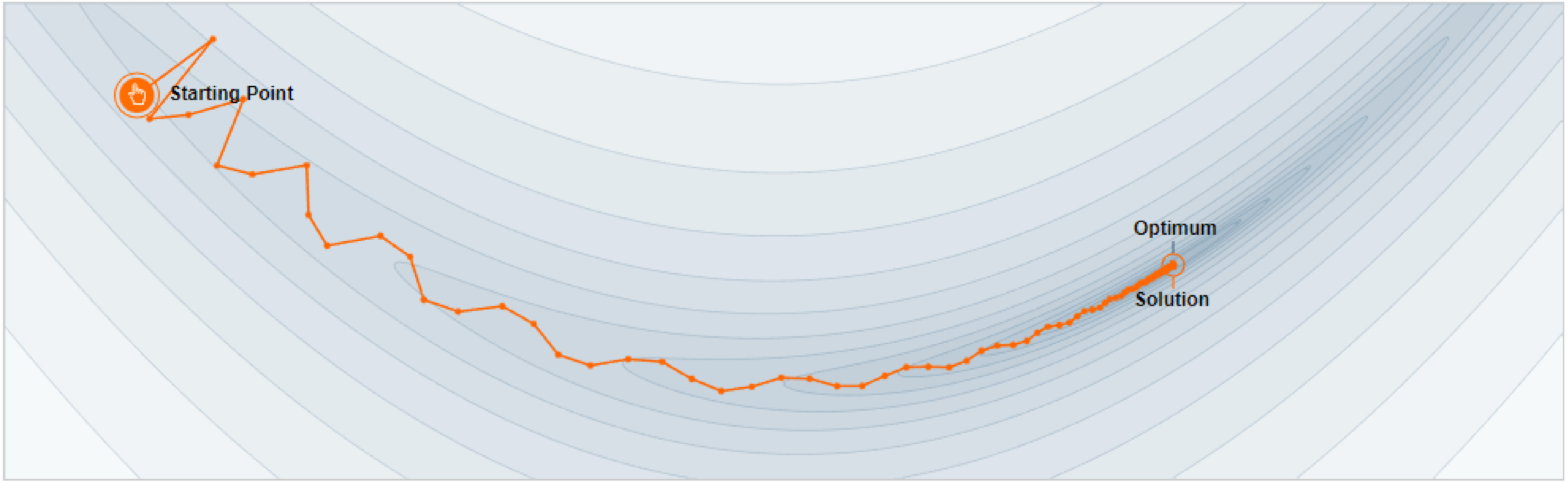


$$\eta = 0.003$$
$$\mu = 0.85$$

source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

SQD with Momentum

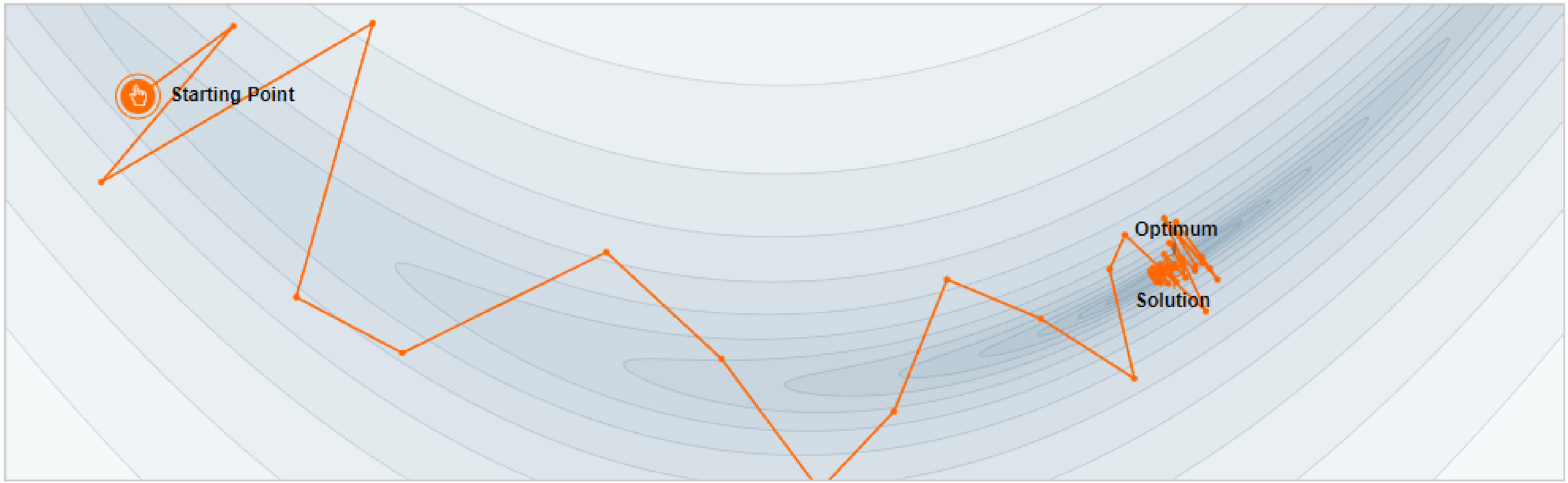


$$\eta = 0.004$$
$$\mu = 0.85$$

source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

SQD with Momentum



$$\eta = 0.005$$
$$\mu = 0.85$$

source

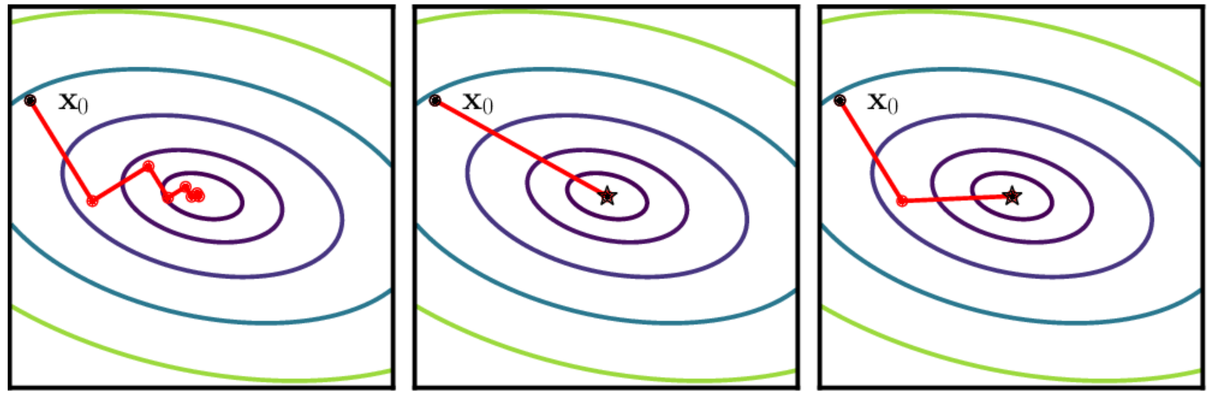
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Advanced optimization methods

- Gradient descent is a first-order optimization method

- Second-order methods use curvature information (2nd derivative) to accelerate convergence to the optimum
  - Newton's method
  - Conjugate gradient method
  - Levenberg–Marquardt algorithm
  - …

source



(a) Steepest descent (b) Newton's method (c) Conjugate gradient

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Advanced optimization methods

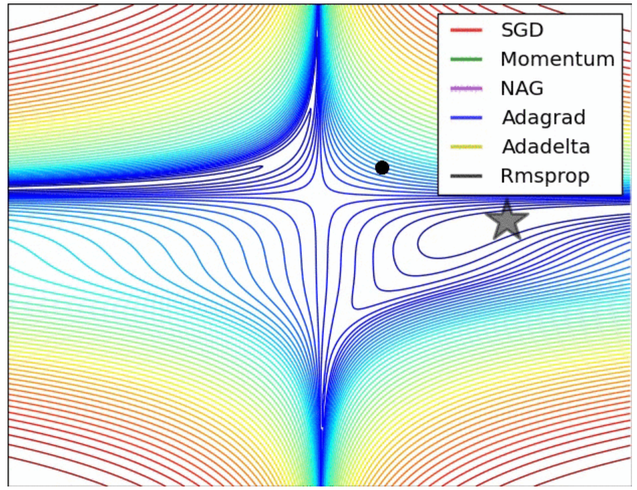
- Second order methods
  - More complex and harder to implement and tune
  - More expensive in terms of iteration cost and memory
  - They consider access to the true gradient
    - Harder to make them work with mini-batch updating

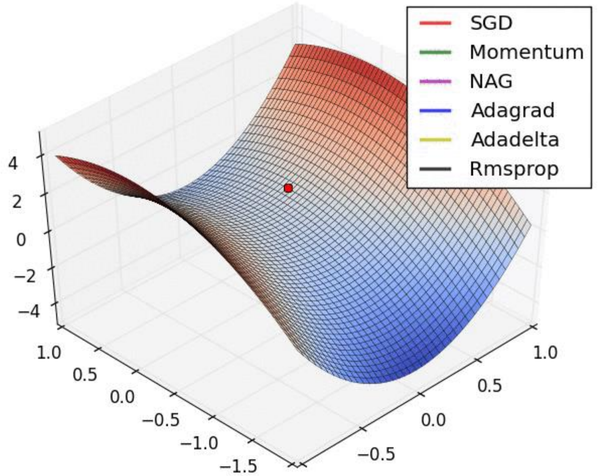
- Modern neural network training is based on variants of SGD

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Advanced optimization methods: SGD variants

- Vanilla SQD
- SQD with momentum
- Nesterov's accelerated gradient
- Adagrad
- Adadelta
- RMSProp
- Adam
- AdaMax
- Nadam
- …

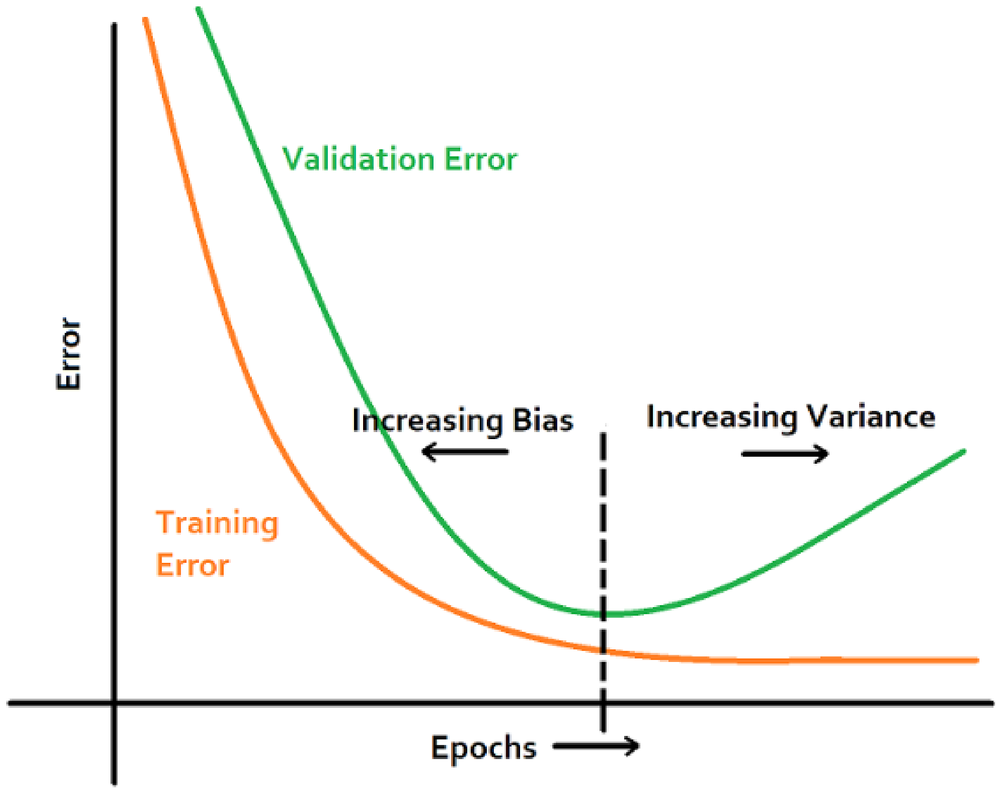




source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Early stopping



Source

- NNs are models with large learning capacity: prone to overfitting

Early stopping as regularization:

- While training, evaluate the performance on the validation set
- Save the network weights if the validation error is decreasing
- When the validation error starts increasing (consistently) stop and return the network with the saved weights

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Hyperparameter tuning

Model hyperparameters:

- Number of hidden layers
- Number of units per layer
- Activation functions per layer (per neuron?)
- Activation function parameters

Learning algorithm hyperparameters:

- Learning rate
- Momentum factor
- Learning rate decay
- Initialization parameters
- Regularization
- …

Model selection is used as before evaluating combinations of hyperparameters using the (cross-)validation set.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Ensembles

- NNs have many parameters: the error landscape is high-dimensional with many optima
- Gradient descent is a noisy process that is not guaranteed to converge to the same local optimum for every run
- We can train multiple independent networks and average their performance
- We can use bagging or boosting or more advanced methods (e.g., dropout) and obtain lower validation errors

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Automatic differentiation

- For different cost functions or activation functions we need to write explicit code that computes their derivative
- For a more flexible neural network architecture, we need to write explicit code for forward and backward propagation

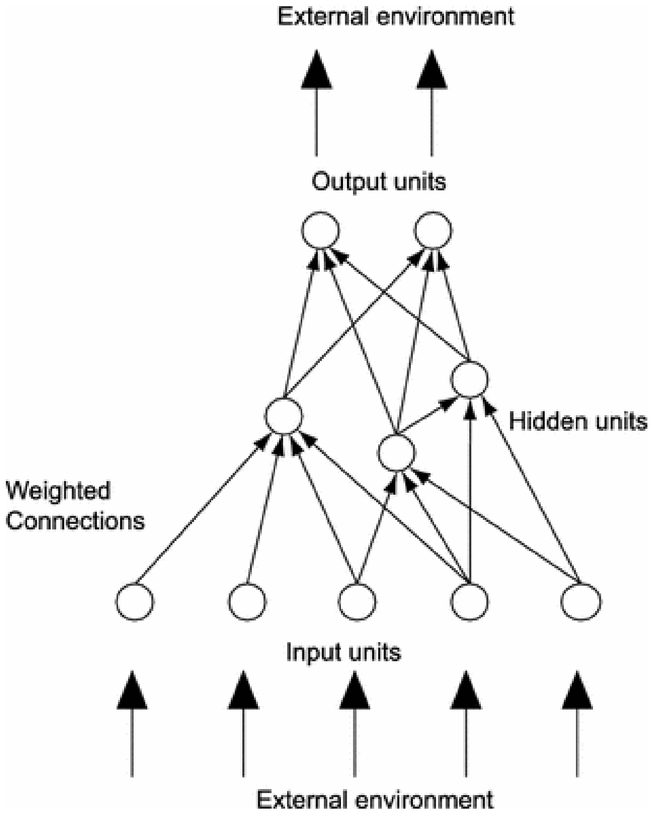
Automatic differentiation:

- A generalized way of using the chain rule with various operations and functions
- Essentially: backpropagation implemented for us
- Libraries: TensorFlow, PyTorch, JAX

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Learning the structure of the network

- Neural networks are function compositions
- Composing these functions in a layered feedforward manner works (universal approximation theorem), however, there more efficient ways of composing them for a given problem.
- Learning the connectivity cannot be (easily) formulated as a gradient-based optimization objective.
  - Need gradient-free methods
  - Grid search or random search can be inefficient
- **Neuroevolution**: using evolutionary algorithms to design/optimize neural networks



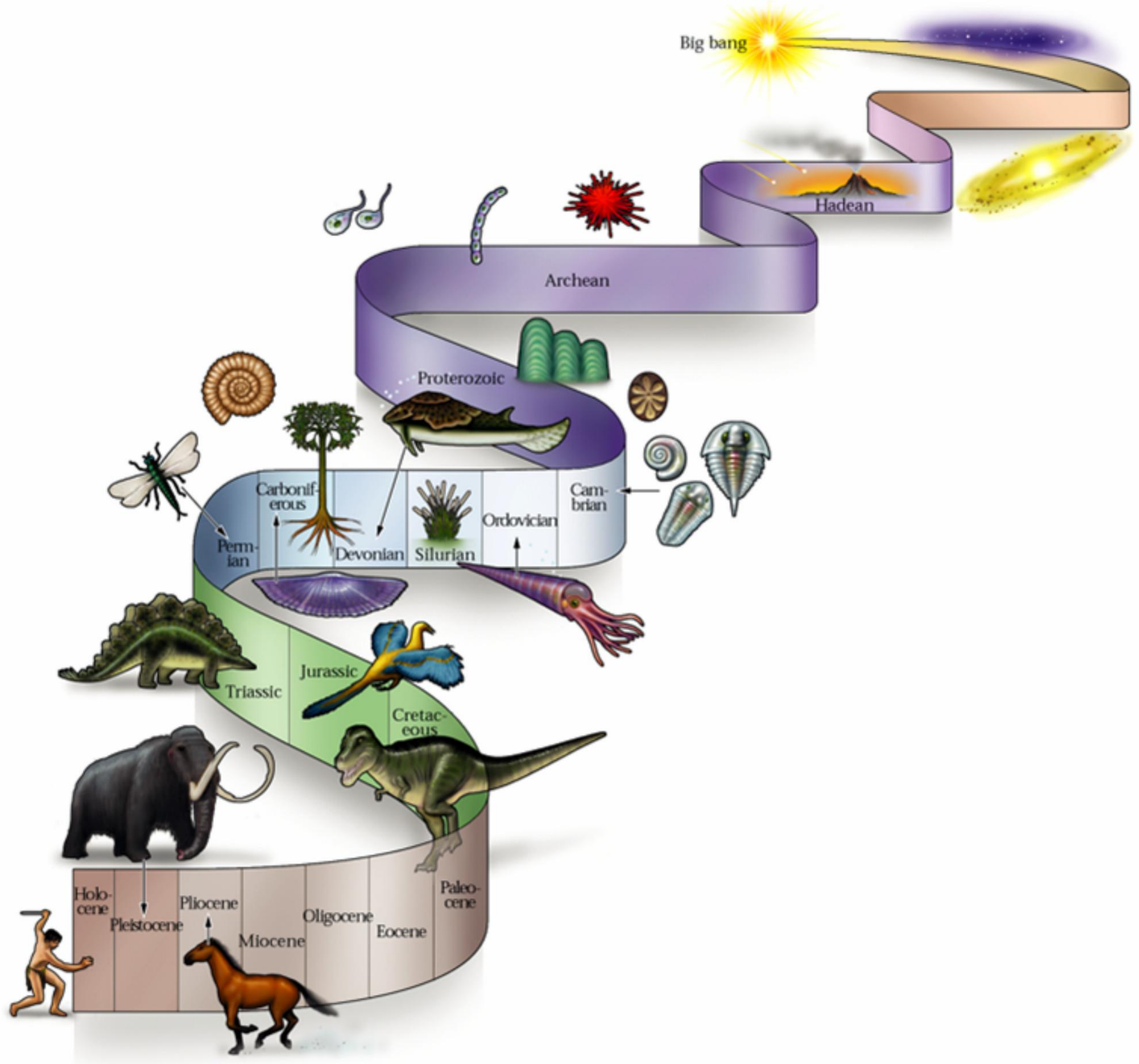
source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Evolutionary Systems

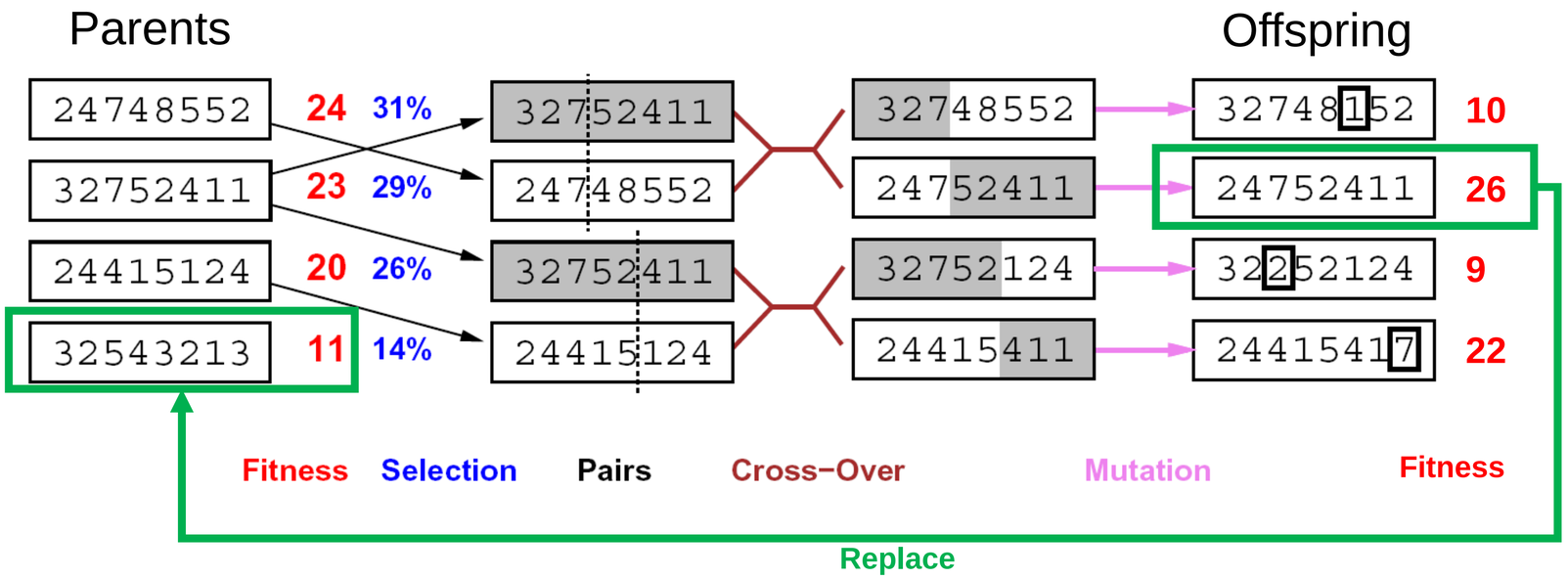
- All biological systems are the result of an evolutionary process
- Biological systems are:
  - robust
  - complex
  - adaptive

- Evolutionary algorithms are abstractions of natural evolutionary processes with the aim of automatically finding solutions to complex problems.
  - ➢ E.g. As global optimizers

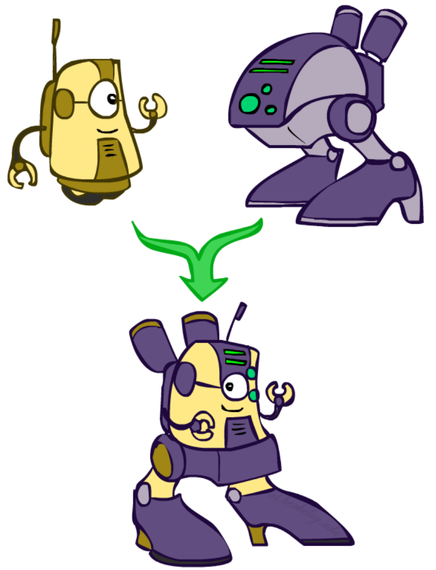


This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Genetic Algorithms



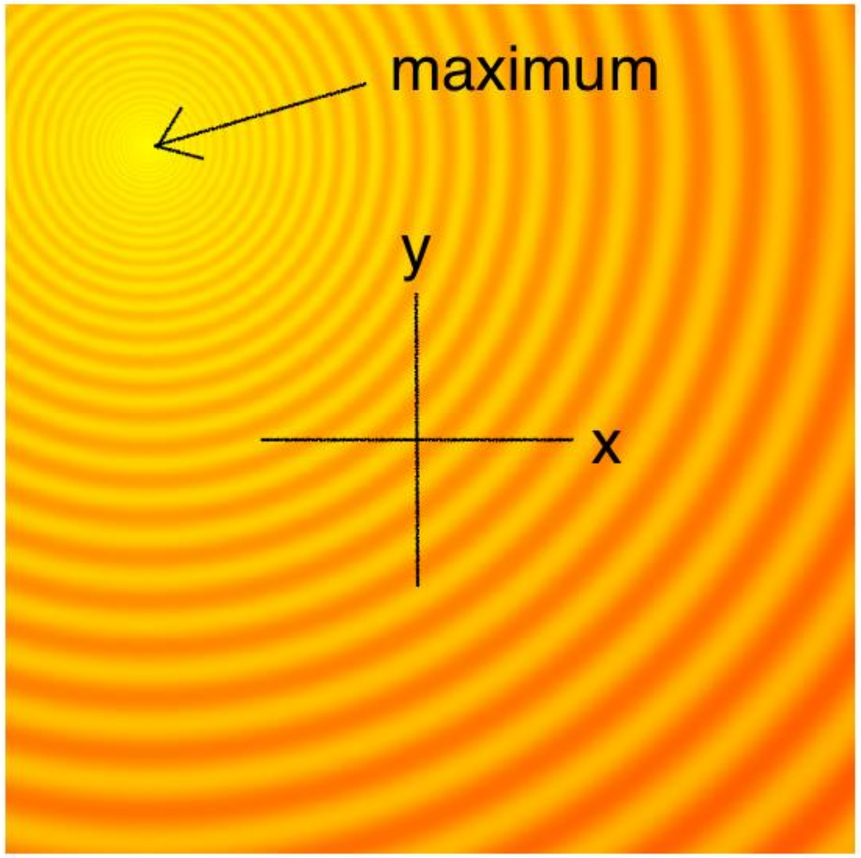
source

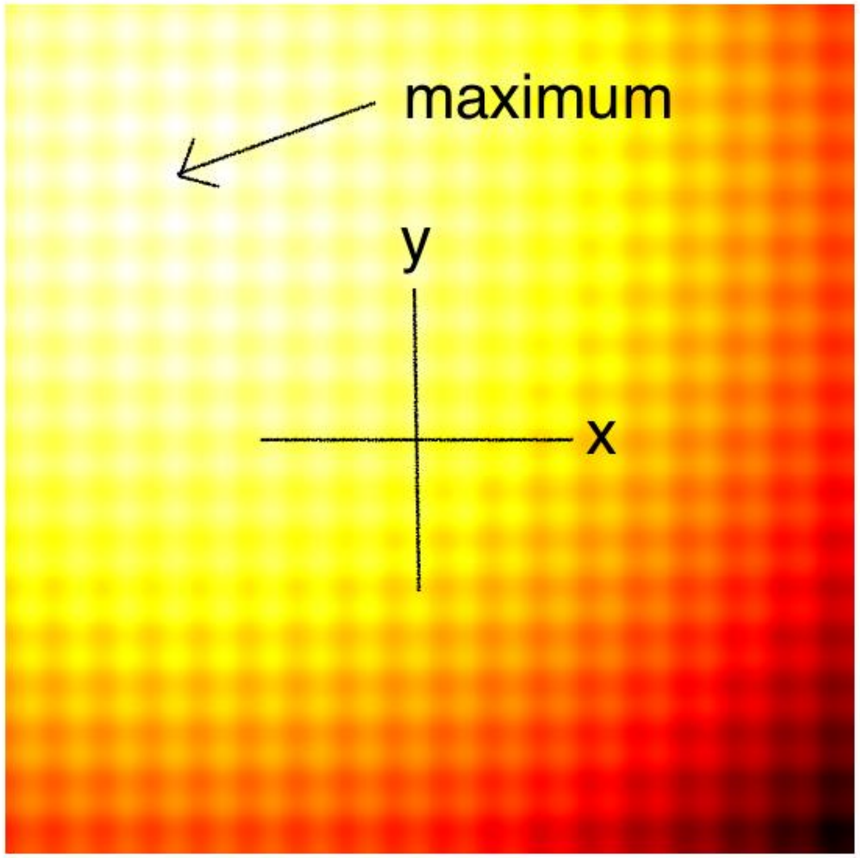


- Natural selection metaphor
  - Parents produce offspring through crossover and mutation of genes
  - If the offspring has traits that are advantageous to its environment (better fitness) it survives, thus, allowed to propagate its genes in the next generation

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Evolutionary Algorithms in Action

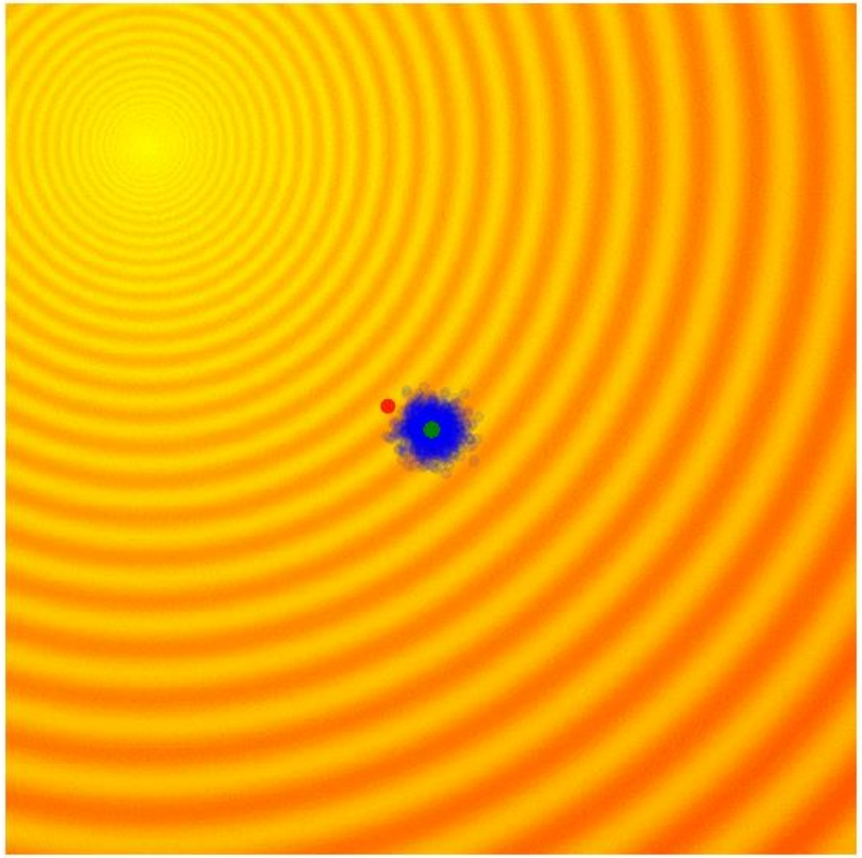


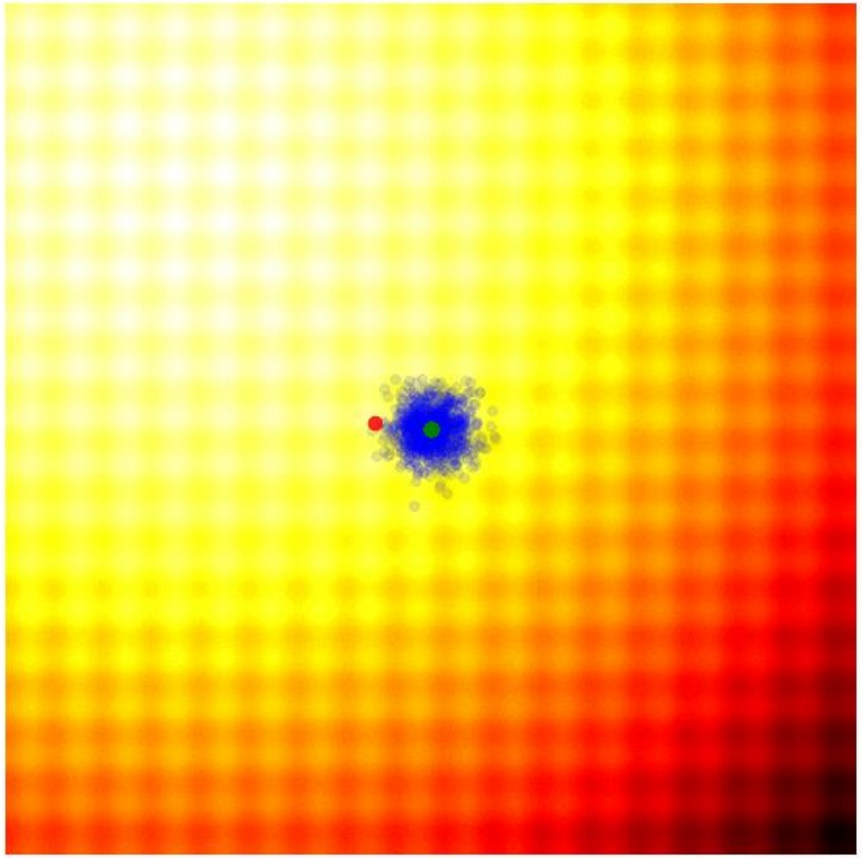


source

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Simple Genetic Algorithms

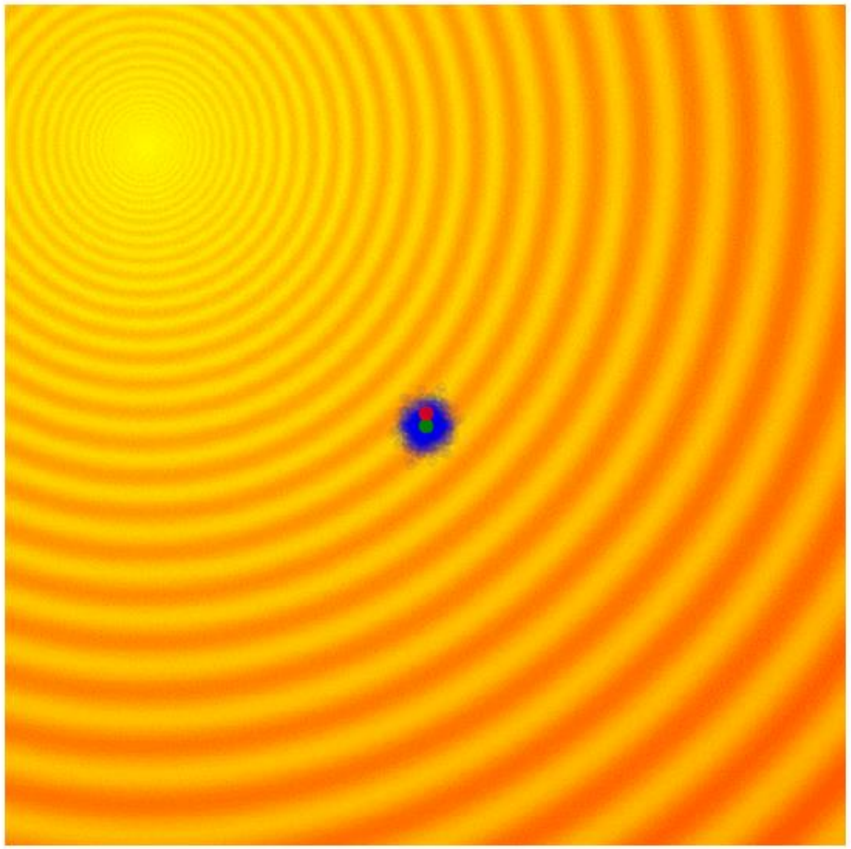


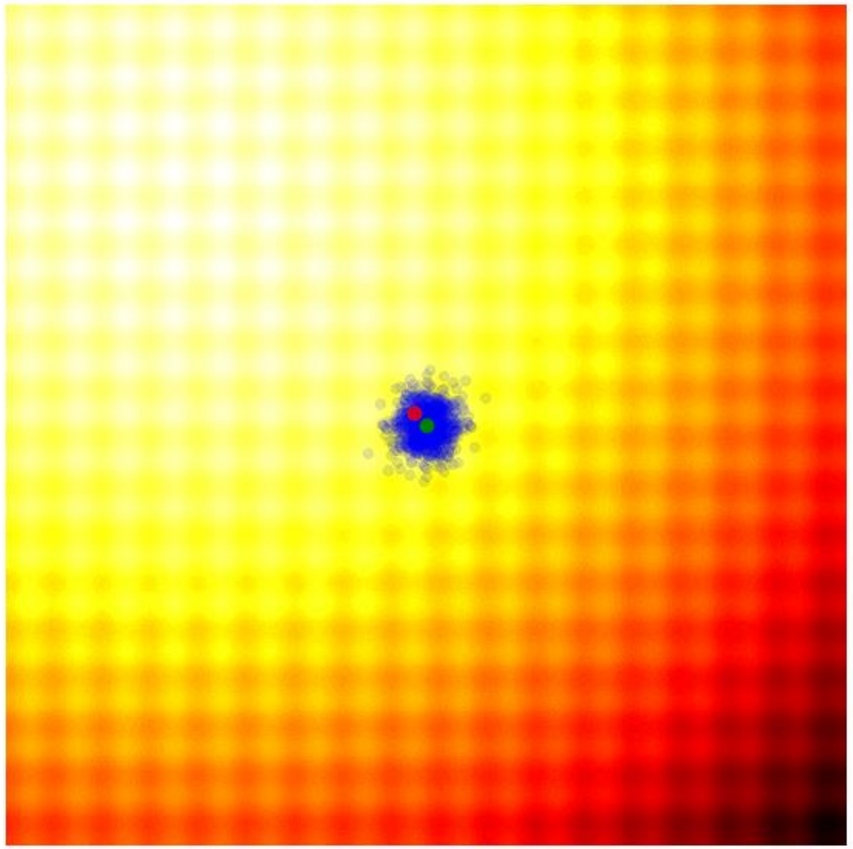


source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Simple Evolution Strategies

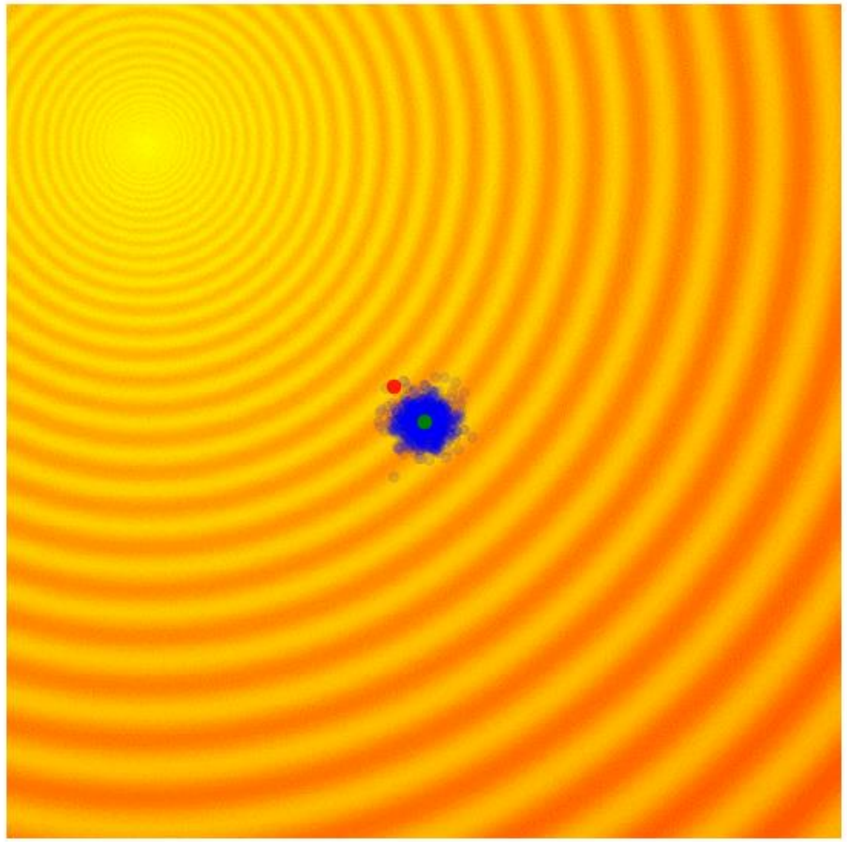


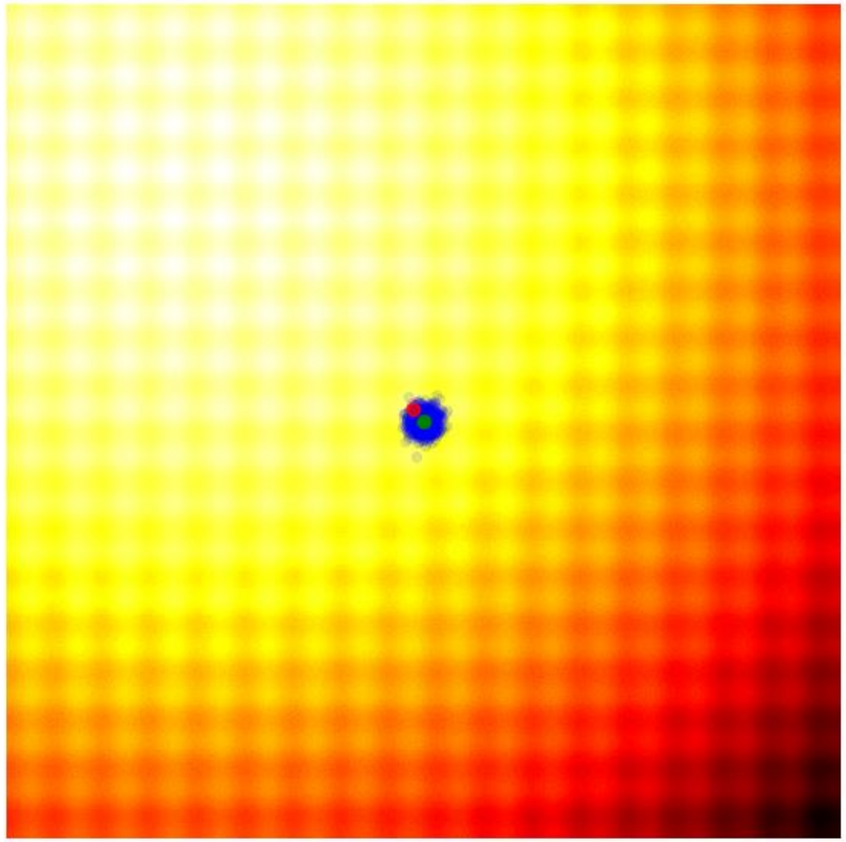


source

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Advanced Evolution Strategies: CMA-ES





source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neuroevolution

- Evolution of neural networks
- Inspired by the fact that natural brains are the products of an evolutionary process
- Can be used to optimize:
  - Parameters (weights) – rather than gradient descent
  - Connectivity (topology)
  - Activation functions
  - Learning algorithms (evolution of learning) – more relevant for reinforcement learning problems
- Need to define the representation of the network and the operators (cross-over, mutation) that modify the representation
- A complementary approach to other machine learning algorithms!

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Next Lectures

- Neural Networks 3: Introduction to Deep Learning

Part 3: Unsupervised Learning

- Clustering

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 11: Neural Networks 3: Introduction to Deep Learning

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Revision

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neural Networks

- Artificial Neural Networks (ANNs) are models inspired by the human brain
- Composed of many simple processing elements called artificial neurons
- Artificial neurons compute the weighted sum of their inputs and feed it through an activation function which then becomes their output
- Activation functions:
  - Heaviside step (non-differentiable) → Perceptron model
  - Linear → Linear regression
  - Sigmoid → Logistic regression
- Perceptrons are linear classifiers
  - By combining multiple perceptrons in layers we can classify nonlinearly separable problems
  - E.g., XOR can be solved using 2 hidden neurons and 1 output neuron
  - However, we cannot train them using gradient descent when they use the Heaviside step function as it is non-differentiable

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neural Networks

- Multilayer Perceptrons is a synonym for feedforward ANNs (typically with differentiable activation functions)
- As a classifier, an MLP with:
  - 1 hidden layer forms open or convex decision regions
  - 2 hidden layers create arbitrary decision regions
- Forward propagation is the process that feeds a data instance to the input layer of a NN and this is gradually transformed into the output prediction (regression or classification) through some nonlinear transformation.
  - This nonlinear transformation computes features of the input which are learned
  - A second hidden layer computes features as functions of existing features
- Learning in NNs can be done using backpropagation and gradient descent

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neural Networks

- Backpropagation: an efficient way to compute partial derivatives of the error function with respect to each parameter using the chain rule (since a NN is a composition of functions)
- Error function: MSE for regression, Cross-Entropy for classification
- Forward propagation computes the activation (output) of each node, while Backpropagation computes the error (delta) of each node
- Delta (error) of each node: A x B
  - A = derivative of the node's activation function
  - B = derivative of error with respect to the node's output
    - For an output node: this is the derivative of the error function wrt to the activation of the output node
    - For a hidden node i at layer k: this is the sum of all deltas at nodes in layer k+1 (which are connected to node i) multiplied by their connecting weight
- Gradient of the Error wrt to a weight = (delta of postsynaptic node) x (output of presynaptic node)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neural Networks

- Stochastic GD: weight update after the presentation of every pattern
- Batch GD: weight update after the presentation of all patterns in the training set
- Mini-batch GD: weight update after the presentation of subsets of patterns in the training set
- Momentum term: memory of previous direction, speeds up learning
- Early stopping: way to prevent overfitting by stopping training when the validation error starts increasing
- We can improve the performance of NN models using regularization, hyperparameter tuning and ensembles
- Learning the NN topology can be done using gradient-free methods, such as evolutionary algorithms.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 11: Neural Networks 3: Introduction to Deep Learning

Learning Outcomes

You will learn about:

1. What deep learning is and why use it
2. Convolutional networks for image data
3. Handling sequential data
4. Recurrent neural networks
5. Backpropagation through time
6. The vanishing and exploding gradient problem
7. Echo State Networks
8. Long short-term memory (LSTM) networks
9. Word embeddings for handling text

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

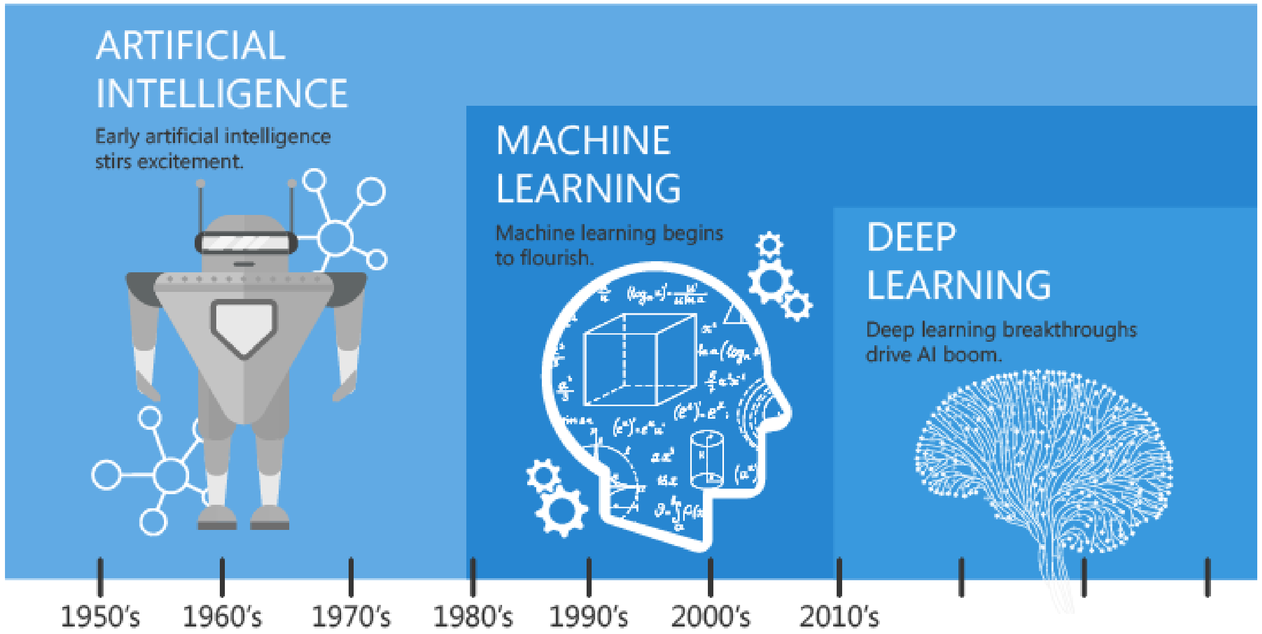
From Machine Learning to Deep Learning

- ML model transforms input data into meaningful outputs
  - Learned using examples
- How to meaningfully transform data?
  - How to learn representations of the input data that get us closer to the expected output
- Representation: different way to look at the data – to represent or encode the data
- Examples:
  - Color image can be encoded in the RGB (red-green-blue) or HSV (hue-saturation-value) format
  - Points on a plane can be represented using Cartesian (x,y) or polar (r,θ) coordinates
- Some tasks may be difficult with one representation and easier with another. Example:
  - Task: “select all red pixels in the image” simpler with RGB format
  - Task: “make the image less saturated” simpler in the HSV format
- ML models are all about finding appropriate representations for their input data – transformations of the data that make it more manageable for the task at hand (e.g., classification)

Adapted from: Chollet, F. (2018). Deep Learning with Python. Manning Publications.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

What is Deep Learning



Since an early flush of optimism in the 1950's, smaller subsets of artificial intelligence - first machine learning, then deep learning, a subset of machine learning - have created ever larger disruptions.

source

- Deep Learning is a subfield of ML that focuses on learning successive layers of meaningful representations using neural networks.
- "Deep" = Successive layers of representation
- Other potential names: layered representations learning, hierarchical representations learning
- How many layers are considered "deep"?
  - 2 layers of representations: shallow learning
  - >2 layers: deep learning
  - Often: tens or hundreds of successive representations

Adapted from: Chollet, F. (2018). Deep Learning with Python. Manning Publications.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Why deep learning? Why now?

1. Hardware

- 1990-2010: CPUs became faster by a factor of ~5000
- Throughout the 2000s companies like NVIDIA and AMD have been heavily investing in developing fast, massively parallel chips (graphics processing units [GPUs]) to power the graphics of increasingly photorealistic video games
- 2007: NVIDIA launched the CUDA programming interface for its line of GPUs
- Small number of GPUs replace massive clusters of CPUs in various highly parallelizable applications (e.g., physics modeling)
- 2011-2012: First CUDA implementations of NNs that won competitions (e.g., ImageNet)
- The deep-learning industry is starting to go beyond GPUs (e.g., Tensor Processing Units [TPUs])

Adapted from: Chollet, F. (2018). Deep Learning with Python. Manning Publications.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Why deep learning? Why now?

2. Datasets and benchmarks

- AI is the new industrial revolution
  - If AI is the steam engine of this revolution, then data is its coal
- Exponential progress in storage hardware over the past 20 years
- Game changer: rise of the Internet, makes it feasible to collect and distribute very large datasets for ML
- Today: companies work with image datasets, video datasets and natural-language datasets that could not have been collected without the internet
- ImageNet dataset: Catalyst for the rise of deep learning
  - 1.4 million images hand annotated with 1000 image categories
  - Yearly competition
- Public competitions (e.g., Kaggle): excellent way to motivate researchers to advance the state of the art

Adapted from: Chollet, F. (2018). Deep Learning with Python. Manning Publications.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Why deep learning? Why now?

3. Algorithmic advances

- Until the 2000s, we were missing reliable ways to train deep NNs
- NNs were shallow (1-2 hidden layers), unable to compete against SVMs or Random Forests
- Key issue: gradient faded away as number of layers increased
- 2009-2016: Algorithmic improvements allow better gradient propagation:
  - Better activation functions for neural layers
  - Better weight initialization schemes, starting with layer-wise pretraining, which was abandoned
  - Better optimization schemes (e.g., RMSProp, Adam)
  - Batch normalization
  - Residual connections
- However, older techniques such as LSTMs were now feasible because of hardware & data

Adapted from: Chollet, F. (2018). Deep Learning with Python. Manning Publications.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Why deep learning? Why now?

4. A new wave of investment

- Deep learning (DL) became the new state of the art for computer vision in 2012-2013 and eventually all perceptual tasks – industry leaders took notice
- Gradual wave of industry investment afterwards, far beyond anything previously seen in the history of AI. For example, the total venture capital investment in AI:
  - 2011 (before deep learning spotlight) = ~$19M
  - 2014: ~$394M
- 2013: Google acquired DeepMind for $500M (largest acquisition of an AI startup)
- 2014: Baidu started a DL research centre in Silicon Valley (investing $300M)
- Deep Learning became central to the product strategy of tech giants
- Research and number of people working on/with deep learning increased exponentially over the last years

Adapted from: Chollet, F. (2018). Deep Learning with Python. Manning Publications.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Why deep learning? Why now?

5. The democratization of deep learning

- Early days required significant C++ and CUDA expertise, which few people possessed
- Nowadays, basic Python scripting skills suffice to do advanced DL research
- This has been driven primarily by the development of Theano, TensorFlow, and PyTorch: symbolic tensor[1]-manipulation frameworks for Python that support autodifferentiation
- Rise of user-friendly libraries, such as Keras, which makes DL easy
- Culture on uploading papers on arXiv and releasing the code as open-source (see paperswithcode)

[1]Tensor: Generalizaton of a matrix to N-dimensional space

Adapted from: Chollet, F. (2018). Deep Learning with Python. Manning Publications.

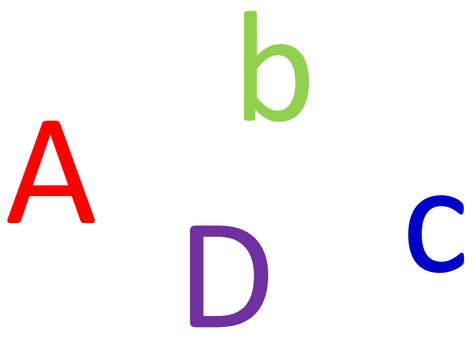
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Structure helps learning

If data has some structure and the NN does not need to learn the structure from scratch, then the NN will perform better

- **Example**: Letter classification



Color does not matter!

What do you think would be easier for a classifier to learn?

1. A model that uses the color image?

2. A model that only looks at the gray scale?

$$\frac{R+G+B}{3} \rightarrow \text{CLASSIFIER}$$

Adapted from Udacity's Deep Learning course

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

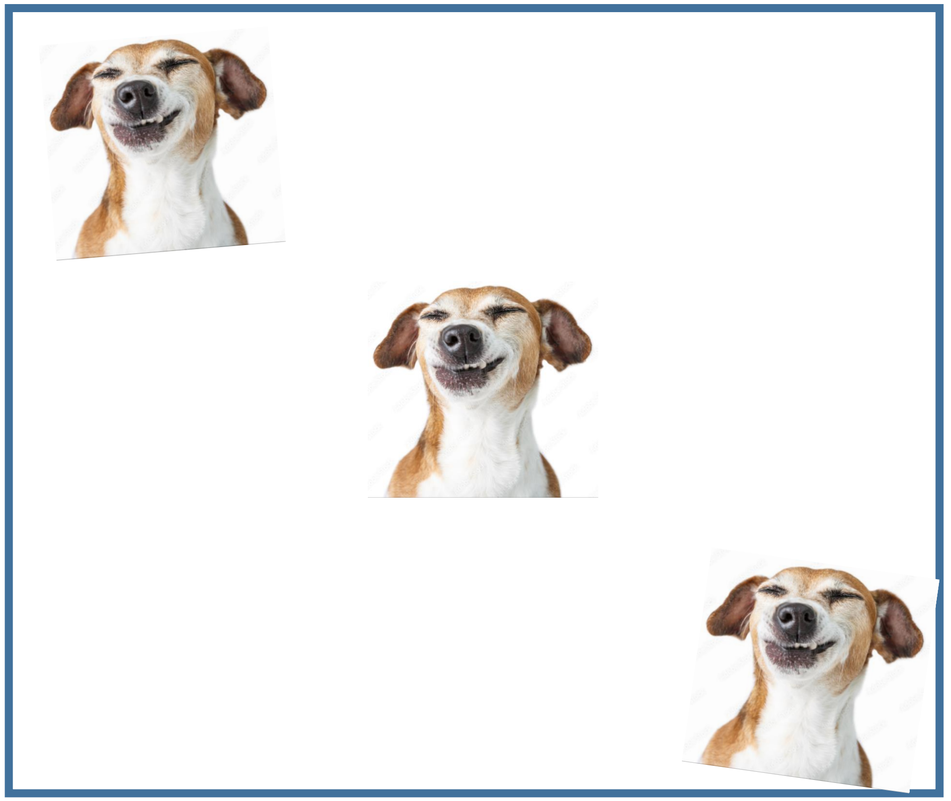
Statistical Invariance

Example: Dog image classification

- Does not matter where the dog is, it is still an image with a dog

We should explicitly teach a NN that objects in images are the same irrespective of where they are in the image

This is called **Translation Invariance**



Adapted from Udacity's Deep Learning course

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Statistical Invariance

Example: Text that talks about dogs

The quick brown dog jumped over the lazy white dog.

Once upon a time, there was a dog with pointy ears and a short tail.

Does the meaning of dog change depending on whether it is in the 1st sentence or the 2nd one?

➢ No

We want the part of the NN that learns what a dog is to be reused every time the NN sees the word *dog*, and not to re-learn it every time.

Adapted from Udacity's Deep Learning course

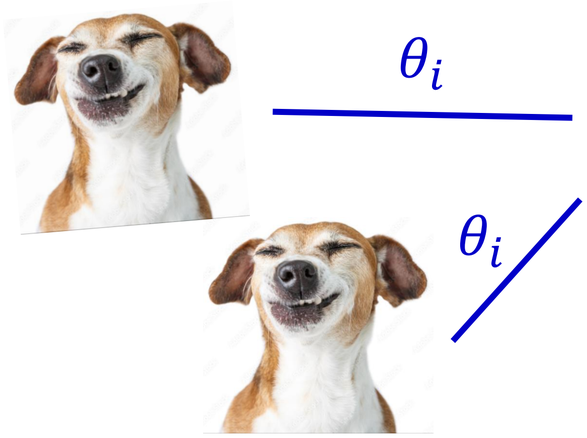
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Statistical Invariance: Weight sharing

When we know that 2 or more inputs contain the same information then we share their weights and train the weights jointly for those inputs

Statistical invariants: things that do not change on average across time or space



Convolutional Networks

Image Data

Embeddings or Recurrent NNs

θ_j *The quick brown dog*

θ_j *jumped over the lazy white dog.*

θ_j *Once upon a time, there was a dog with pointy ears and a short tail.*

Text or Sequences

Adapted from Udacity's Deep Learning course

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Image Data

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Image Data

Example: Classification task

- Input: 256x256x3 (RGB) → tensor
- Output: Cat or not cat

Suppose that we use an MLP with 1 fully connected (FC) hidden layer of 100 neurons

- 196,608 x 100 (input to hidden) + 100 (hidden to output) = 19,660,900 parameters!

Cons:

- MLPs require a huge amount of parameters (prone to overfitting)
- MLPs do not account for translational invariance

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Convolutional Networks (ConvNets)

Convolutional NNs are regularized versions of MLPs, i.e., they reduce the number of parameters and try to account for translation invariance, based on 3 ideas:

- Local receptive fields
  - Each neuron receives input only from a restricted area of the previous layer (typically a square, e.g., 5x5)
  - The learnable parameters of each such neuron are called a filter or a kernel
  - In contrast, the receptive field of FC layers is the entire previous layer

- Weight sharing
  - Each filter is convolved with the previous layer, i.e., slices over the previous entire activation map (e.g., the original image) and computes the dot product between the filter entries and the input patch
  - The output of this operation is an activation map for each filter

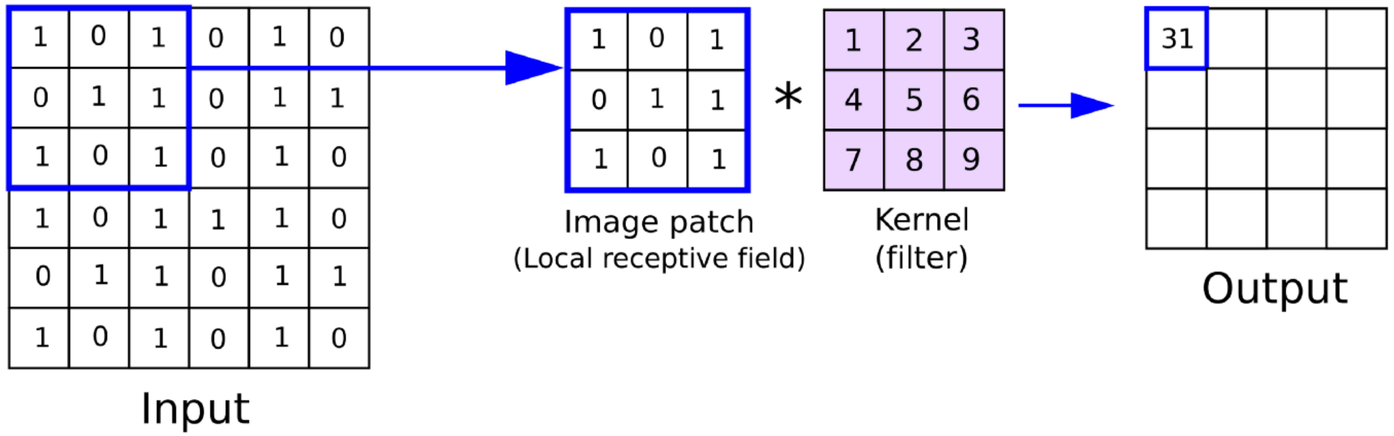
- Spatial sub-sampling (pooling) [optional]
  - Reduces the dimensions of the data by combining the outputs of neuron clusters into a single number
  - Typically, max-pooling or average pooling

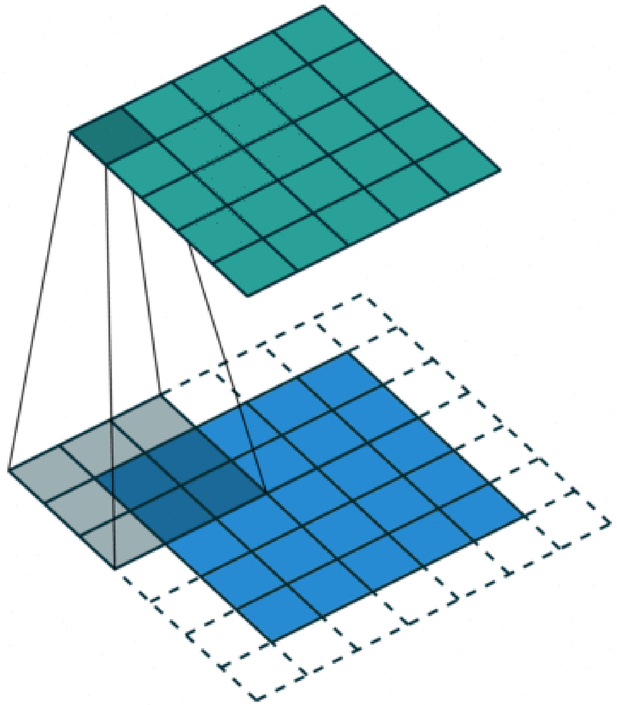
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Convolutional Networks (ConvNets)

- Convolution



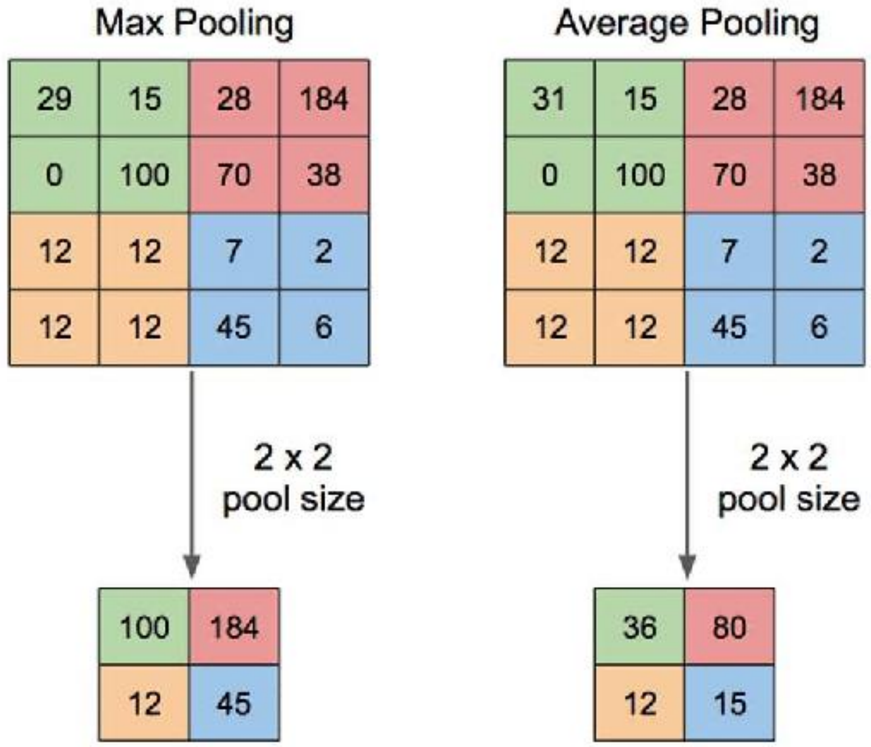


Local receptive field: 3x3 filter
Weight sharing: only 9 parameters are learned to produce the output activation map (of this filter)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Convolutional Networks (ConvNets)

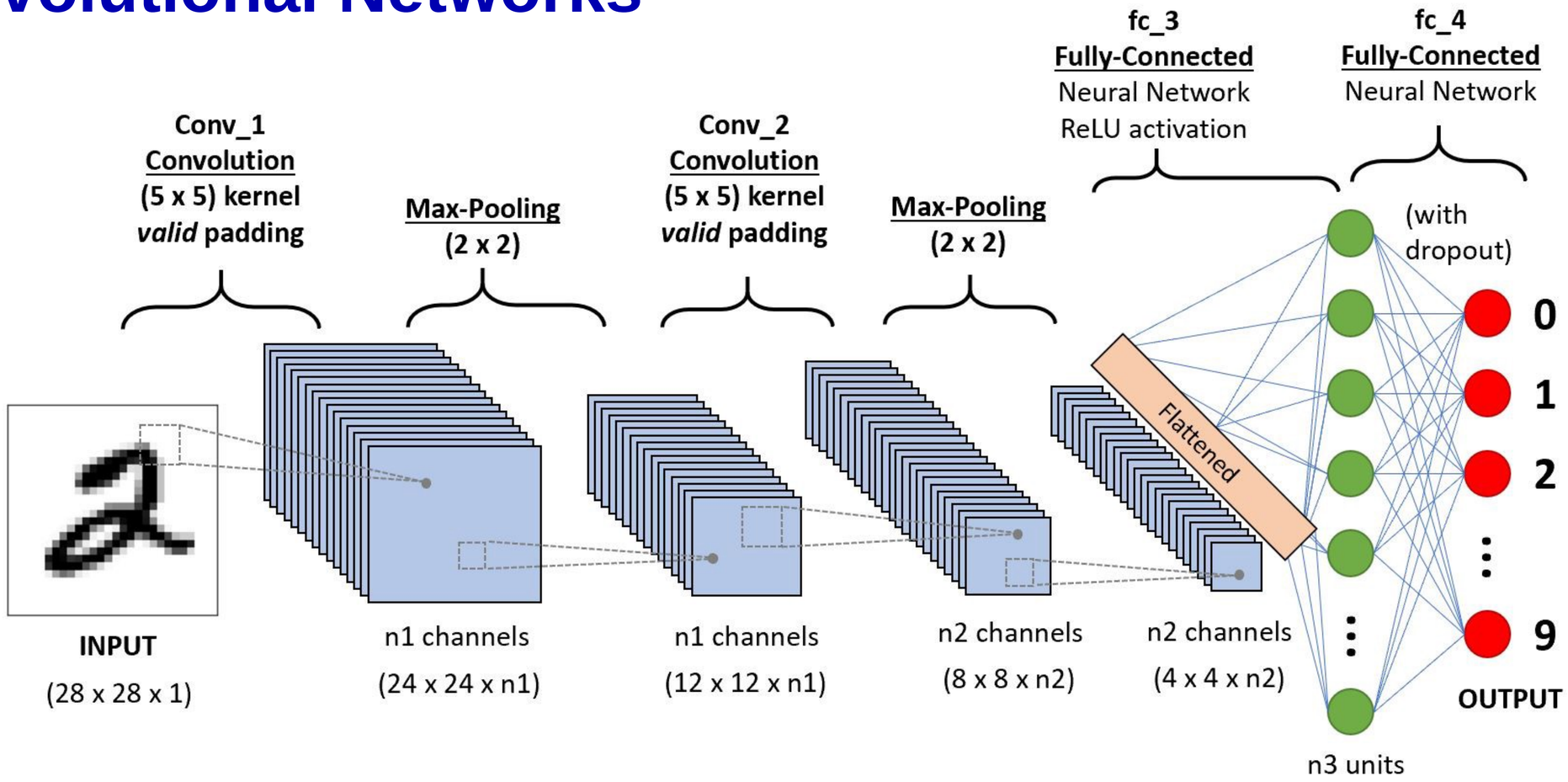
- Pooling



source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Convolutional Networks



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

What ConvNets learn

Each layer in the hierarchy learns progressively better representations

- Input layer: original pixel values
- Layer 1: presence/absence of edges of particular orientation at particular position
- Layer 2: local arrangements of edges
- Layer 3: assemblies of local arrangements that may correspond to parts of objects
- Layer 4: combinations of parts of objects that may represent complete objects
- Output Layer: classification

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Applications of ConvNets

Every task that requires image data:

- Image recognition
- Video recognition
- Image classification
- Image segmentation
- Medical image analysis
- …

Can also be used for other types of data:

- Natural language processing
- Time series forecasting
- …

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Sequential Data

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Sequential data

- Sequential data refers to any data that contain elements that are ordered into sequences
- The order differentiates this data from other cases we have seen so far

- Examples:
  - The price of a stock over time
  - Environmental data (pressure, temperature, precipitation etc.) over time
  - Sequence of queries in a search engine, or the frequency of a query over time
  - EEG signals
  - A DNA sequence of nucleotides
  - The words in a document as they appear in order
  - Event occurrences in a log
  - Video-based action recognition

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Time-series data

- Time series:
  - Typical case of sequential data
  - Vector of numeric values that change over time
  - Typically, collected at regular intervals

- **Components**:
  - Trends: smooth long-term direction
  - Seasonality: patterns of change within a year which tend to repeat each year
  - Cyclic variation: rise and fall over periods typically longer than a year
  - Noise: random variation / unpredictable component

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Quiz

Sequential data and time-series data mean exactly the same thing. True or False?

False. There exist data that contains elements whose order is not defined by a time axis. For example, a DNA sequence, or words in a document.

Nevertheless, in most applications, sequential data are temporal data.

In the following slides, we will be using the word “order” and “time” synonymously.

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Handling sequential data

Why deal with sequential data?

- Stock prices do not make sense without the time information
- Individual words in a sentence do not make sense without their context

How to handle sequential data?

- Need some form of memory.
- Two approaches:
  - Order is externally processed
  - Order is internally processed

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

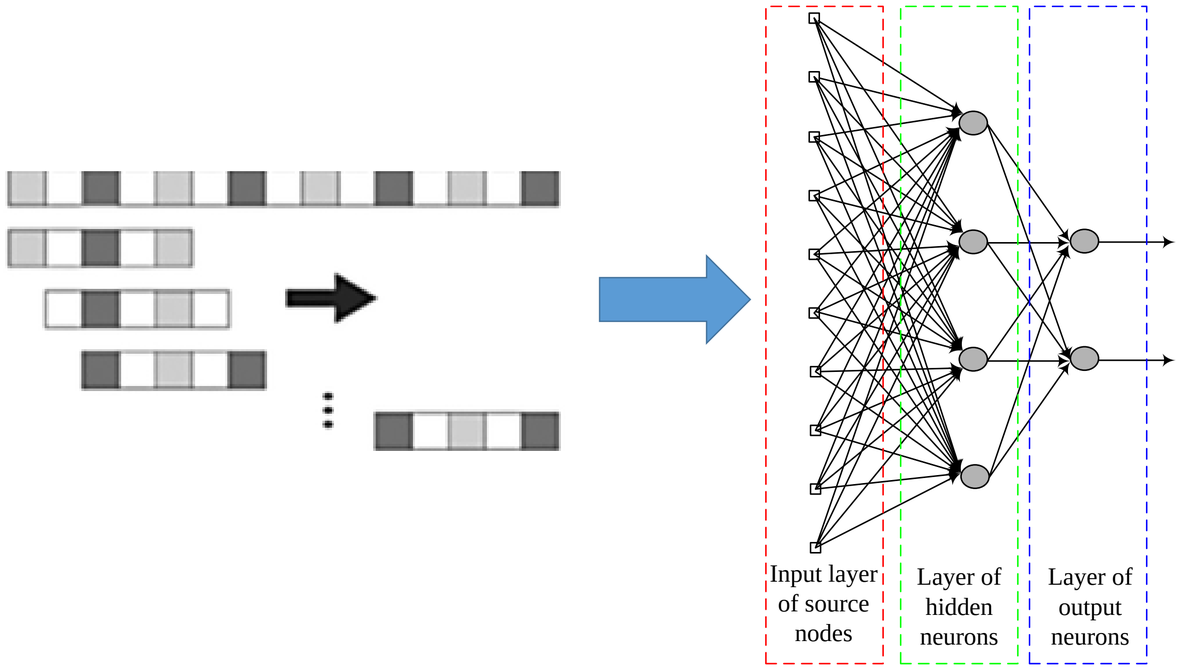
External processing of order

- Order (or time) is preprocessed in an order-to-space transformation. Only the spatial transformation is accessed by the network, which contains a dimension whose semantics are related to the order.

- Feature engineering approach: introduction of new features (inputs)
  - Shifted versions of the input signal, e.g., $x_t, x_{t-1}, x_{t-2}, \dots, x_{t-k}$
  - Moving averages, e.g., $x_{MA,t} = \frac{\sum_{i=t-k}^{t} x_i}{k}$
  - Need to define the length of the context (time) window, k
  - Sliding window on sequential data converts the sequence within the window into static data

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

External processing of order: MLP



Inflexible approach:

- Rigid limit on duration of patterns
- Does not capture the translational invariance of the same temporal pattern at different absolute times

[0,0,1,1,1,0,0,1,0,0] t

[0,0,0,0,1,1,1,0,0,1] t+2

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Internal processing of order

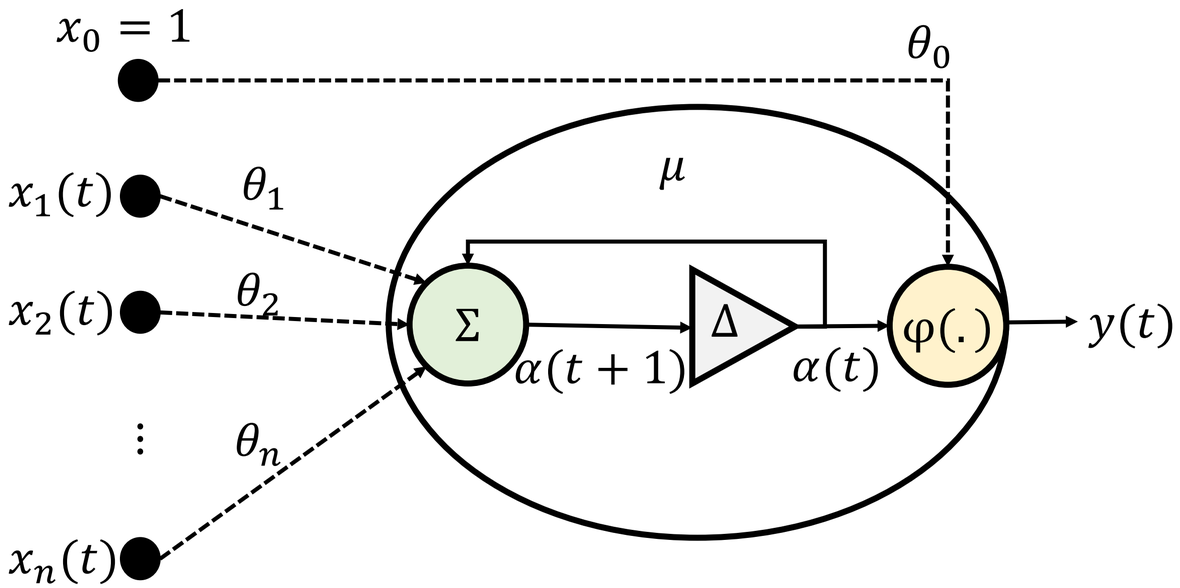
- The model uses an internal state which is constantly updated according to an input stream, and maintains the preceding state of the nodes which is reintroduced at the following step.

- Typically done:
  - At the **neuron** level: the neuron model introduces some delay when integrating the input and calculating the response

  - At the **connection** level: the signal propagates from one neuron to the other with some delay

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Internal processing of order: neuron level



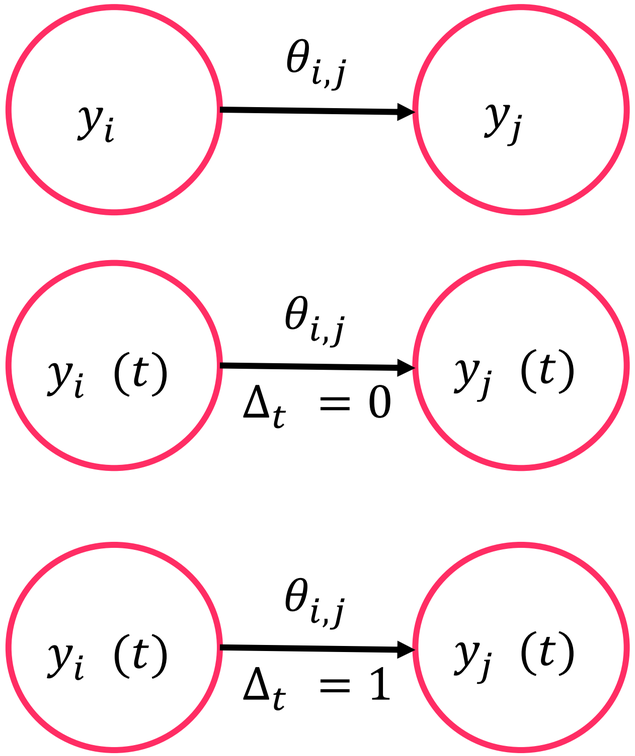
Example: discrete-time dynamic neuron model

$$\alpha(t+1) = \mu\,\alpha(t) + \sum_{i=1}^{n} \theta_i x_i$$

$$y(t) = \varphi(\alpha(t) - \theta_0)$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Internal processing of order: connection level



In feedforward networks, we consider that forward propagation, i.e., the propagation of an input signal all the way to the output layer is done in a single time step.

$$y_j = \varphi(\theta_{i,j} y_i)$$
$$y_j\ (t) = \varphi(\theta_{i,j} y_i\ (t))$$

$$y_j\ (t) = \varphi(\theta_{i,j} y_i\ (t-1))$$
$$y_j\ (t) = \varphi(\theta_{i,j} y_i\ (t-\Delta_t))$$

Connections with a delay are known as **recurrent connections**.

Neural networks that use recurrent connections are known as **recurrent neural networks**.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

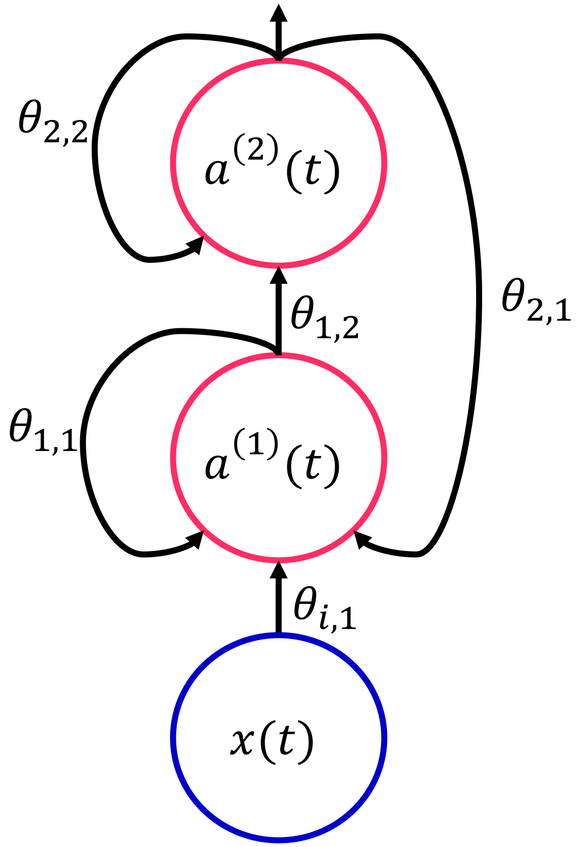
Master programmes in Artificial Intelligence 4 Careers in Europe

Recurrent Neural Networks

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Recurrent Neural Networks: forward propagation



Recurrent connections typically have a delay of 1

$$a^{(1)}(t) = \theta_{i,1}\, x(t) + \theta_{1,1}\, a^{(1)}(t-1) + \theta_{2,1}\, a^{(2)}(t-1)$$

$$a^{(2)}(t) = \theta_{1,2}\, a^{(1)}(t) + \theta_{2,2}\, a^{(2)}(t-1)$$

Every neuron that has an outgoing recurrent connection is said to maintain some **state**

➢ This is its activation which is needed at the next time step

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Training: Backpropagation through time

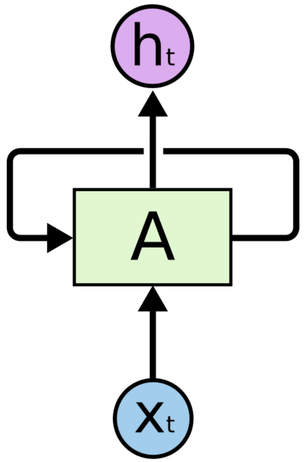
- Backpropagation can still be used to train recurrent neural networks

- Modification:
  - forward propagation: unroll the computational graph over time, i.e., maintain all neuron states in memory for the length of the sequence, t = 1 to T
  - backpropagate errors starting from the last time step (T) towards the first time step, while at each time step we also backpropagate errors from the output layer (if any)
  - accumulate all gradients and apply gradient descent

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Unroll the computational graph

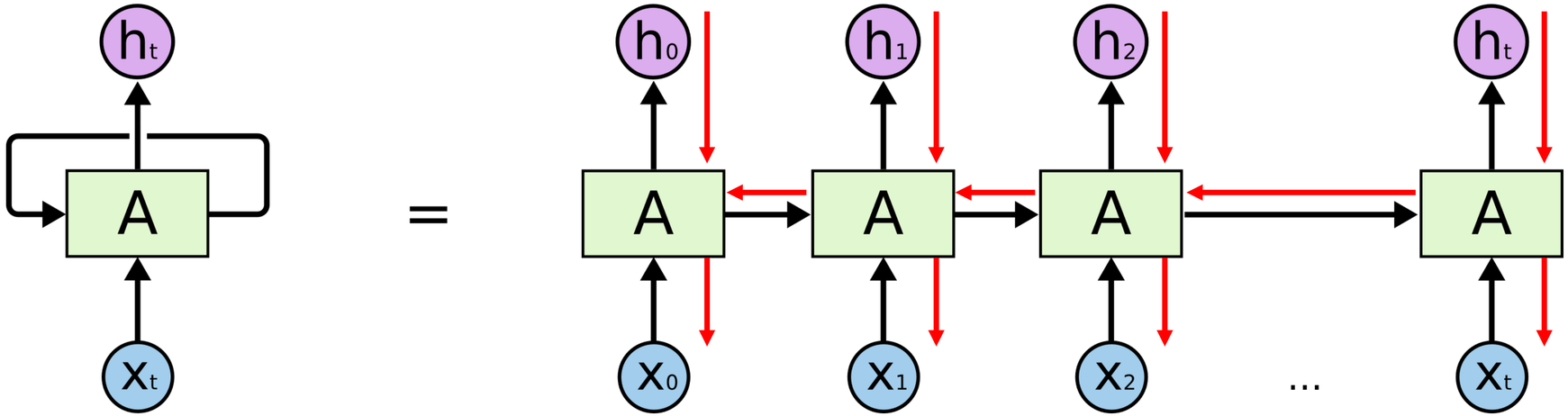
The network becomes very deep



Source

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

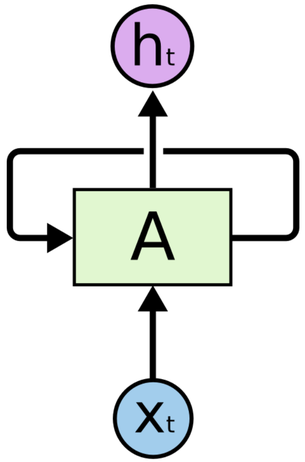
Backpropagate errors



Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Apply parameter update



Source

Once all gradients are accumulated, we do a gradient descent step and modify the parameters

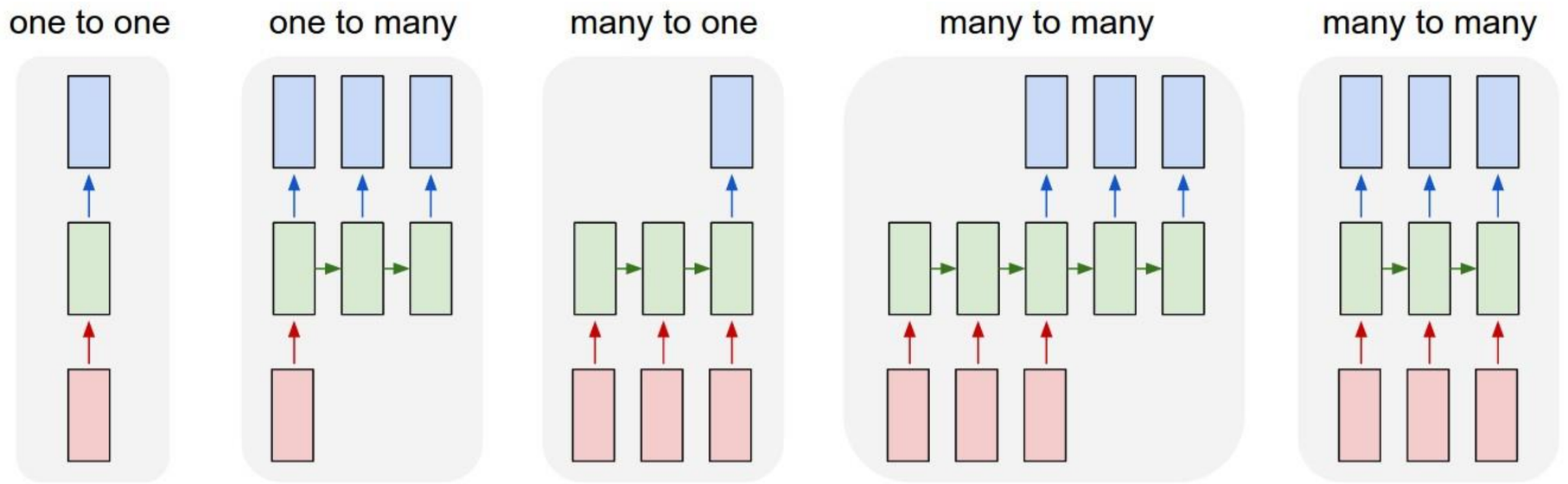
Notice that we use **weight sharing**: the unrolled graph uses the same weights for the processing of different inputs in the sequence

- If the weights were different then it would be a type of feedforward neural network

This enables RNNs to limit computational resources and process sequences of variable length

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

RNN Examples

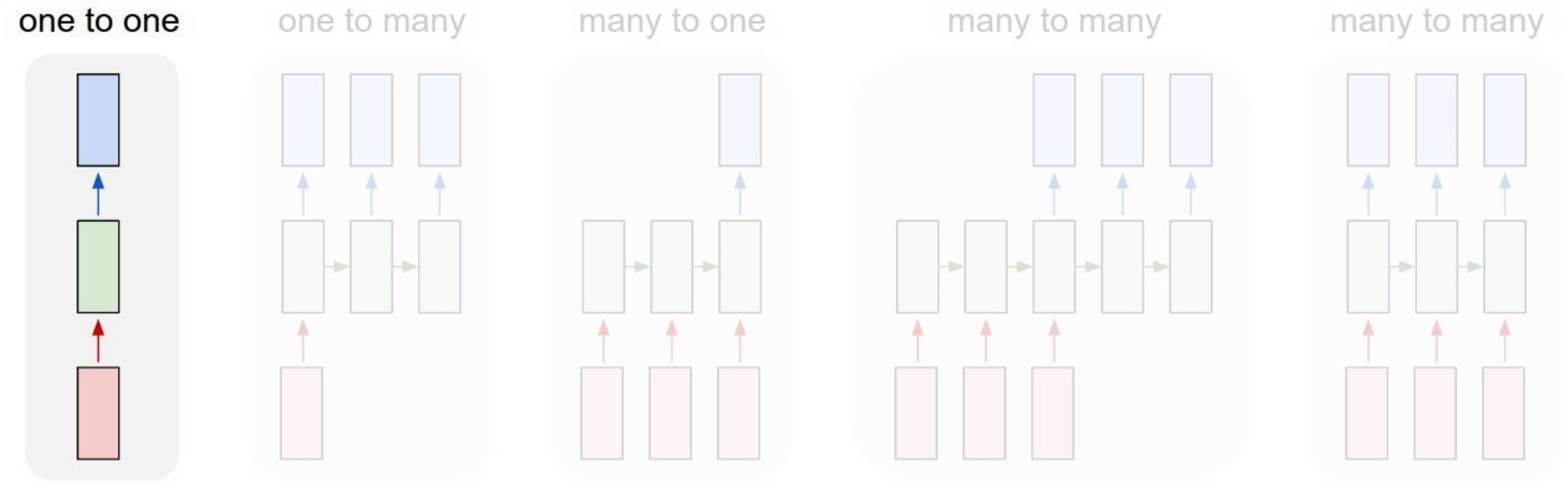


source

Each rectangle is a vector and arrows represent functions (e.g. matrix multiply). Input vectors are in red, output vectors are in blue and green vectors hold the RNN's state

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

RNN Examples

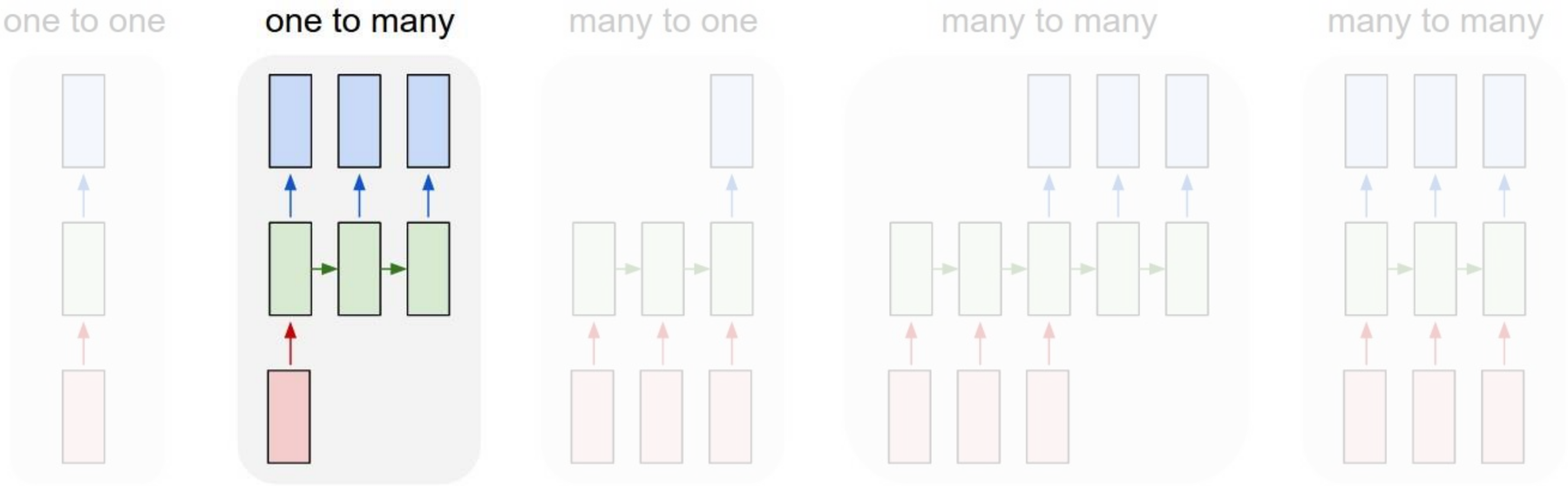


source

Vanilla mode of processing without RNN, from fixed-sized input to fixed-sized output (e.g. image classification).

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

RNN Examples

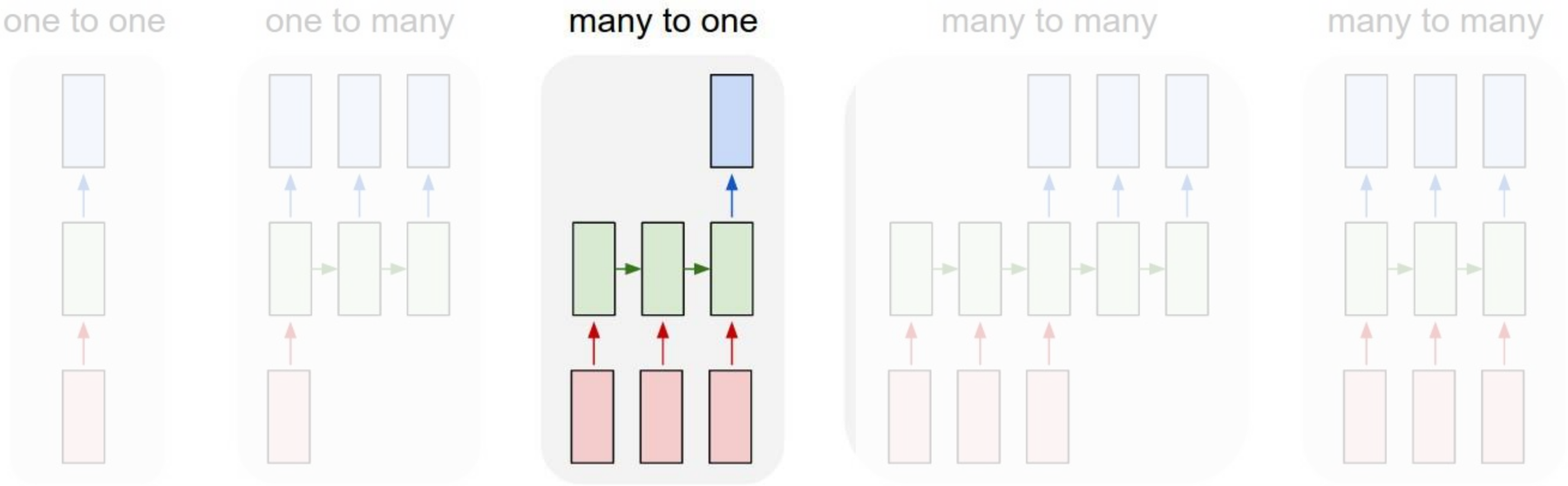


source

Sequence output (e.g. image captioning takes an image and outputs a sentence of words).

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

RNN Examples

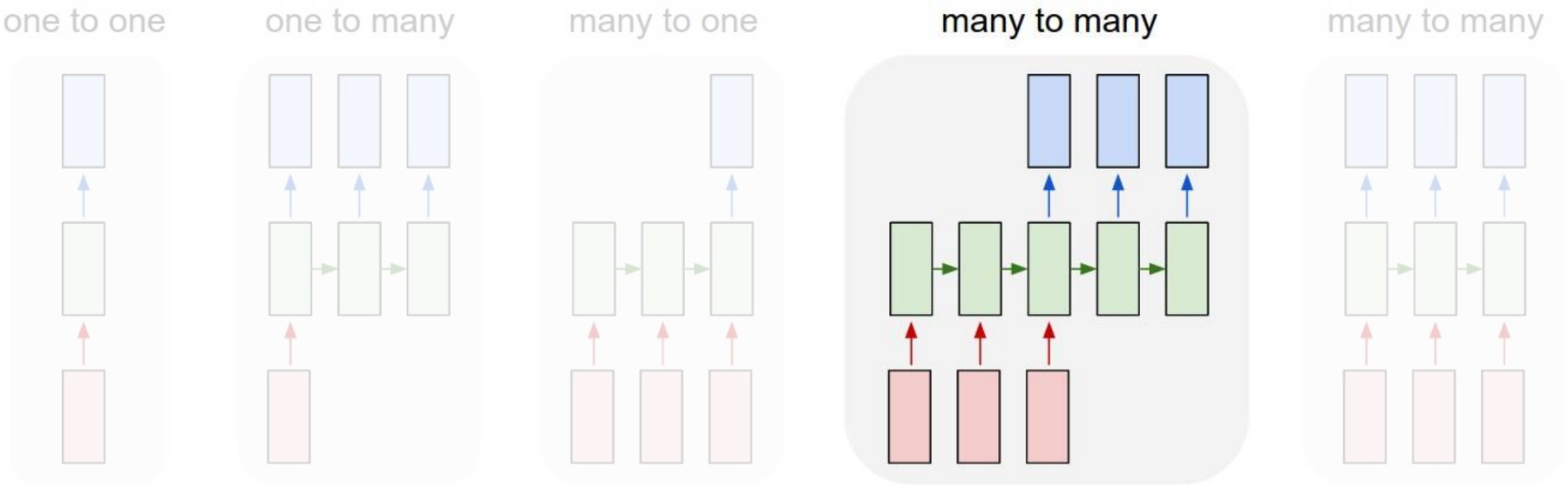


source

Sequence input (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment).

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

RNN Examples

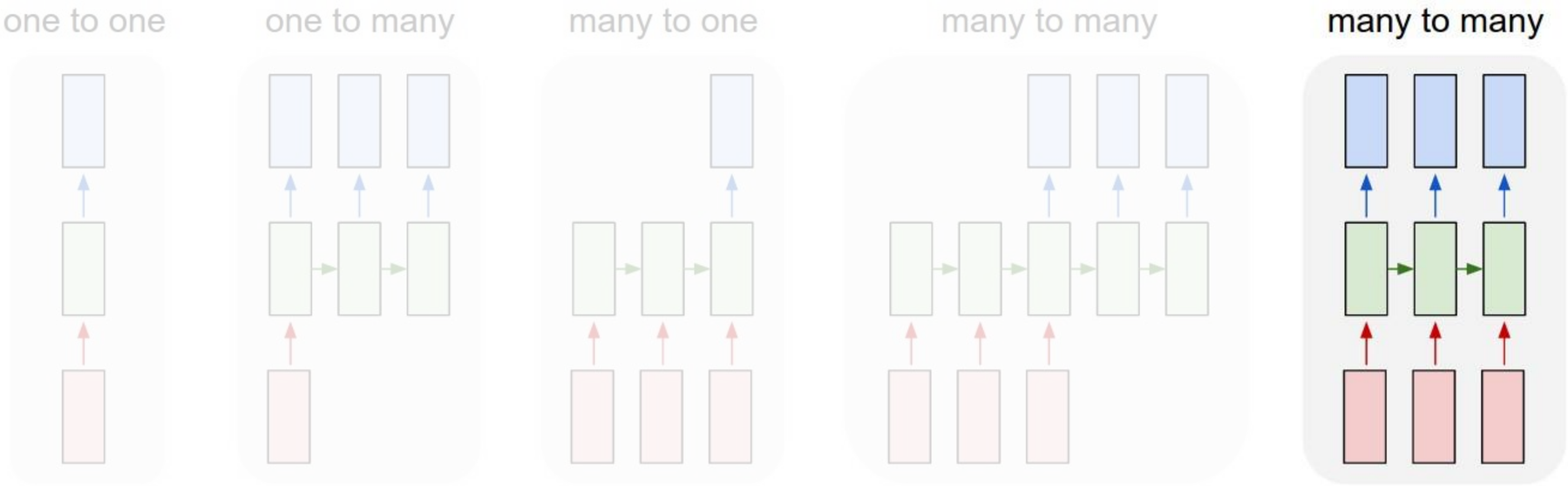


source

Sequence input and sequence output (e.g. Machine Translation: an RNN reads a sentence in English and then outputs a sentence in French).

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

RNN Examples

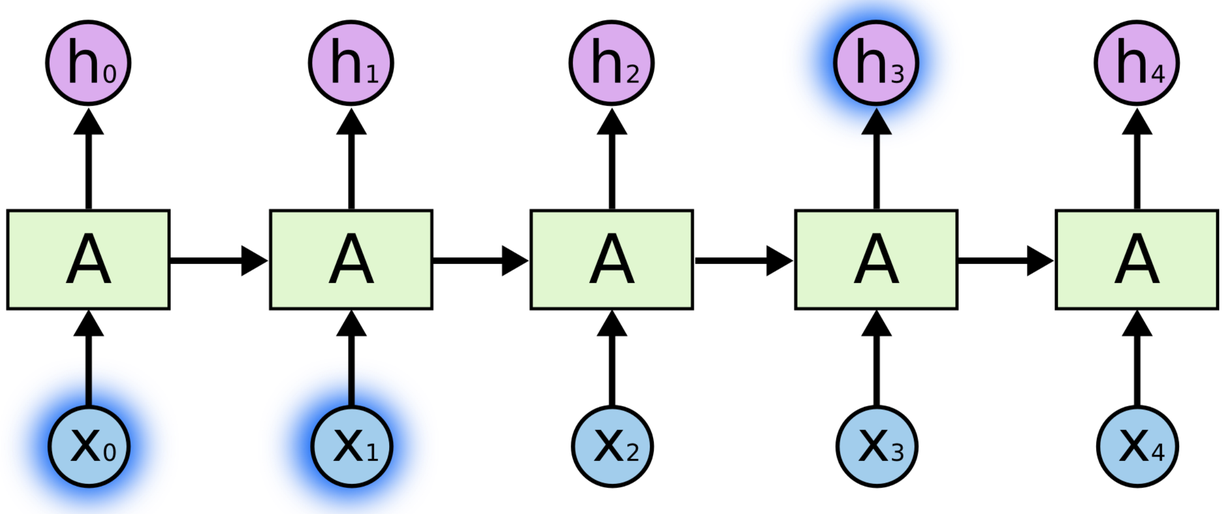


source

Synced sequence input and output (e.g. video classification where we wish to label each frame of the video).

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

The problem of long-term dependencies



Source

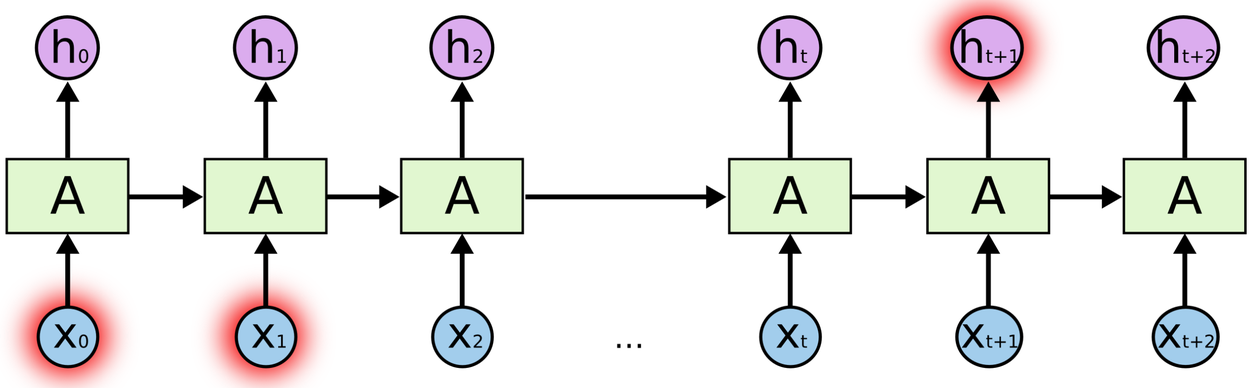
Consider a language model trying to predict the next word based on the previous ones

If we are trying to predict the last word in “the clouds are in the *sky*,” we don’t need any further context – it’s pretty obvious the next word is going to be *sky*.

In such cases, where the gap between the relevant information and the place that it’s needed is small, RNNs can learn to use the past information.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

The problem of long-term dependencies



Source

Consider trying to predict the last word in the text “I grew up in France […] I speak fluent *French*.”

Recent information suggests that the next word is probably the name of a language.

If we want to narrow down which language, we need the context of France, from further back.

Gap between relevant information and point where it needed can become very large.

As that gap grows, RNNs become unable to learn to connect the information.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

The vanishing and exploding gradient problem

- When backpropagating the gradients in a deep network (e.g., a recurrent neural network unrolled for many time steps) from the output layer to the input layer it is often the case that the gradients get smaller and smaller and approach zero.
  - Gradient descent leaves the weights of the lower layers unchanged
  - Never converges to the optimum
  - This is known as the **vanishing gradients problem**

- In some cases, the gradients become larger and larger
  - This causes large weight updates which makes gradient descent to diverge
  - This is known as the **exploding gradients problem**

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Why do gradients vanish?

- Activation functions such as the sigmoid, squashes a large input space into a smaller output space that lies in [0,1]
- Larger inputs (positive or negative) saturate the function which results in a derivative close to zero
- This means that there are practically no gradients to propagate backward while any residual ones keep on diluting.

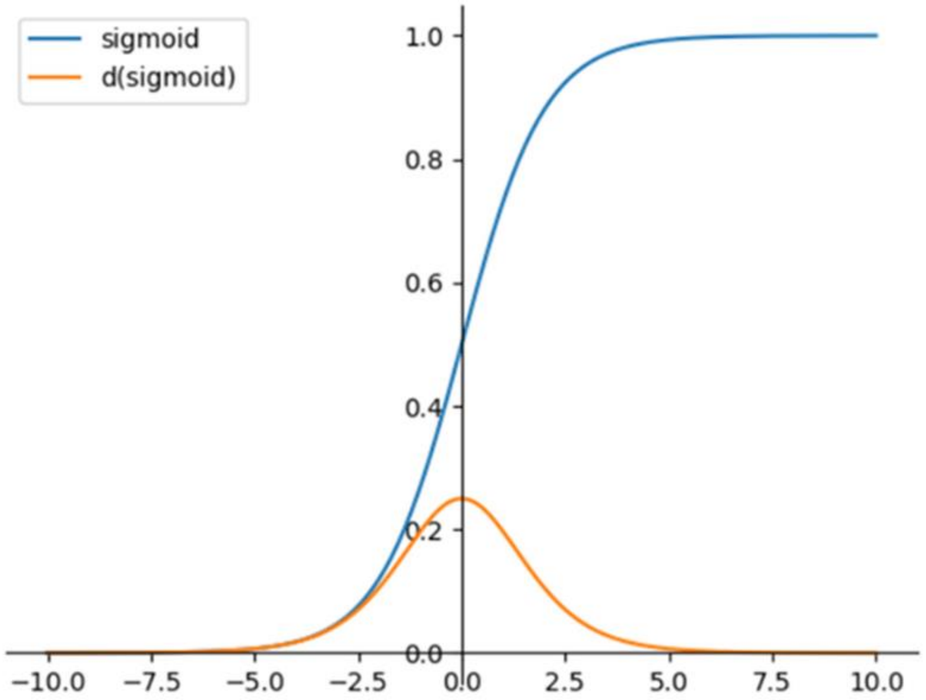


Image source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Why do gradients explode?

- Suppose that the initial parameters result in some large loss.
- The gradients can accumulate during an update and result in very large gradients which eventually result in very large updates to the network parameters
- It is sometimes the case that the parameters overflow and produce NaN values

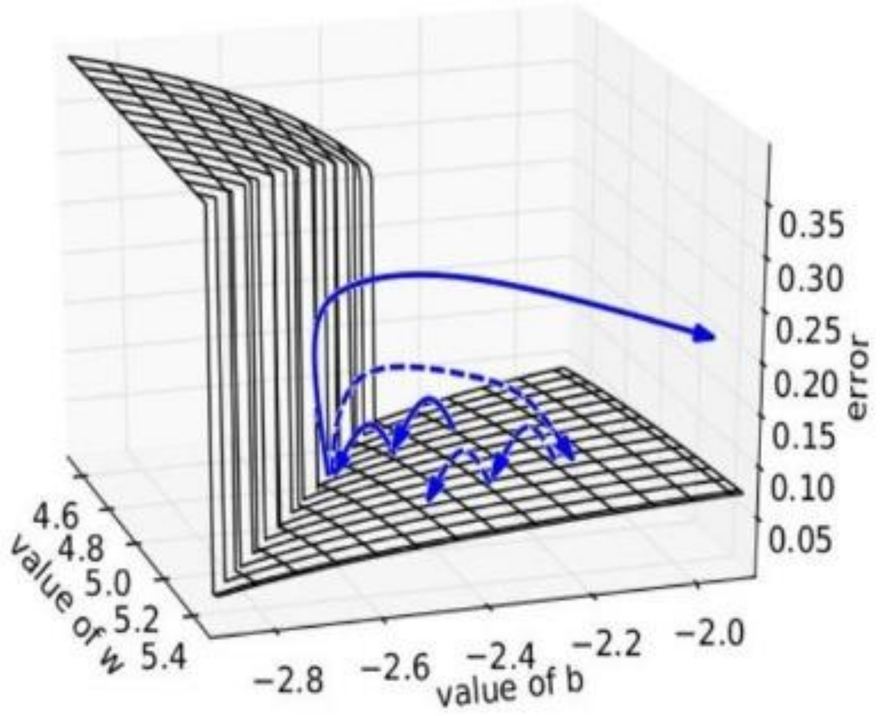


Image source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Potential solutions?

- Better weight initialization (e.g., considering the variance of inputs and outputs at each layer)
- Use non-saturating activation functions (ReLU, Leaky ReLU, ELU,…)
- Batch normalization (zero-mean normalization of input to a layer based on mini-batch data)
- Gradient clipping (gradient values beyond -1 or +1 are clipped to this range)
- Do not learn the recurrent weights: ignore the problem (see next slide)
- Use a more advanced architecture (see next slides)

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Ignoring the problem: Echo State Networks

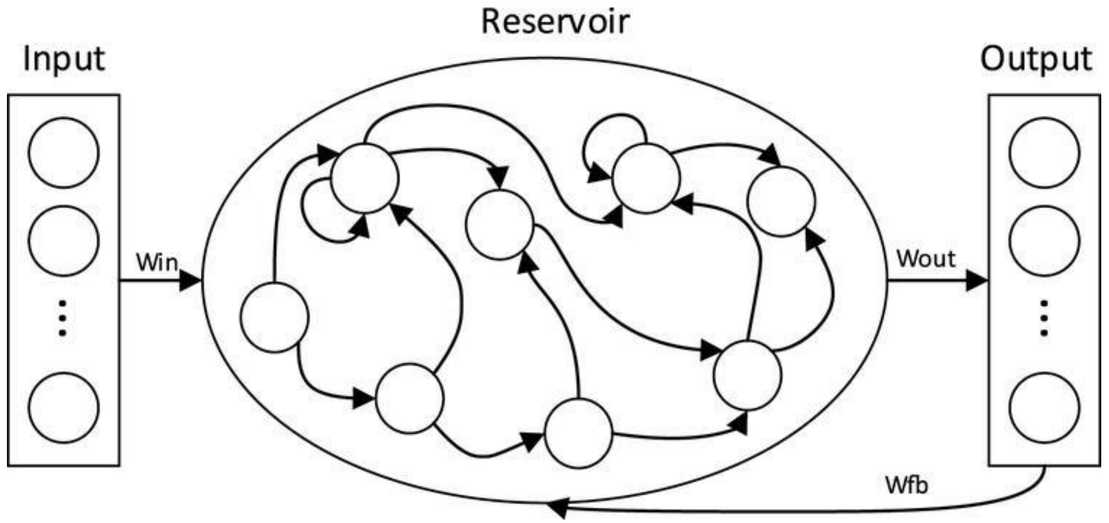
- We unroll the computational graph when we want to train recurrent connections.
- By **NOT** training the recurrent connections, we do not have the problem of vanishing/exploding gradients.

Echo State Networks:

- **Reservoir computing**: large sparsely connected hidden layer (1% connectivity) with connectivity and weights fixed and randomly assigned.
- Input-to-hidden, and output-to-hidden connections also random
- Only connections from hidden layer to output are trainable
- Regression problems: output is a **linear combination** of the hidden state

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Echo State Networks: how they work

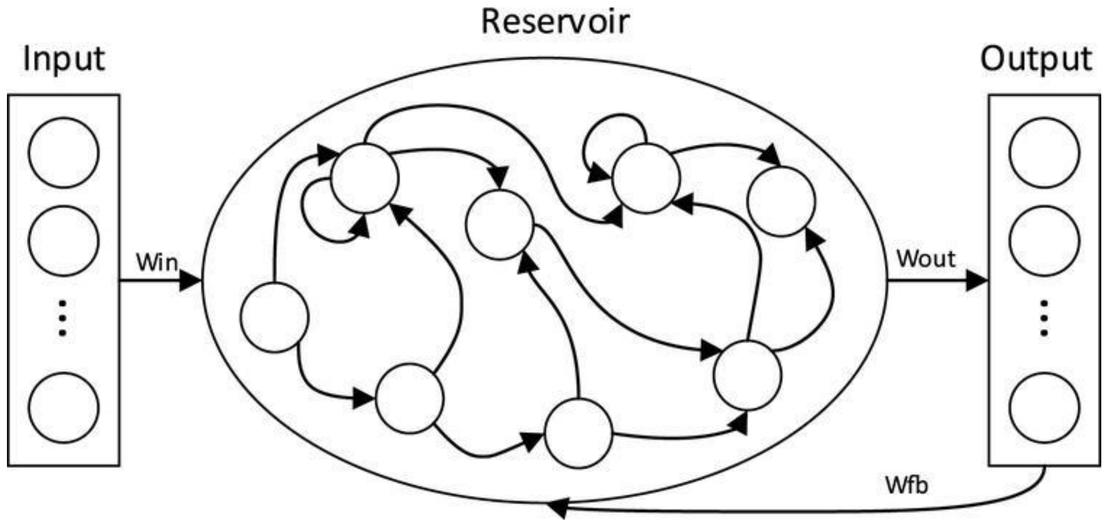


Input at each time step is fed into the dynamical reservoir, which along with its internal recurrent connections and recurrent connections from the output layer calculates its next state.

The output is computed as a linear combination of the current state of the reservoir and the output weights.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Echo State Networks: how to train them



The hidden units act as features that are computed from the sequence.

- Can be seen as a type of feature engineering

We can use linear regression to train the output weights!

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Echo State Networks

- Good performance in many signal processing applications, and time series prediction tasks.
- Easy to implement
- Consider them when the problem is not too complex and when cheap, fast, and adaptive training is desired

However, for tasks that have many variables and long-term memory (e.g., in natural language processing), they would need a reservoir of an excessive size.

The advent of autodifferentiation libraries, and more stable recurrent architectures (see next slide) enable the training of RNNs

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Mitigating the problem: Long Short-Term Memory

LSTM: Special type of RNN capable of handling long-term dependencies

Introduces special **gating** mechanisms that enable to learn:

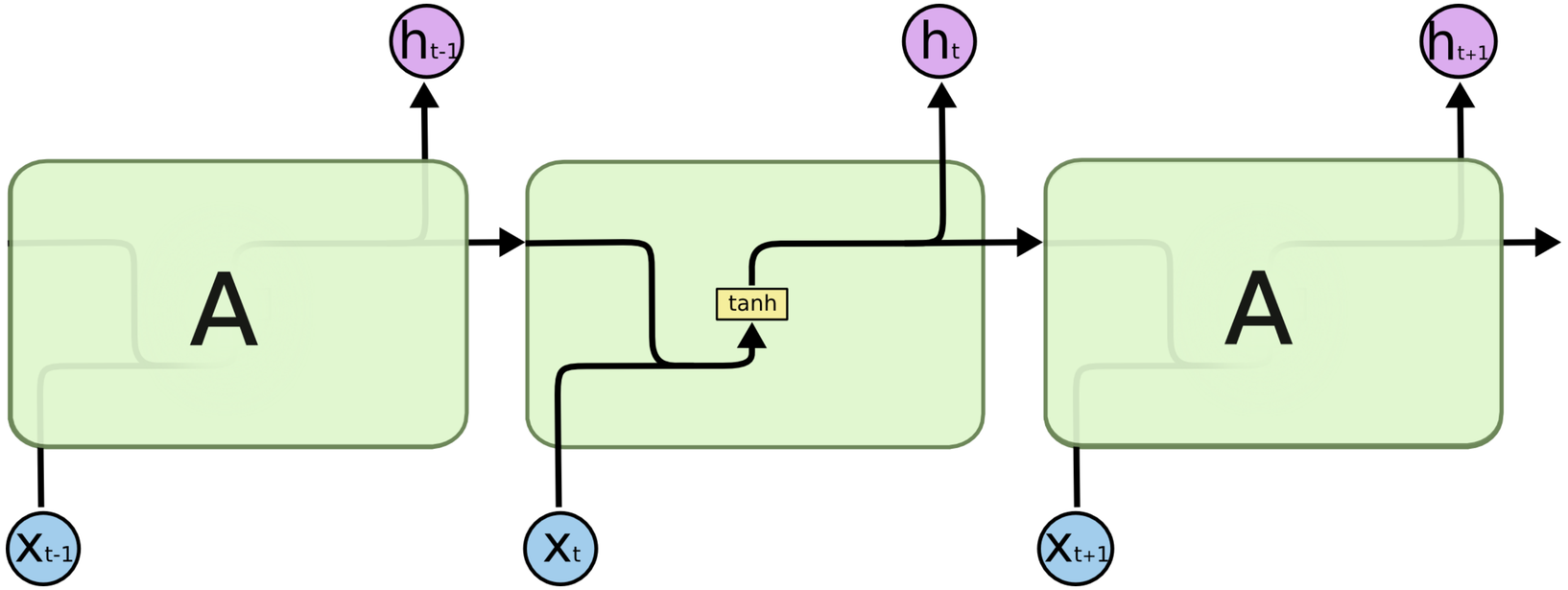
- what information to **forget**,
- what new information to **store in memory** and
- what information to **output**

How it achieves gating:

- feed a signal through a **sigmoid** operation: produces a value in $[0,1]$
- **multiply** the result with another signal x: makes the result in $[0, x]$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

A recurrent tanh layer

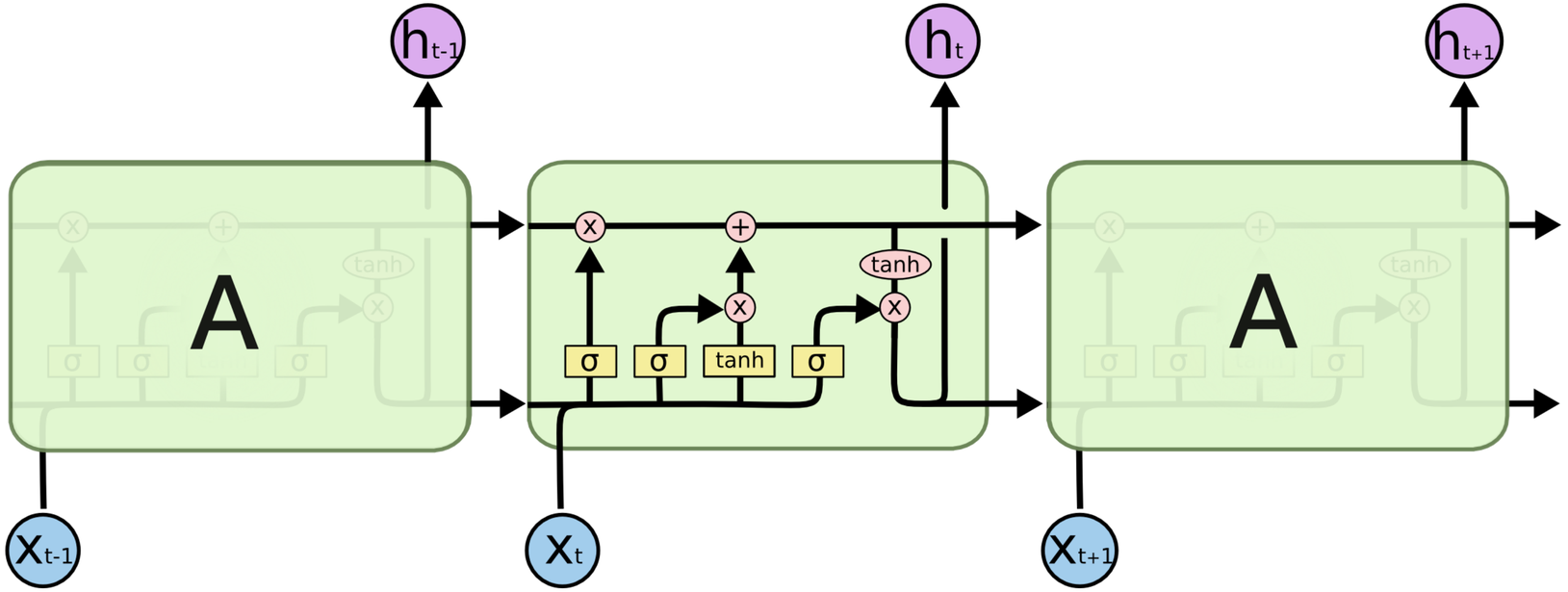


Source

A hidden layer with recurrent connections that uses the tanh activation function takes as input the input vector at time t ($\boldsymbol{x_t}$) and the hidden state at time t-1 ($\boldsymbol{h_{t-1}}$), and outputs the hidden state at time t ($\boldsymbol{h_t}$) which can be used in another layer (on top) or propagated forward in time (on the right).

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

LSTM layers



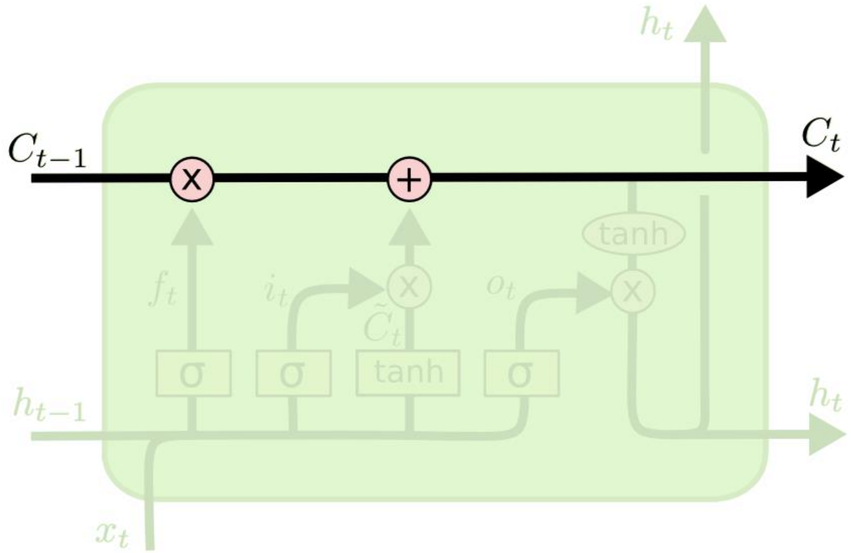
Source

An LSTM cell outputs two states: (1) the cell state which is propagated forward in time, and (2) the hidden state which can be used in other layers on top or propagated forward in time.

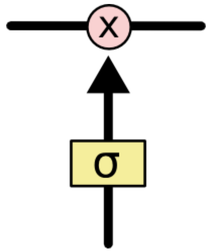
LSTM: has 3 gates to protect and control cell state

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

LSTM layers



Source



Cell state runs with only some minor **linear** interactions. It is propagated forward in time.

Gates: way to optionally let information through

Composed of sigmoid layer and pointwise multiplication operation

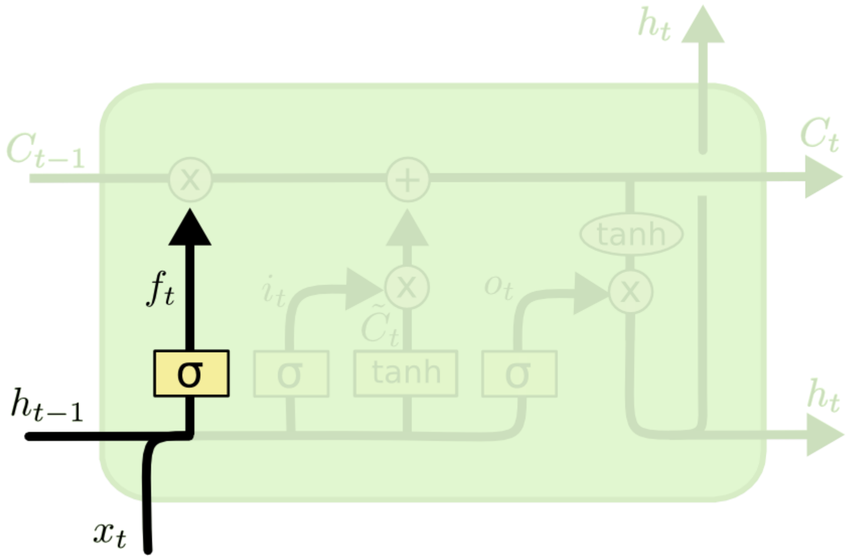
Sigmoid output: [0,1]

Value of 0: let nothing through

Value of 1: let everything through

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

LSTM layers



Forget gate: what information to throw away

Outputs [0,1]

1: completely keep this

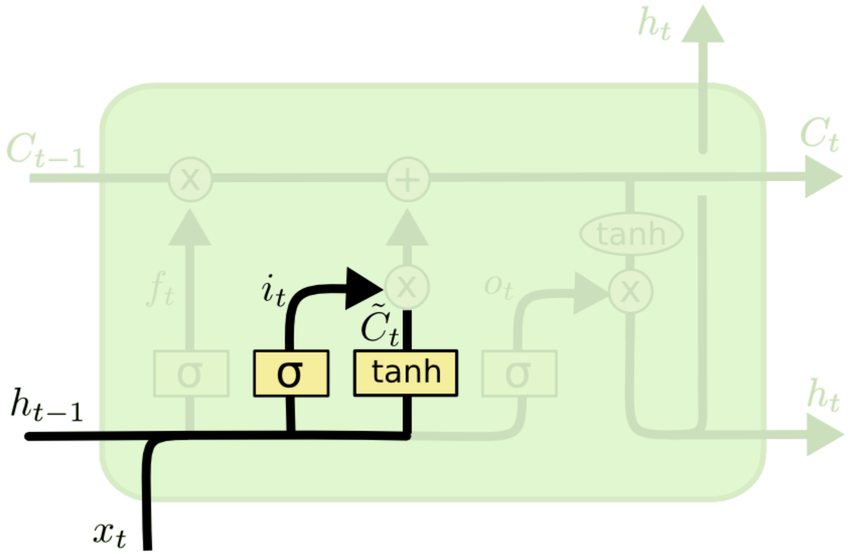
0: completely forget this

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

LSTM layers



Source

Input gate: what information to store in the cell state

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

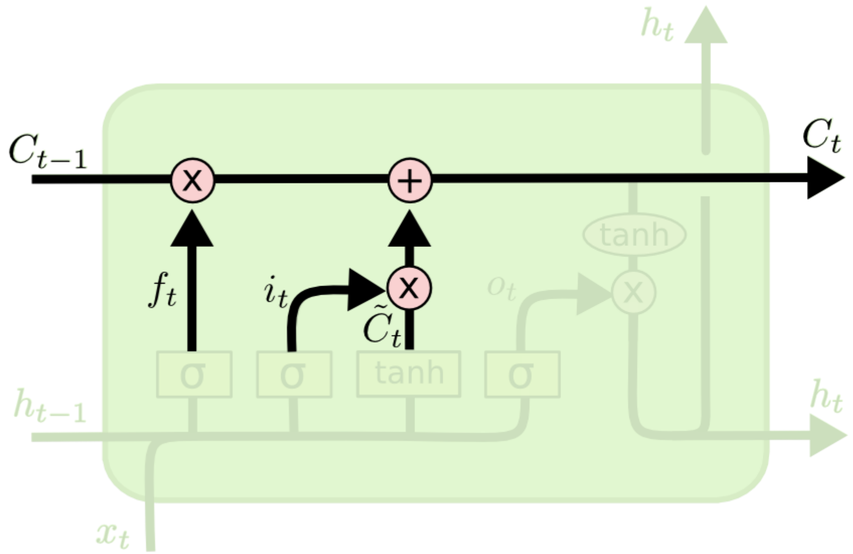
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Vector of new candidate values that could be added to the state: pushed through tanh to become in [-1,1]

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

LSTM layers

Update the old cell state into the new state



Controls what to forget

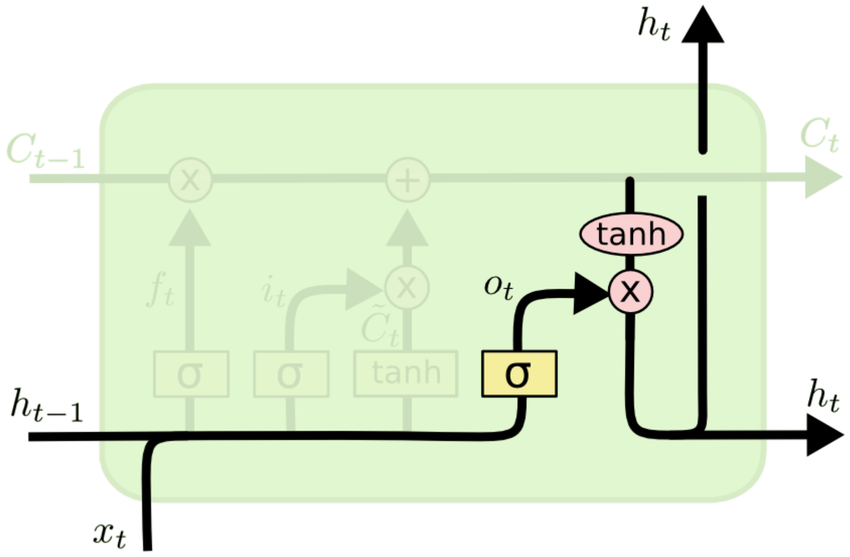
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Controls what to add

[Source]

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

LSTM layers



Source

Output gate: decide what part of the cell state to output

$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Hidden state: cell state through tanh to become in [-1,1] gated by the output gate

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

LSTM applications

- Robot control
- Time series prediction
- Speech recognition
- Rhythm learning
- Music composition
- Grammar learning
- Handwriting recognition
- Human action recognition
- Drug design
- …

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Text Data

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Word representation

V = [a, aaron, …, zulu]

$|V| = 10{,}000$

1-hot encoding

Man (5391)	Woman (9853)	King (4914)	Queen (7157)	Apple (456)	Orange (6257)
$\begin{bmatrix}0\\0\\0\\0\\\vdots\\1\\\vdots\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\0\\0\\\vdots\\1\\\vdots\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\0\\\vdots\\1\\\vdots\\0\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\0\\0\\\vdots\\1\\\vdots\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\\vdots\\1\\\vdots\\0\\0\\0\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\0\\0\\\vdots\\1\\\vdots\\0\\0\end{bmatrix}$

Weakness: Does not generalize well across words!

I want a glass of orange _____
I want a glass of apple _____

Distance between any pair of these words will be the same

Adapted from: Andrew Ng, Deep Learning course, Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Word embeddings

- Convert one-hot representation into a featurized one, i.e., a numerical vector representation in a predefined N-dimensional space
  - This transformation is learned
  - Each word has a unique vector representation

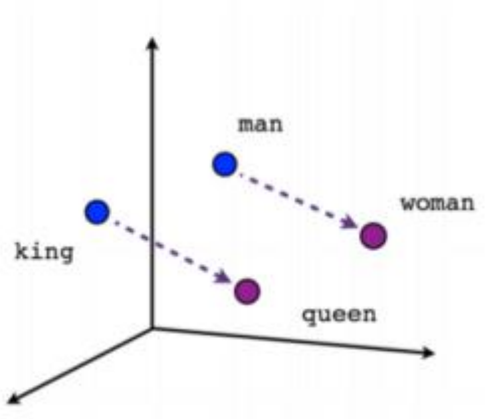
- Word embeddings try to capture the semantic, contextual and syntactic meaning of each word based on the usage of these words in sentences
  - Words that have similar semantic and contextual meaning also have similar vector representations

Popular models: Word2Vec, Glove, ELMo

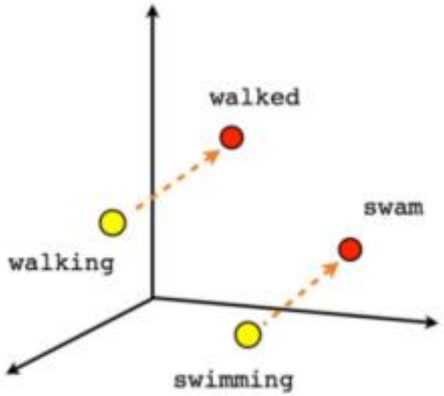
This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Properties of word embeddings

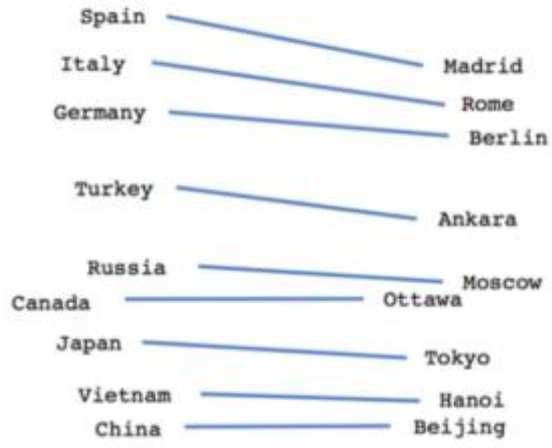
Word2Vec["King"] - Word2Vec["man"] + Word2Vec["woman"] = Word2Vec["Queen"]



Male-Female



Verb tense



Country-Capital

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Applications

- Named entity recognition
- Sentiment classification
- Document classification
- Analyzing survey responses
- Information retrieval
- Language Translation
- Spam detection
- Recommender systems
- …

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 12: Clustering

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 12: Clustering

Learning Outcomes

You will learn about:

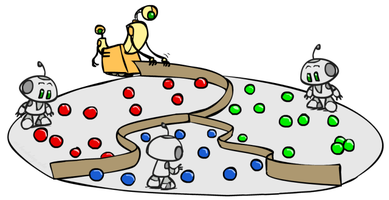
1. The problem of clustering in unsupervised learning and its difference between supervised classification
2. The k-means clustering algorithm: how it works, and how to tune it
3. How clustering can help supervised learning

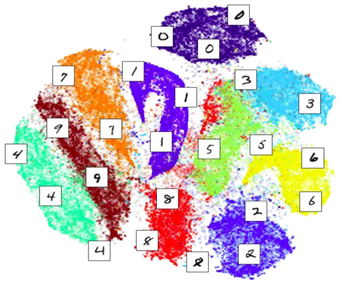
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Unsupervised Learning

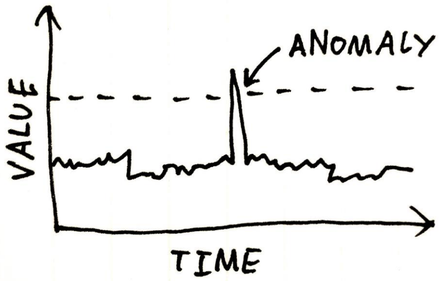
We present an input to the system, but we have no "teacher" to provide any "target" output

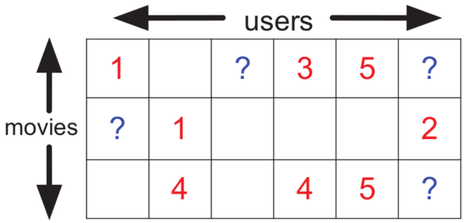




Common problems:

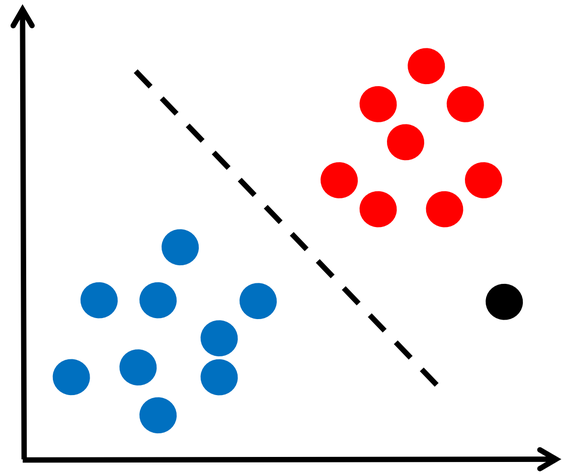
1. Clustering
2. Dimensionality reduction
3. Anomaly detection
4. Matrix completion (e.g., recommender systems)





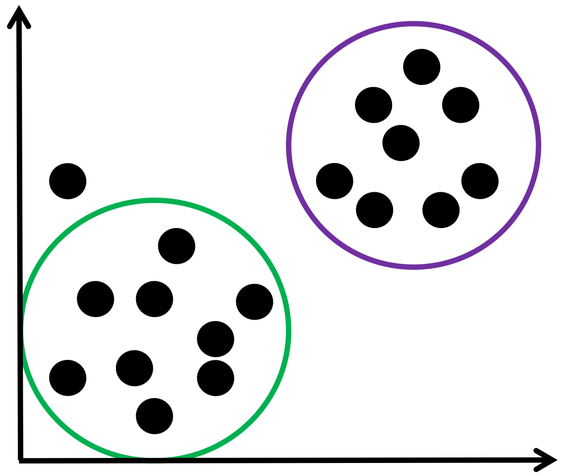
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Supervised Classification vs Clustering



Supervised Binary Classification

Training set: { $(x^{(1)},y^{(1)})$, ... , $(x^{(m)},y^{(m)})$ }



Clustering

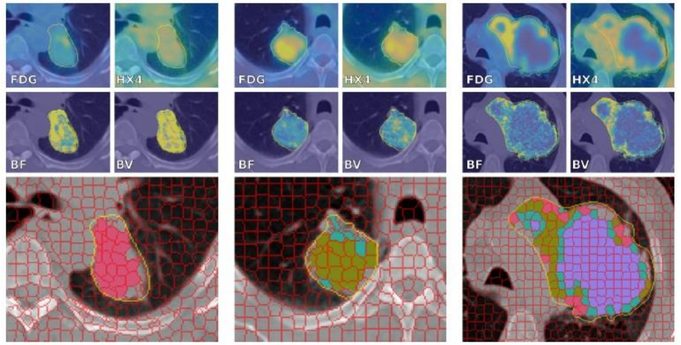
Training set: { $x^{(1)}$, $x^{(2)}$, ... , $x^{(m)}$ }

Co-financed by the European Union
Connecting Europe Facility

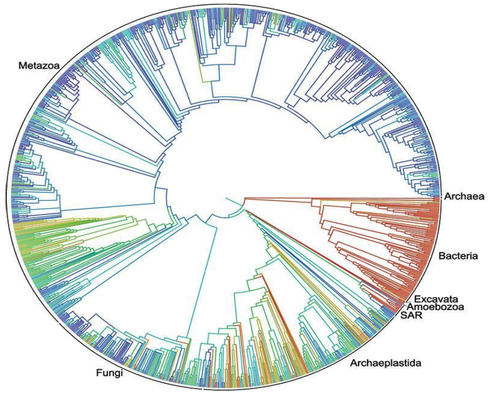
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Applications

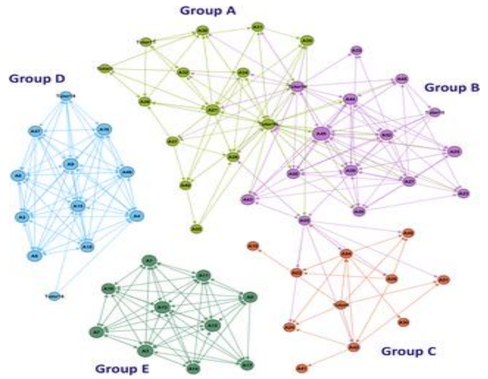
Market segmentation



Medical imaging



Animal ecology



Social Network Analysis

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

K-means clustering algorithm

Input: Training set: $\{ x^{(1)}, x^{(2)}, \dots, x^{(m)} \}$, $x^{(i)} \in R^n$
Number of clusters K

Randomly initialize K cluster centroids $\mu_1, \mu_2, \dots, \mu_K \in R^n$

$$min_k \left\| x^{(i)} - \mu_k \right\|^2$$

Repeat:

Assign each training point to a cluster centroid

for i = 1 to m

c(i) = index (1 to K) of cluster centroid closest to $x^{(i)}$

j=2

c(3)=2, c(6)=2, c(7)=2

Update the cluster centoids

for j = 1 to K

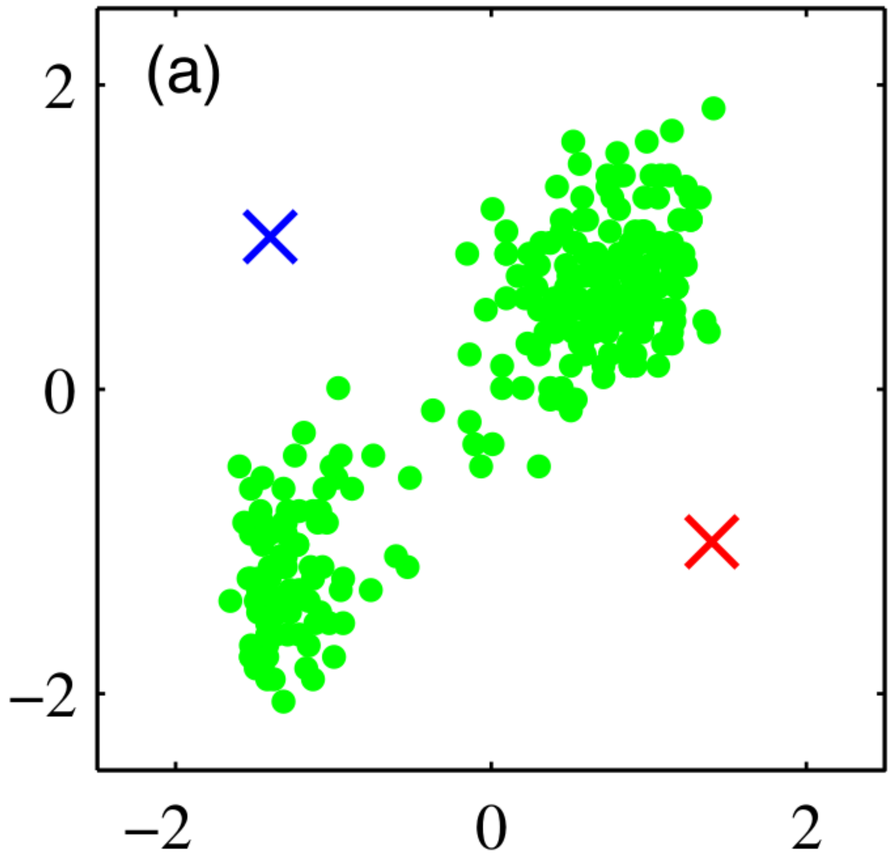
μ_j = mean of points assigned to cluster j

$\mu_2 = 1/3 \, (x^{(3)}+x^{(6)}+x^{(7)})$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

K-means clustering algorithm

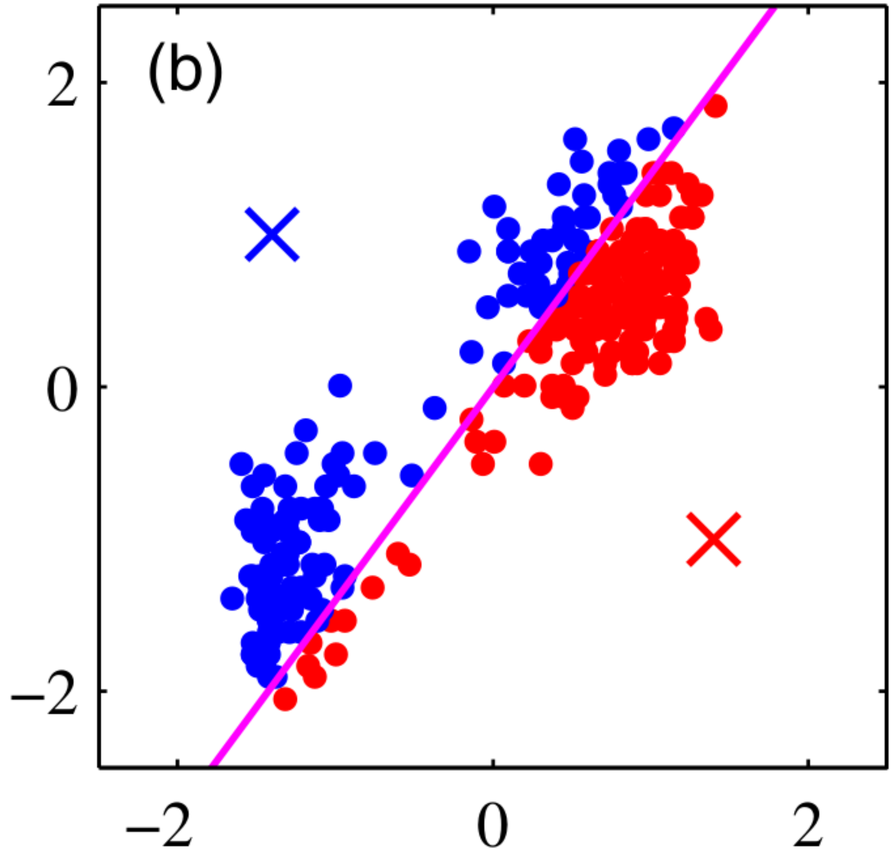


source: Bishop 2006

Randomly initialize
K=2 cluster centroids

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

K-means clustering algorithm



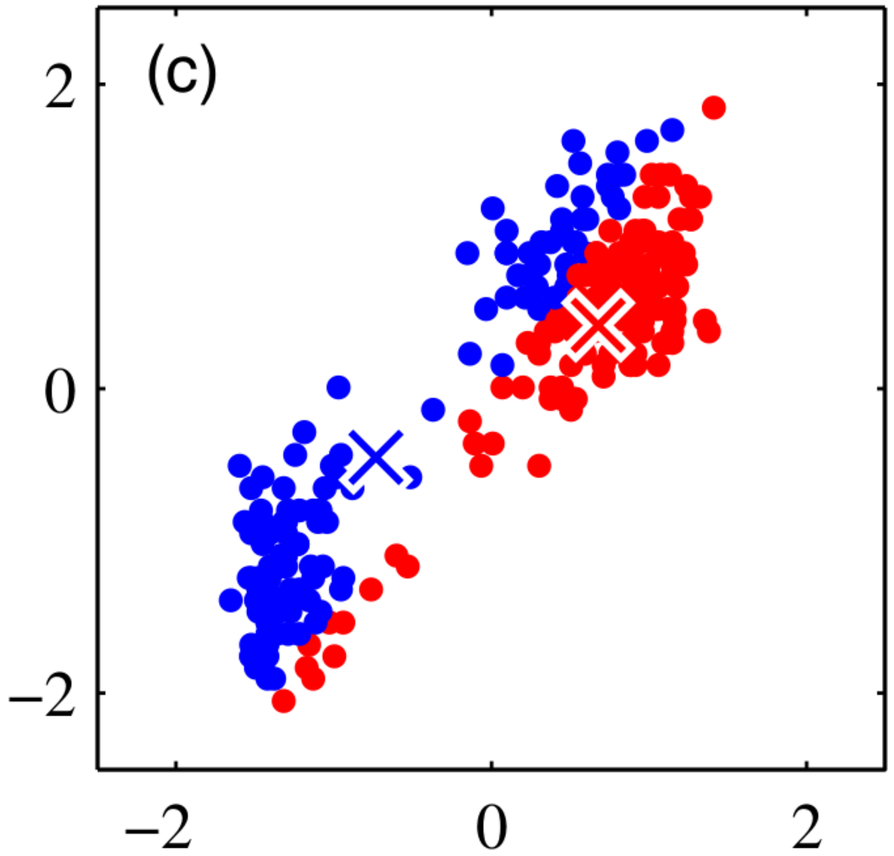
source: Bishop 2006

Iteration: 1

Assignment step

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

K-means clustering algorithm



source: Bishop 2006

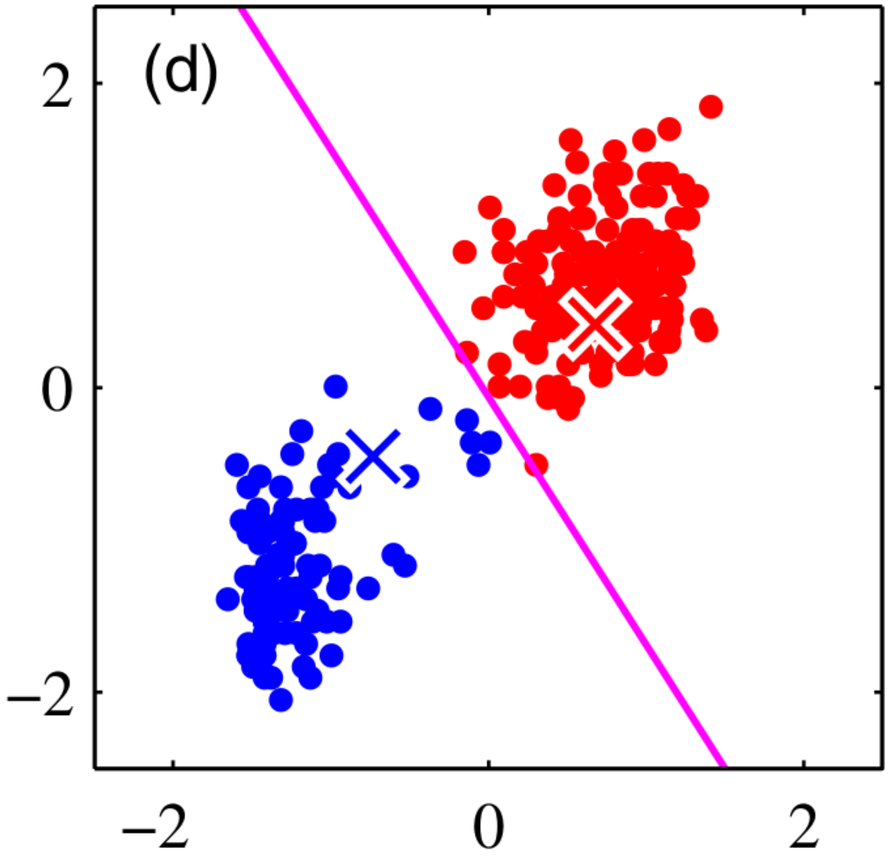
Iteration: 1

Update step

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

K-means clustering algorithm



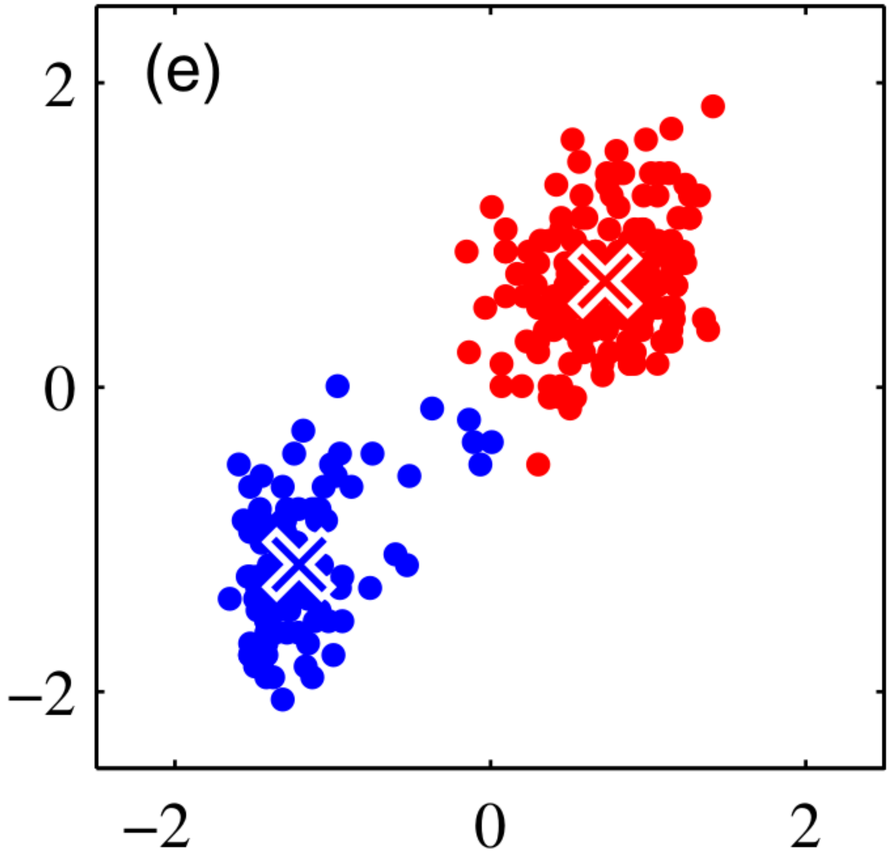
source: Bishop 2006

Iteration: 2

Assignment step

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

K-means clustering algorithm



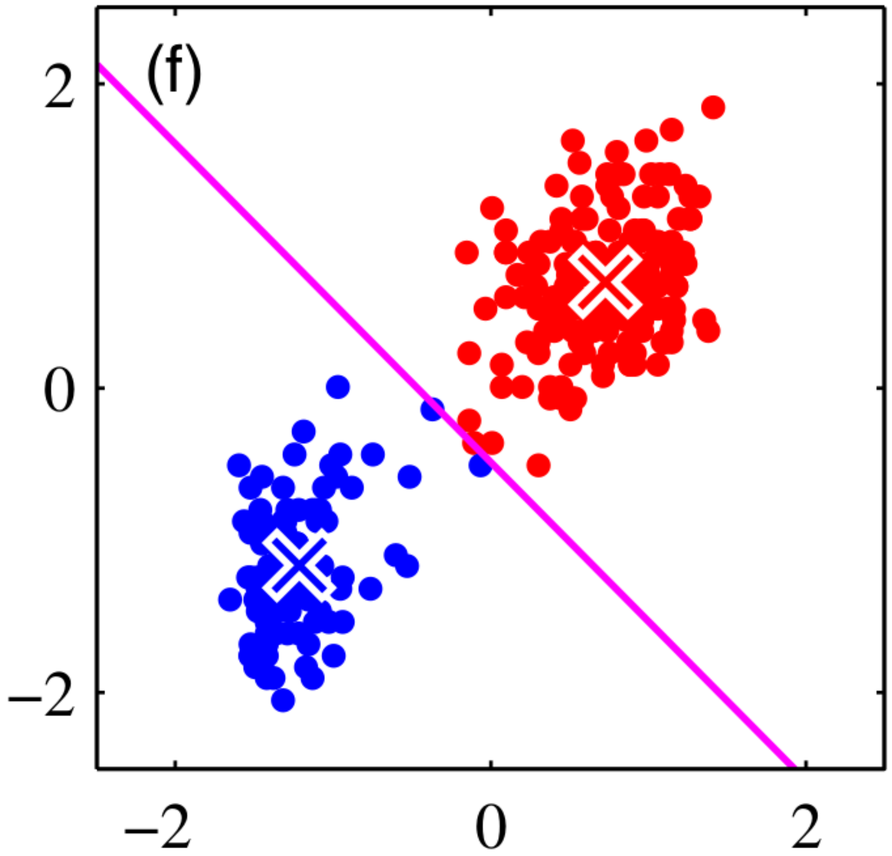
source: Bishop 2006

Iteration: 2

Update step

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

K-means clustering algorithm



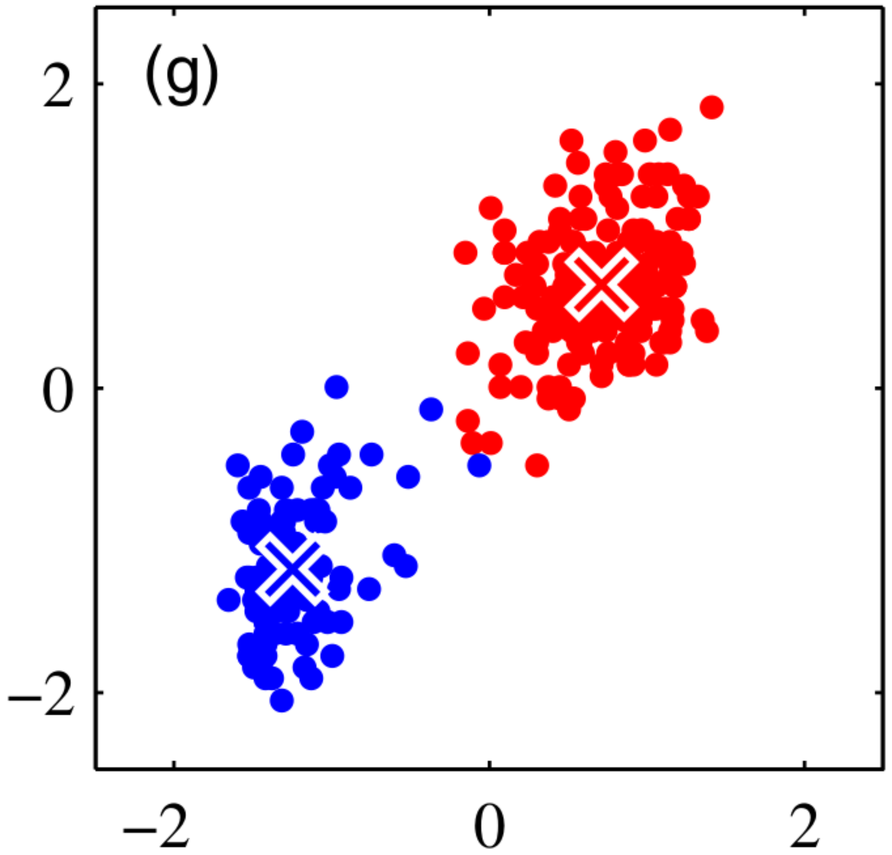
source: Bishop 2006

Iteration: 3

Assignment step

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

K-means clustering algorithm



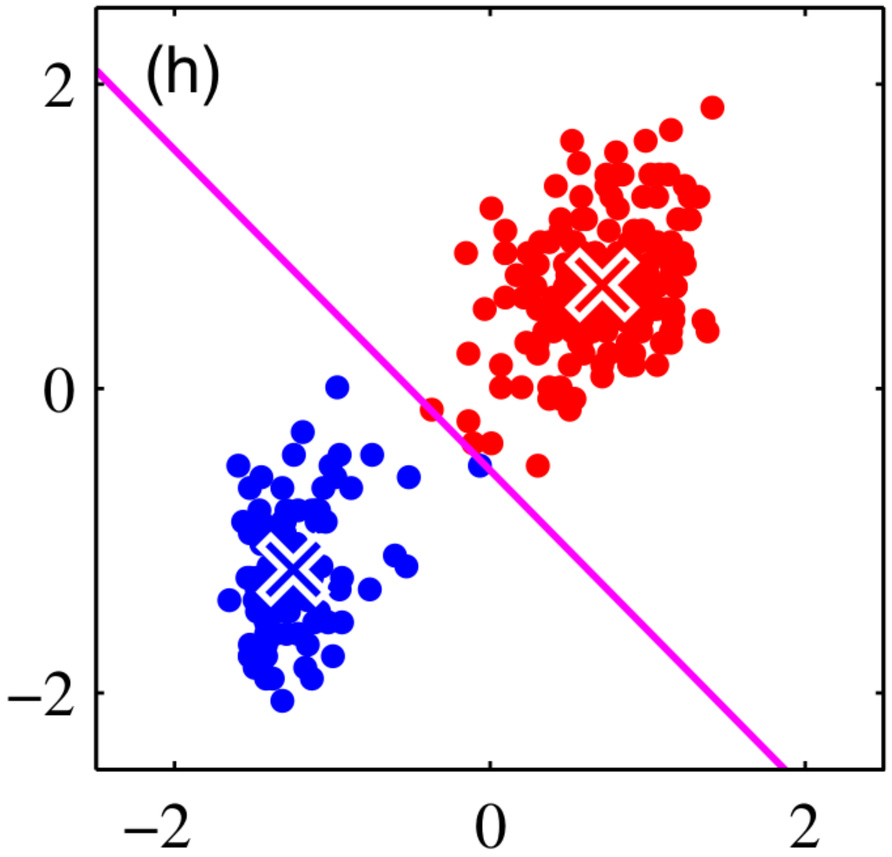
source: Bishop 2006

Iteration: 3

Update step

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

K-means clustering algorithm



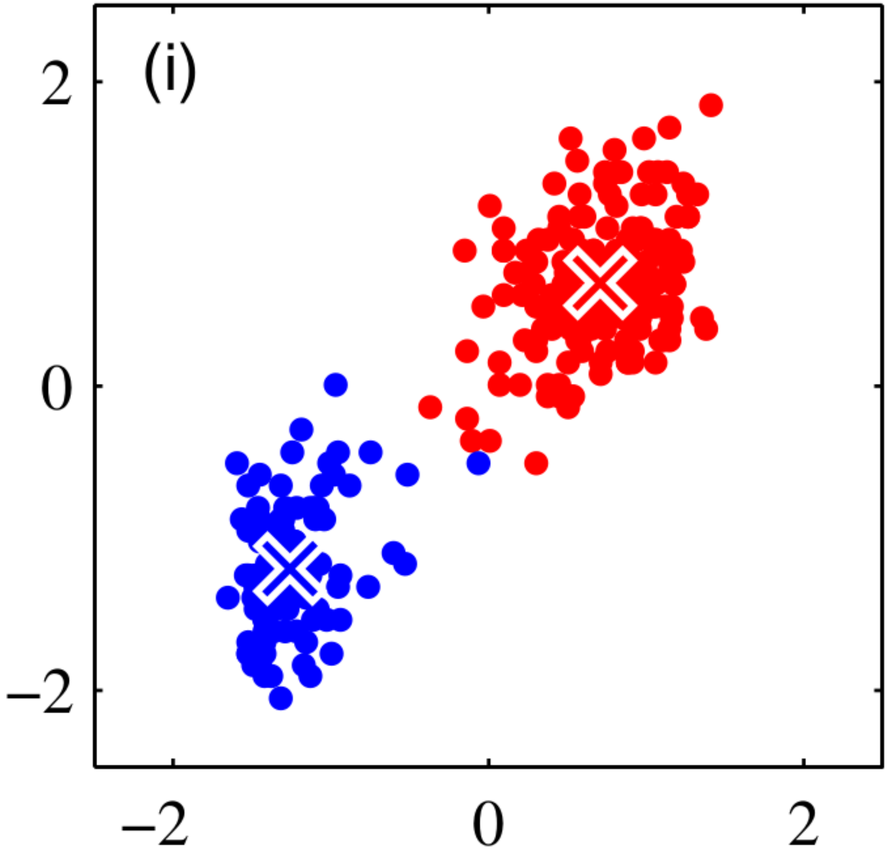
source: Bishop 2006

Iteration: 4

Assignment step

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

K-means clustering algorithm



source: Bishop 2006

Iteration: 4

Update step

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Random initialization

Instead of random initial centroids, randomly pick K training instances

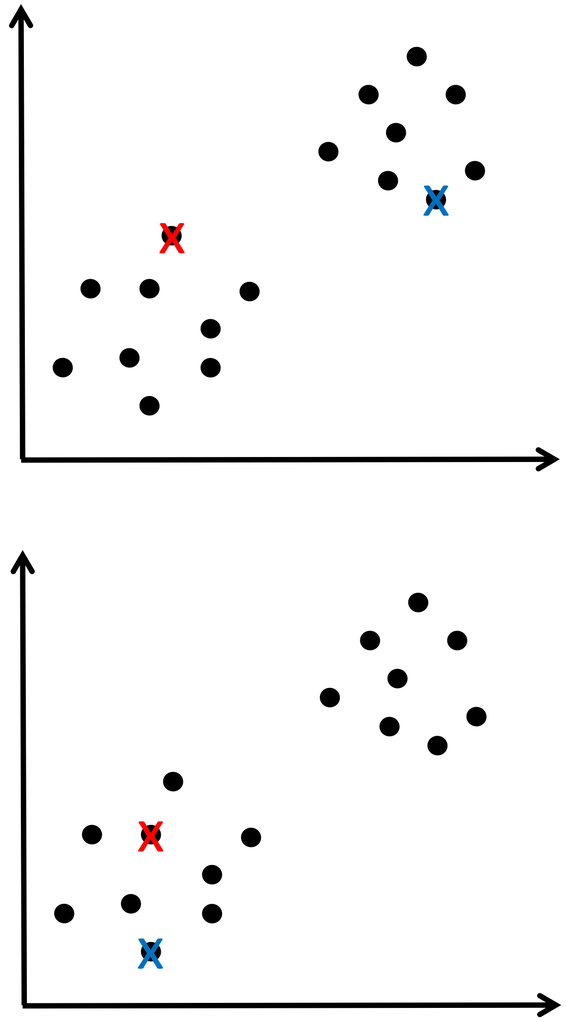
Set $\mu_1, \mu_2, \ldots, \mu_K$ equal to these K points

Should have K < m

K=2

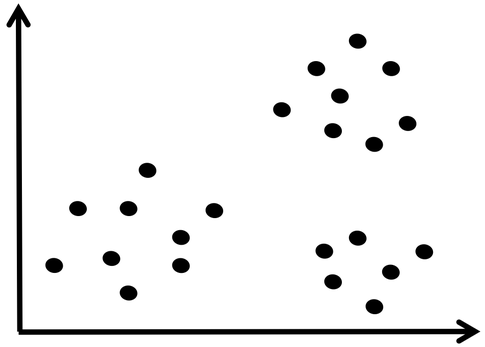
$\mu_1 = x^{(i)}$

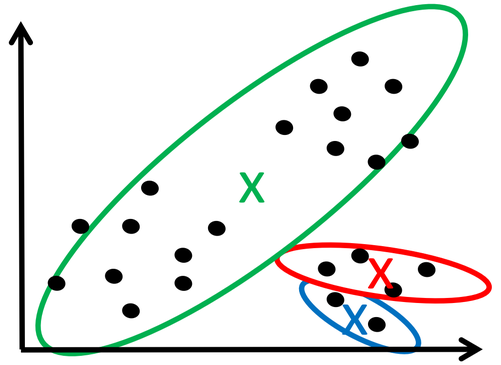
$\mu_2 = x^{(j)}$



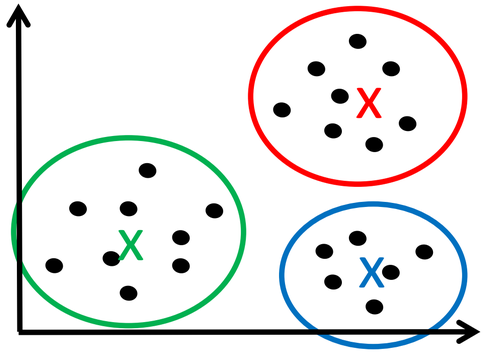
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Random initialization

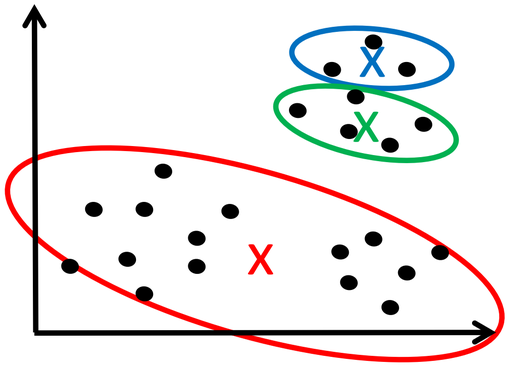




Local Optima



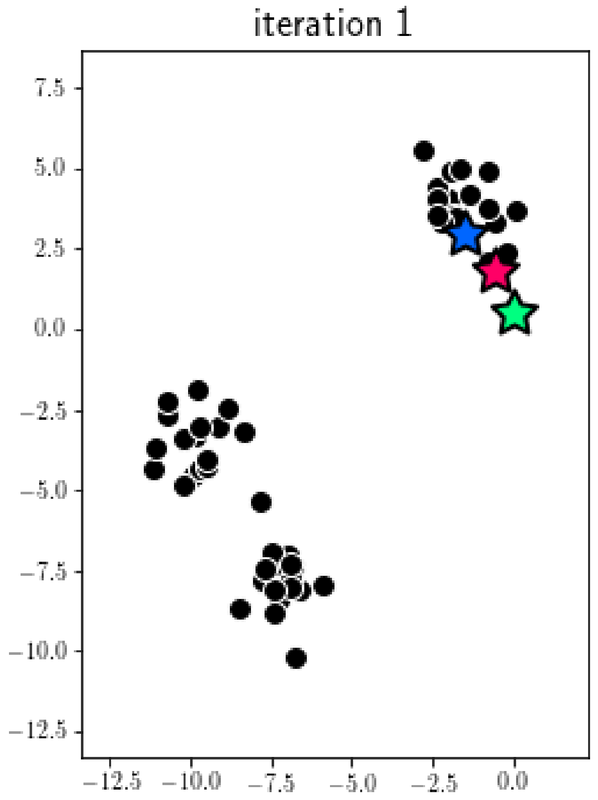
Global Optimum



Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

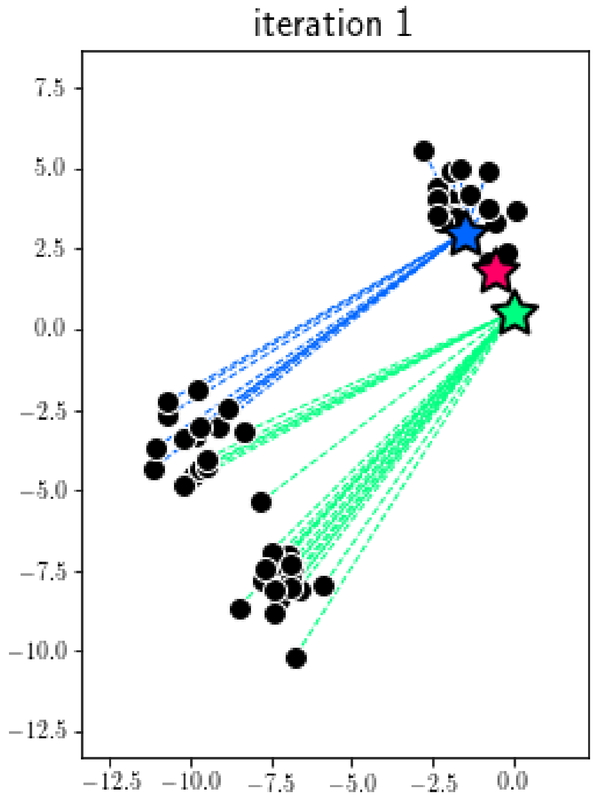
Empty clusters



source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Empty clusters

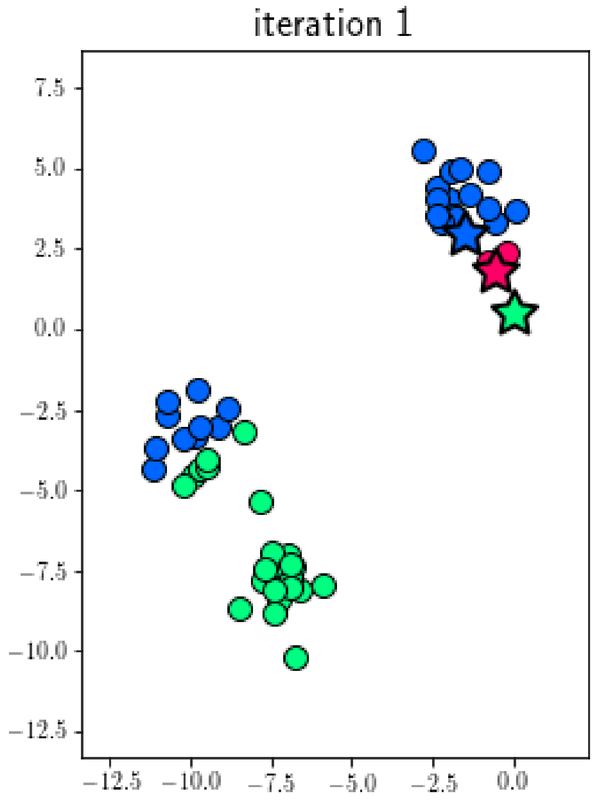


source

Assignment step

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Empty clusters

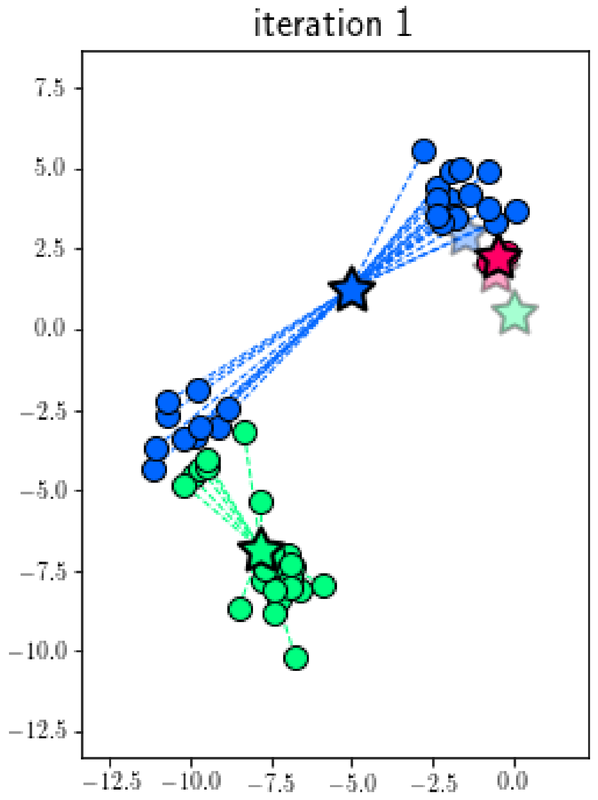


source

Assignment step

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Empty clusters

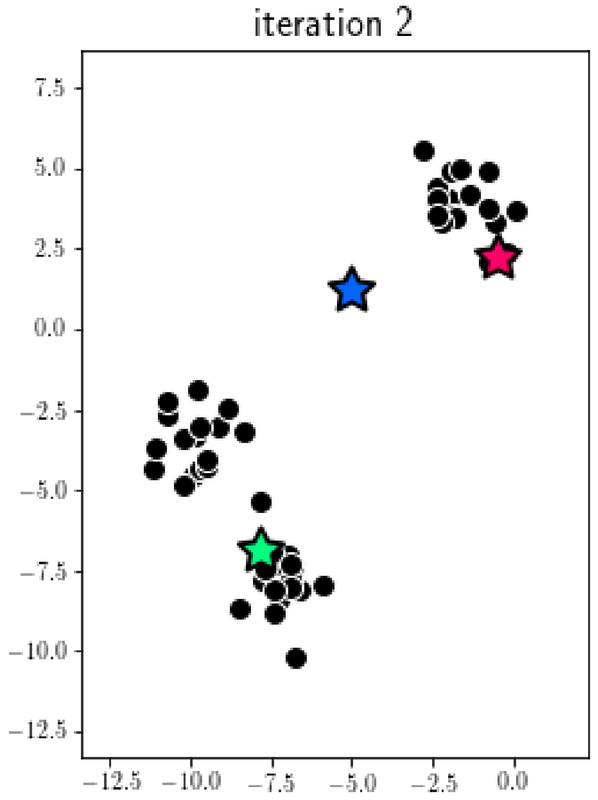


source

Update step

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

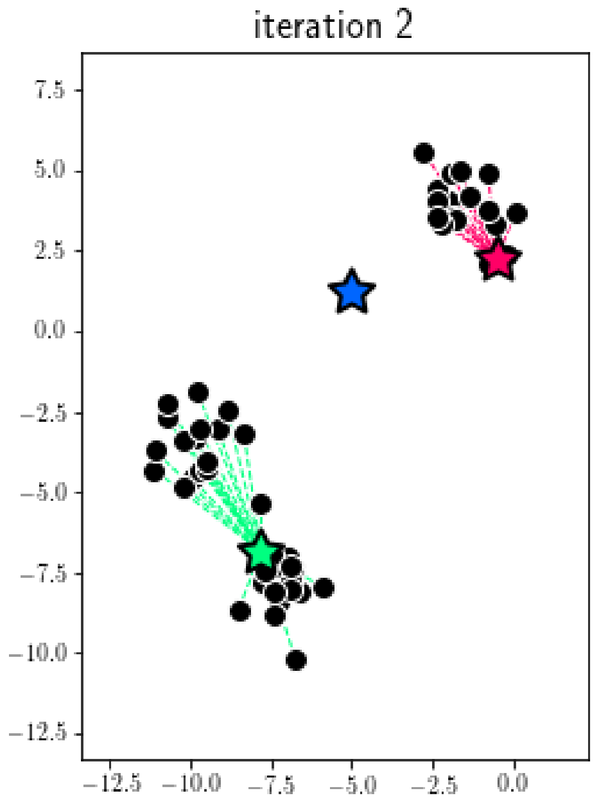
Empty clusters



source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Empty clusters



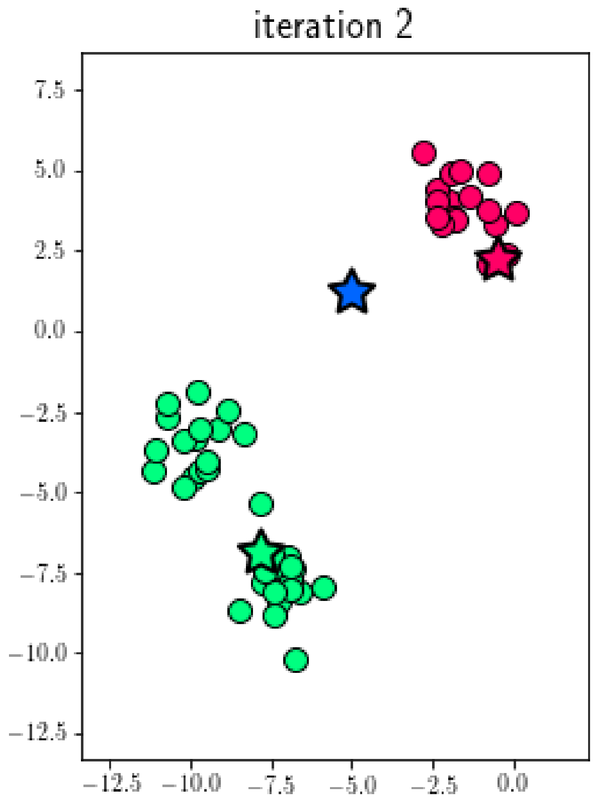
source

Assignment step

No points are close to the blue centroid

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Empty clusters



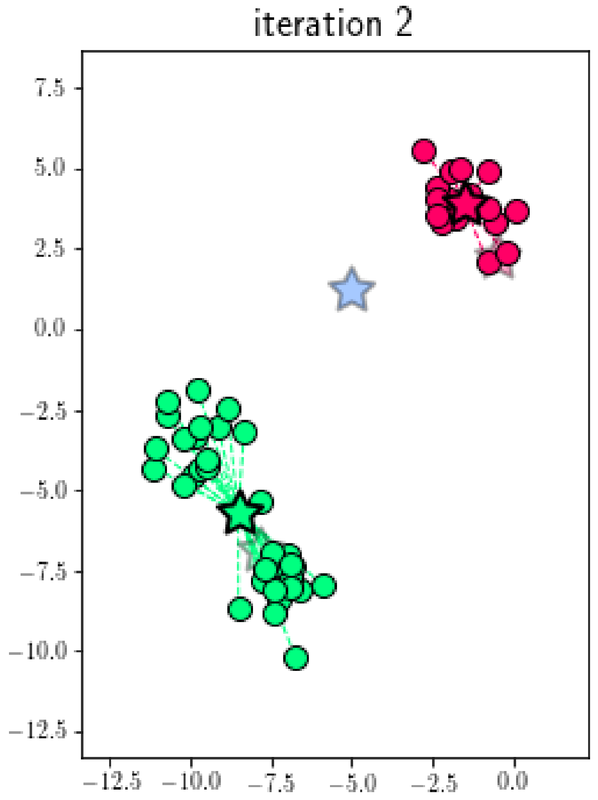
source

Assignment step

The blue centroid ends up without any points assigned to it

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Empty clusters

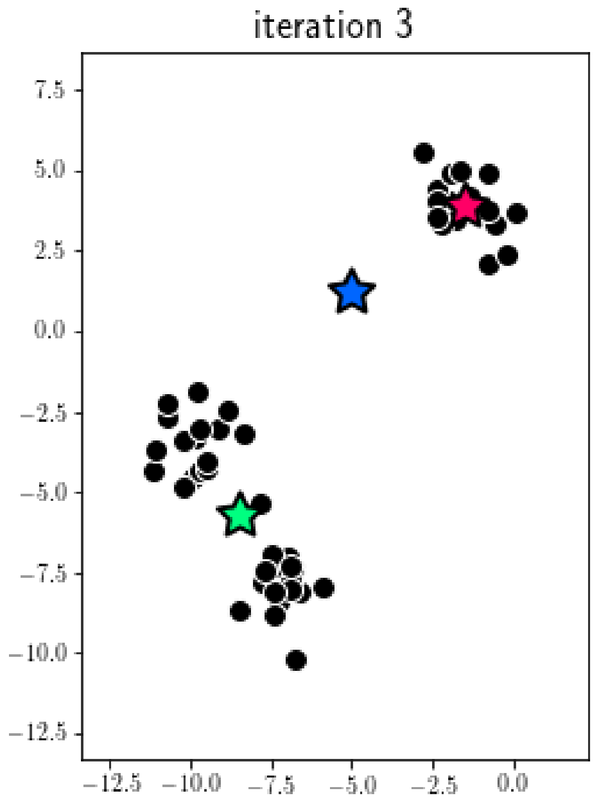


source

Update step

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

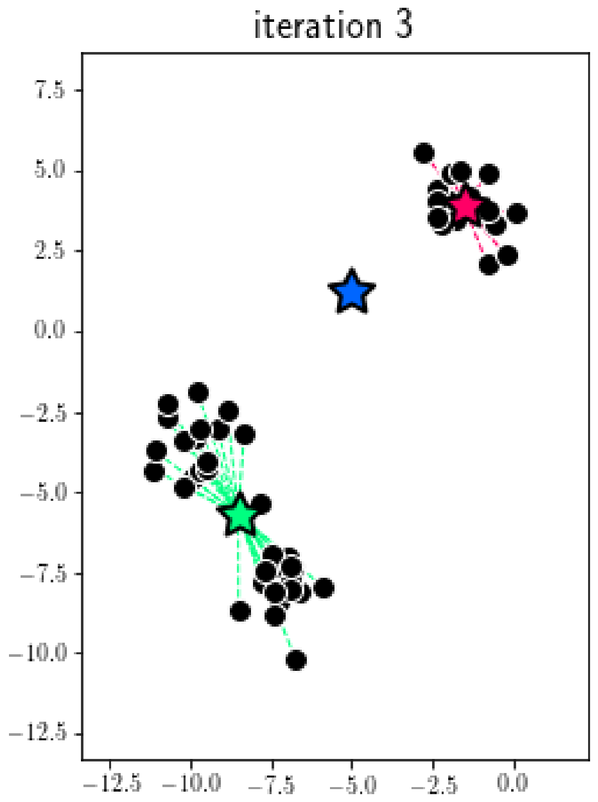
Empty clusters



source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Empty clusters

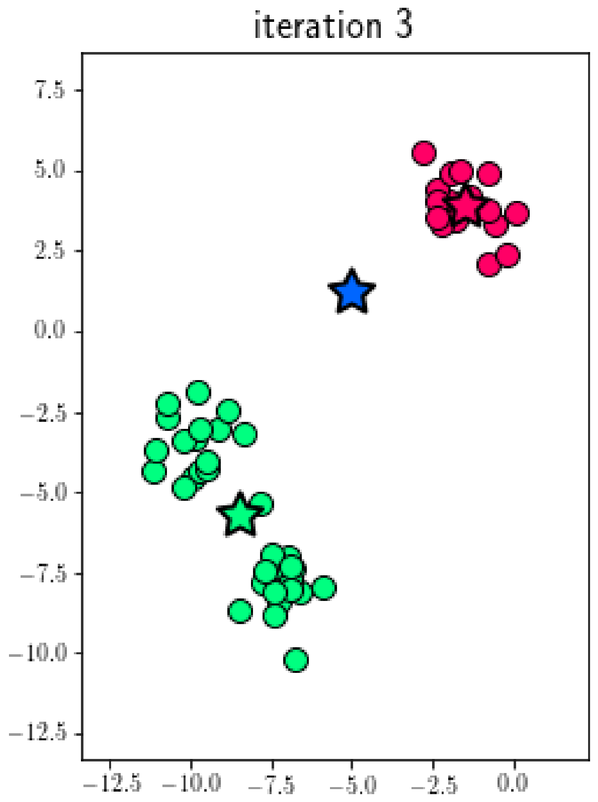


source

Assignment step

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Empty clusters

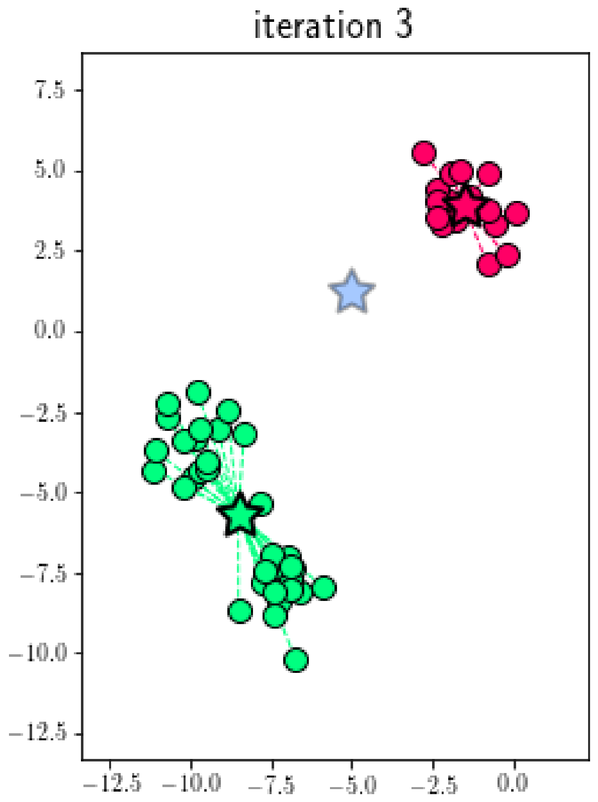


source

Assignment step

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Empty clusters



source

Update step

The blue centroid will never be assigned any points.

Potential remedy: when discovering empty clusters, reduce K by 1, or restart from a different random initialization

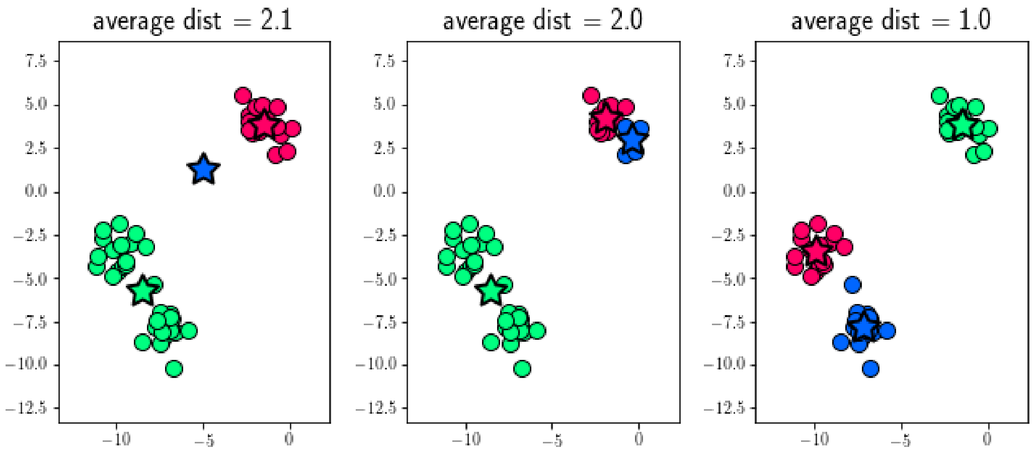
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Avoiding local optima

Run K-means multiple times:

for i = 1 to N
  Randomly initialize K-means
  Run K-means. Get $\boldsymbol{\theta} = c(1),...,c(m),\mu_1,...,\mu_K$
  Compute Cost Function: $L(\boldsymbol{\theta}) = L(c(1),...,c(m),\mu_1,...,\mu_K)$

Pick clustering with the lowest cost $L(\boldsymbol{\theta})$



What is the cost function?

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Cost function

Notation

μ_j = cluster centroid j ($\mu_j \in R^n$)

c(i) = index of cluster (1 to K) to which $x^{(i)}$ is assigned

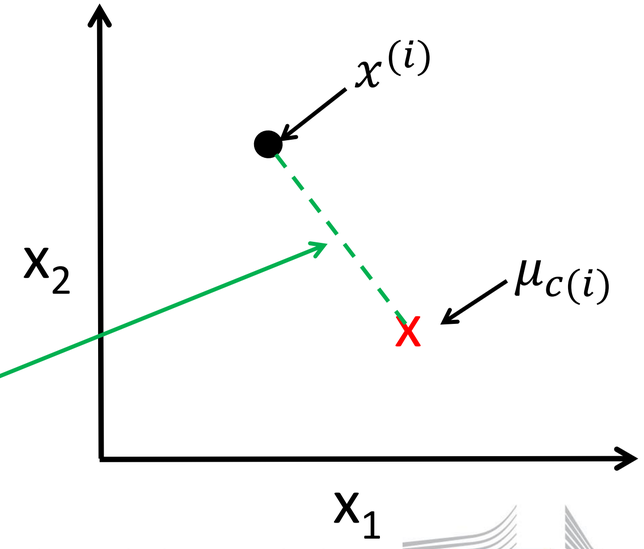
$\mu_{c(i)}$ = cluster centoid of cluster to which $x^{(i)}$ is assigned

Example

$x^{(i)}$ assigned to cluster 3

c(i) = 3

$\mu_{c(i)} = \mu_3$



Objective $\underset{\theta}{minimize}\ L(\boldsymbol{\theta})$

Average Intra-cluster distance

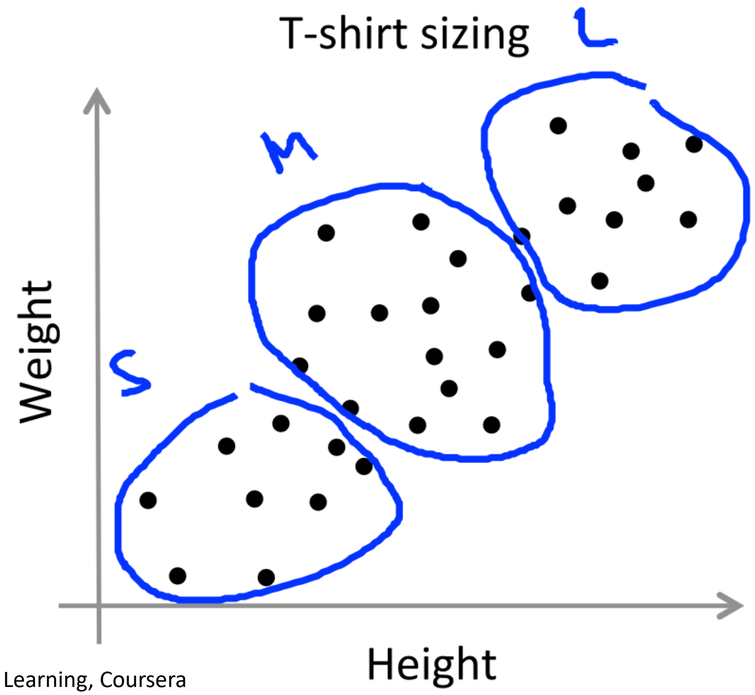
$$L(\boldsymbol{\theta}) = L(c(1),\ldots,c(m),\mu_1,\ldots,\mu_K) = \frac{1}{m}\sum_{i=1}^{m} || x^{(i)} - \mu_{c(i)} ||^2$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

How to choose K

Domain knowledge

T-shirt sizing: K=3



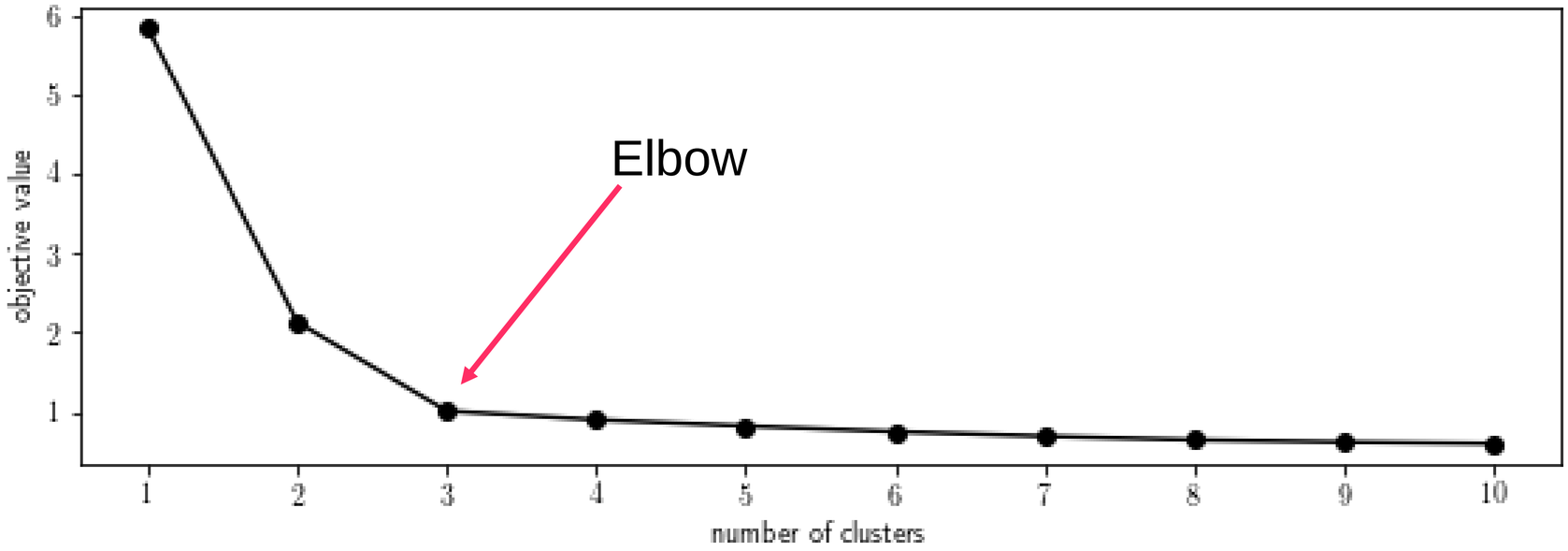
source: Andrew Ng - Machine Learning, Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

How to choose K

“Elbow” method



K = 3

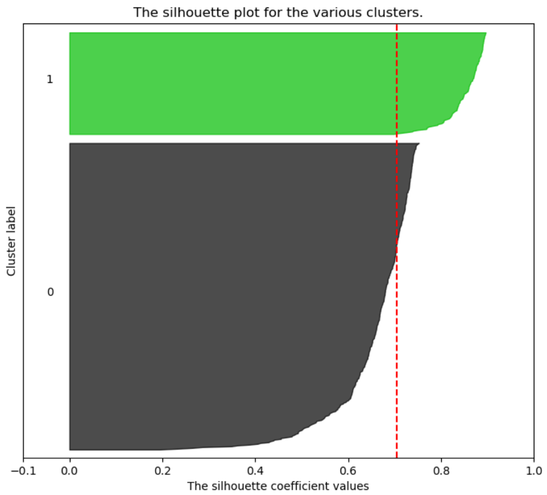
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

How to choose K

- Uniform thickness
- No negative values
- All beyond avg

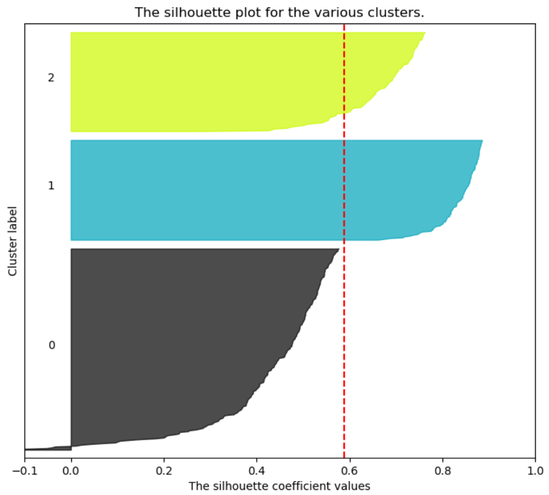
Silhouette Score

K=2



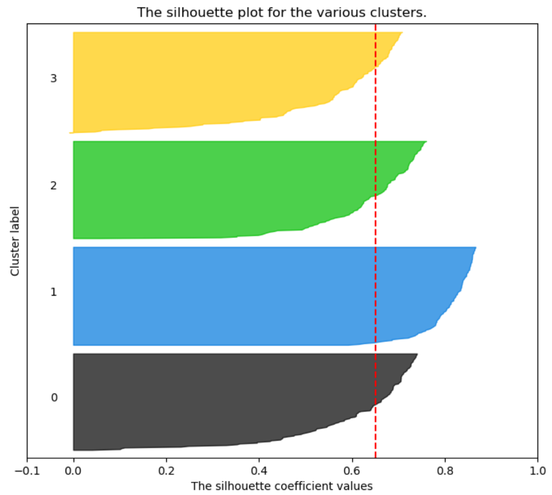
0.705

K=3



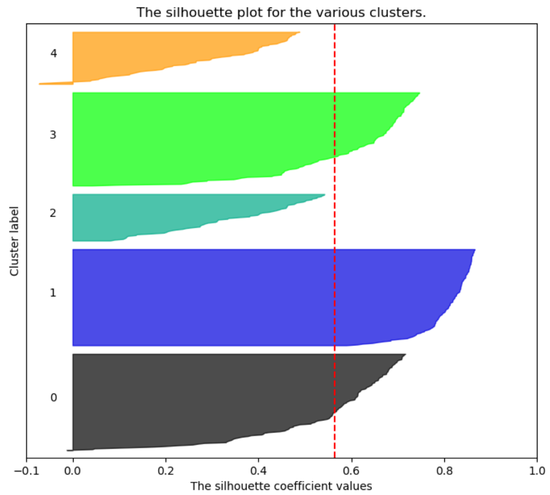
0.588

K=4



0.651

K=5



0.564

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

K-medoids algorithm

- Similar to k-means, by attempting to minimize the distance between points to be in the cluster and points designated as the center of the cluster

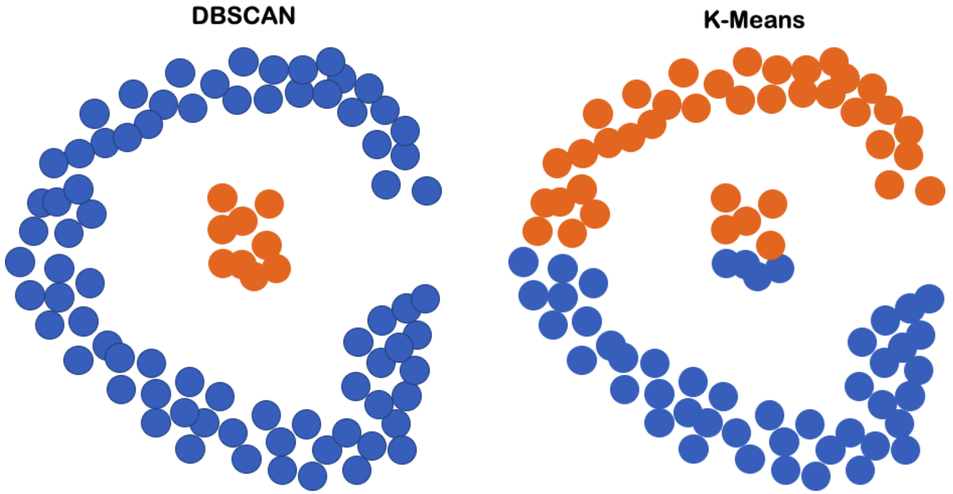
- Difference:
  - k-medoids chooses actual data points as centers
  - It minimizes the sum of pairwise dissimilarities instead of sum of squared Euclidean distances

- This makes it more robust to noise and outliers than k-means

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

DBSCAN

- Density-based clustering algorithm
- Does not need to specify the number of clusters
- It finds it automatically based on the density of the points
- Need to define the maximum distance between two samples for one to be considered as int the neighborhood of the other



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

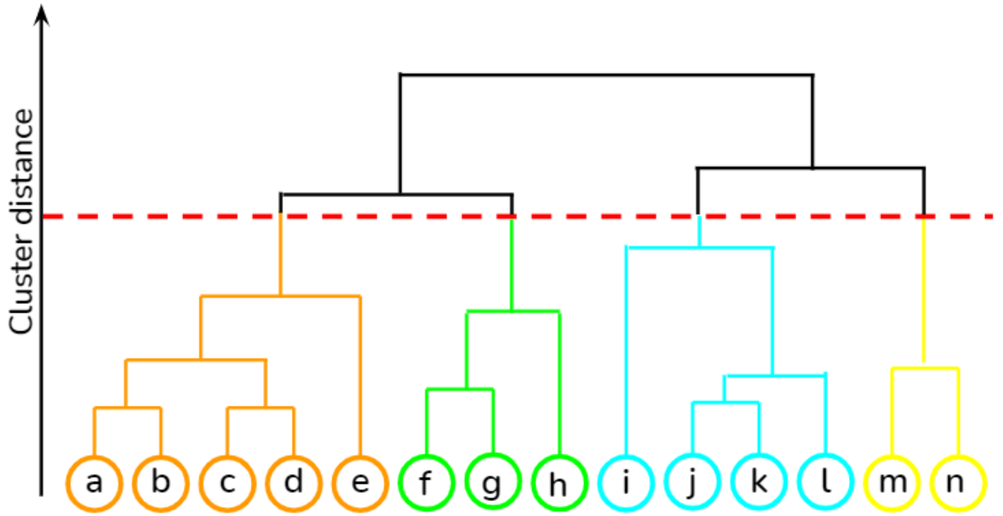
Hierarchical Clustering

Used when we want to build a hierarchy of clusters, and do not have knowledge about the number of clusters

Two categories:

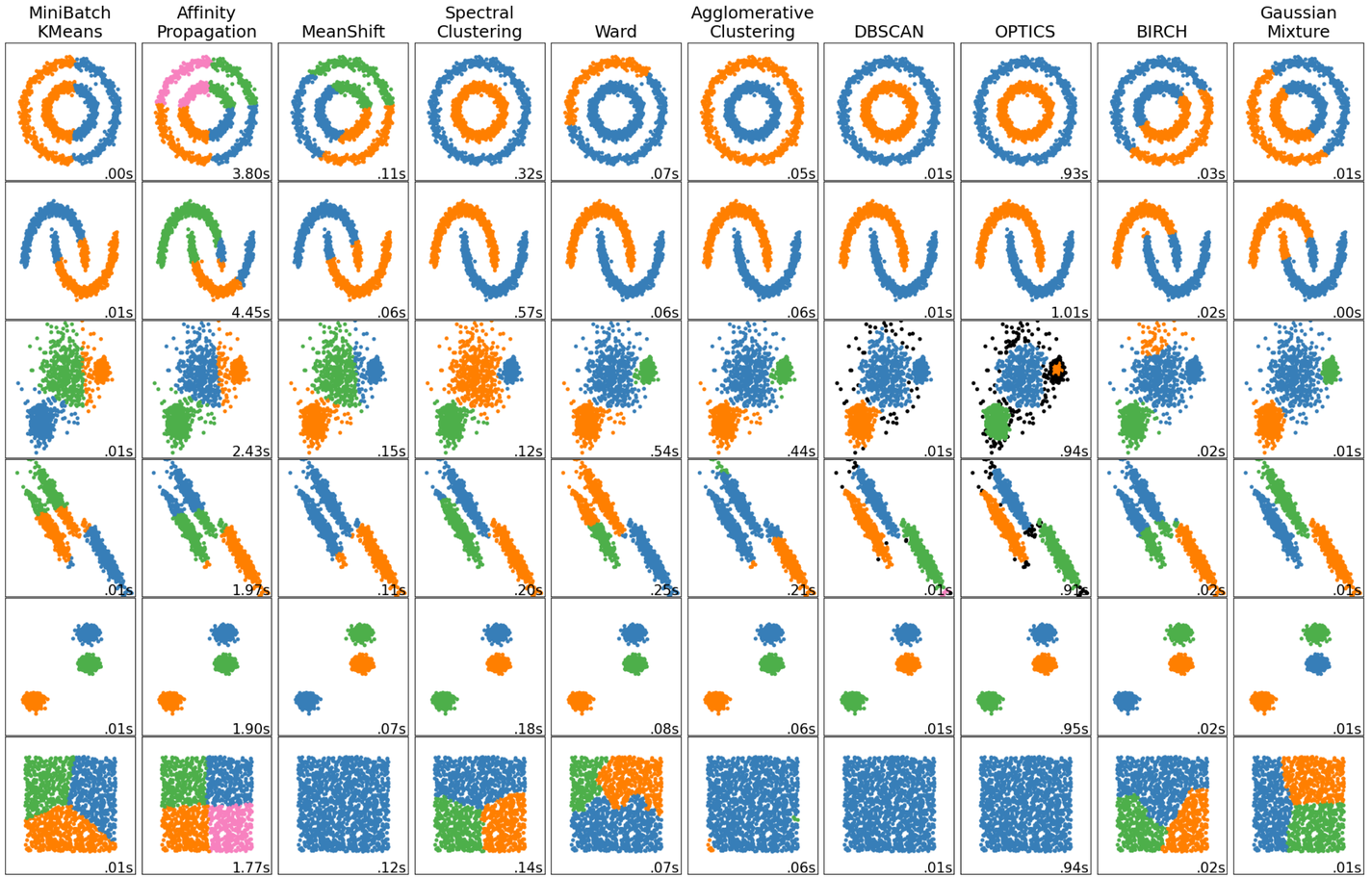
- Agglomerative clustering: bottom-up approach
  - Each point starts in its own cluster and pairs of clusters are merged as we move up the hierarchy
- Divisive clustering: top-down approach
  - All points start in one cluster and splits are performed recursively as we move down the hierarchy

Output: a tree-shaped structure known as a dendrogram



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Clustering in scikit-learn



K-means in scikit-learn uses a better default initialization strategy (k-means++)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

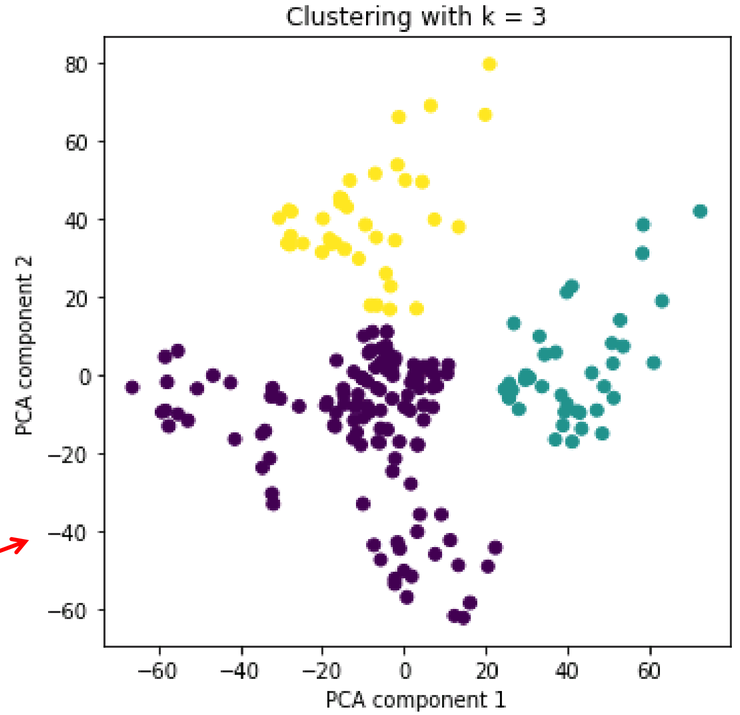
Clustering high-dimensional data

High-dimensional data: dozen to thousands of dimensions
Examples: images, text documents

Problem: As the dimensionality increases, data points become more sparse, thus, distance of the farthest point becomes closer to the distance of the closest point

Thus, distance metrics in clustering algorithms become meaningless

Mitigation: Reduce data dimensionality, then cluster
- Feature Selection
- Dimensionality reduction e.g., using PCA



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Clustering can help supervised learning

Finding the initial centres for RBF networks

- RBF networks with adjustable centres have an initialization phase where we choose the initial coordinates of the centres
- The algorithm then proceeds in adapting the centre coordinates using gradient descent (GD)

- **How to choose initial centres:**
  - **Approach 1**: Choose centres uniformly at random from data points
    - Simple, however, if centres not properly chosen, GD might get stuck in local optima
  - **Approach 2**: Use clustering
    - Results in better initial coordinates, which helps GD
    - Some clustering methods can even decide the optimal number of centres

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Clustering can help supervised learning

When there are more unlabeled examples than labelled ones

- Supervised learning works on labeled examples
  - $D_{labelled} = \{(x^{(i)}, y^{(i)})\}_{i=1:l}$
- Unlabeled examples, however, are often easier to collect
  - $D_{unlabelled} = \{(x^{(j)})\}_{j=1:u}$
- Usually $u >> l$
  - Example: for an animal recognition task, we might have abundant images for various animals, but only very few labels
- It is also often costly to label
  - Example: medical images, we need to employ experts to properly tag them

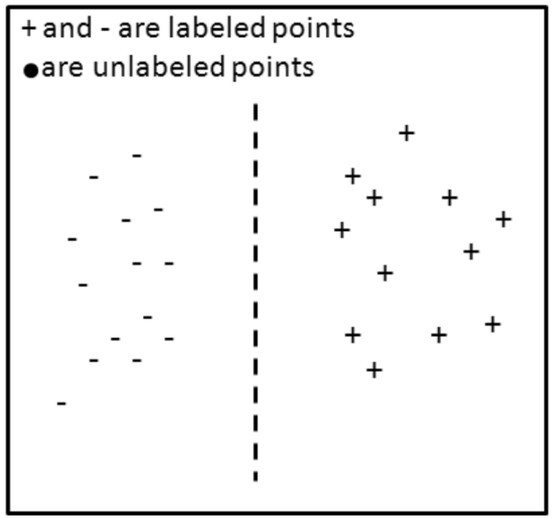
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Clustering can help supervised learning

When there are more unlabeled examples than labelled ones

If we consider only labeled examples:

$$D_{labelled}$$

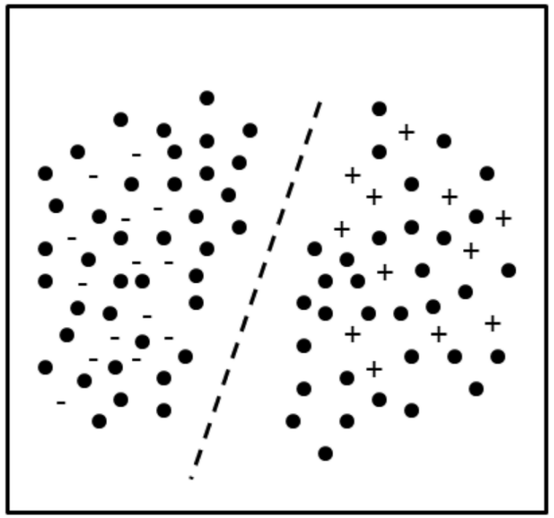


If we consider all data as training data:

$$D = D_{labelled} \cup D_{unlabelled}$$

Different decision boundary

Improved generalization performance



Images source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Next Lecture

- Dimensionality Reduction

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 13: Dimensionality Reduction

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Revision

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to Deep Learning

- Deep learning is about learning successive layers of representations
  - Using NNs with more than 2 hidden layers
- We have it now mostly because of hardware advancements (GPUs) and abundance of data
- When we want to detect a particular concept that could be in different places of the input we use weight sharing, i.e., we build a single feature detector for this concept by training the weights of these inputs jointly
  - This helps generalization
  - Example 1: creating a dog image classifier: a dog could appear everywhere in an image
  - Example 2: text completion network: we want the part of the NN that learns what a dog is to be reused every time the NN sees the word dog

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to Deep Learning

- Convolutional networks are NNs more suitable for image data
- They use local receptive fields (filters) and shift them (convolve) over the activation map of the previous layer to create the activation map of the current layer
  - This reduces the number of parameters compared to fully connected feedforward networks
  - Each filter becomes a feature detector over different parts of the input (translation invariance):
    - Layer 1: edge detectors
    - Layer 2: corner detectors
    - Layer 3: parts-of-objects detectors
    - Layer 4: complete-objects detectors

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to Deep Learning

- Sequential data: data ordered into sequences
  - Typically: time series data
- A way to handle sequential data using NNs is by using feedback (delayed) connections: recurrent NNs (RNNs)
  - A recurrent NN creates its own internal representation of time
- We can train RNNs using backpropagation through time
  - We unroll the RNN over time and backpropagate the errors from the last time step to the first
  - We accumulate the gradients and apply gradient descent
  - Unrolled RNNs can become very deep networks
  - When doing so we might have the vanishing or exploding gradients problems

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to Deep Learning

- Echo state networks:
  - use a large sparsely connected hidden layer which is not trained
  - they do not use backprop through time
  - they do not have the vanishing or exploding gradients problems
  - we can compute the analytic solution for a regression problem (similarly to linear regression)
  - good performance in tasks that require fast, adaptive training
  - not good performance in tasks with many variables and long-term dependencies

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to Deep Learning

- Long short-term memory networks:
  - Use gating mechanisms that allow the network to learn what to forget, what to store in memory and what to output
  - Gating: sigmoid multiplied by signal : sigmoid modulates how much of the signal is allowed to pass through
  - Can be applied to tasks requiring long-term dependencies
- Text data:
  - One-hot word representation: sparse, high dimensional, does not generalize well
  - Word embeddings: learned, numerical vector representation
    - Try to capture the meaning of words based on their usage in sentences
    - Words with similar meaning have similar vector representations
    - King – Man + Woman = Queen

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Clustering

- Clustering is the problem of grouping data with similar characteristics
  - We do not have labels that specify the correct outputs
- K-means clustering uses a pre-specified number (K) of clusters
  - Randomly initializes the K cluster centroids
  - Alternates between assigning each point to a centroid and updating the centroids
- We can avoid local optima by running K-means multiple times and selecting the clustering with the lowest cost
- We choose K using domain knowledge, the Elbow method or the Silhouette score
- Clustering can help supervised learning, e.g., by
  - Finding the initial centres of RBF networks
  - Allowing to use both labelled and unlabelled data to improve generalization

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 13: Dimensionality Reduction

Learning Outcomes

You will learn about:

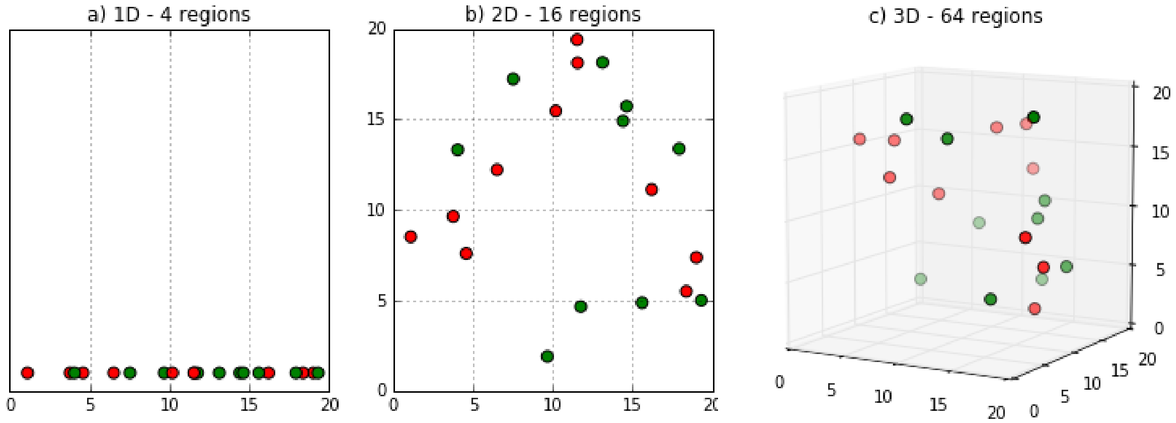
1. The problem of dimensionality reduction
2. The Principal Components Analysis (PCA) algorithm
3. Nonlinear dimensionality reduction using Kernel PCA and Autoencoders
4. Manifold Learning approaches
5. The t-distributed stochastic neighbor embedding (t-SNE) for visualizing high dimensional data

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

The Curse of Dimensionality

- Given the same number of samples, as the dimensionality of the data increases, they become exponentially sparser
- Causes data to be more easily separable



dense sparser even sparser

- However:
  - Easier to overfit on high-dimensional datasets
  - Need exponentially large number of samples to cover the space and avoid overfitting
  - Distance metrics do not make sense in high dimensions (e.g., algorithms like K-means clustering or K-Nearest Neighbors cannot deal with high dimensional data)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Dimensionality reduction motivation

- Find meaningful low-dimensional structures hidden in high-dimensional observations
  - The human brain confronts the same problem in perception:
    - $30{,}000$ auditory nerve fibers
    - 10^6 optic nerve fibers
    - ➢ Extract small number of perceptually relevant features
- Difficult to visualize data in dimensions greater than three

source

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

What is Dimensionality Reduction

- Transformation of data from a high-dimensional visible space, $\boldsymbol{x} \in \mathbb{R}^n$, to a low-dimensional latent space, $\boldsymbol{z} \in \mathbb{R}^k$, $k < n$
- Low-dimensional representation maintains some meaningful properties of the original data
- The mapping can be:
  - **Nonparametric**, where we compute an **embedding** $\boldsymbol{z}^{(i)}$ for each input $\boldsymbol{x}^{(i)}$ in the dataset but not for any other points.
    - This is mostly used for **Data Visualization** ($k = 2$ or $k = 3$) and **interpretability**
  - A **parametric** model $\boldsymbol{z} = f_{\boldsymbol{\theta}}(\boldsymbol{x})$, which can be applied to any input. Can be used:
    - For **Data Compression** ($1 \leq k < n$)
    - As a **preprocessing step** for other learning algorithms (e.g., supervised learning or clustering)
    - For **Data Visualization**
- When we talk about dim. reduction, we typically mean feature extraction (not selection)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Visualization

Why?

- Understand the relationship between data (e.g., if there are clusters)
- Interpretability: can be a requirement in certain domains (e.g., medical, governmental)

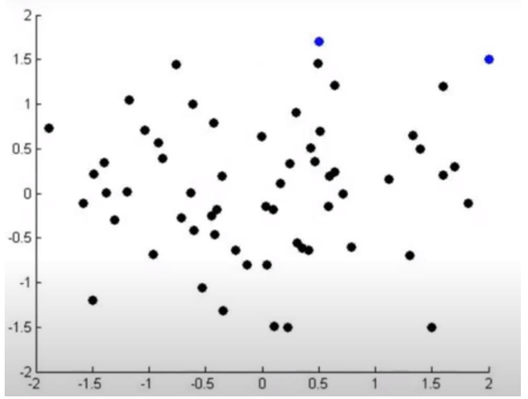
Example: Dataset with statistics and facts about different countries in the world

x_1: GDP
x_2: Per capita GDP
x_3: Life expectancy
…
x_{50}: Mean household income

⇨

z_1: Country size / GDP
z_2: Per capita GDP

⇨



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Compression

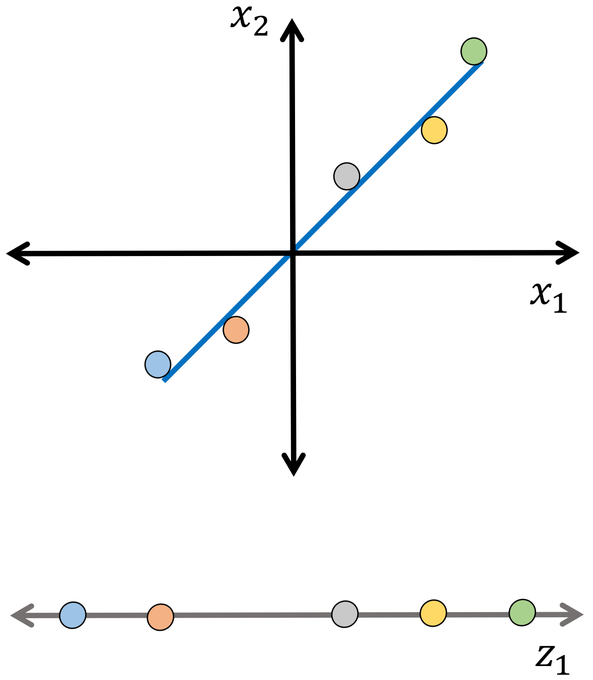
Advantages

- Reduce redundant features and noise
- Reduce the memory needed to store the data
- Reduce overfitting
- Make ML algorithms computationally efficient (e.g., SVMs)

Disadvantages

- May lead to some amount of data loss
- May lead to reduced accuracy

Example: Reduce data from 2D to 1D



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Principal Component Analysis

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

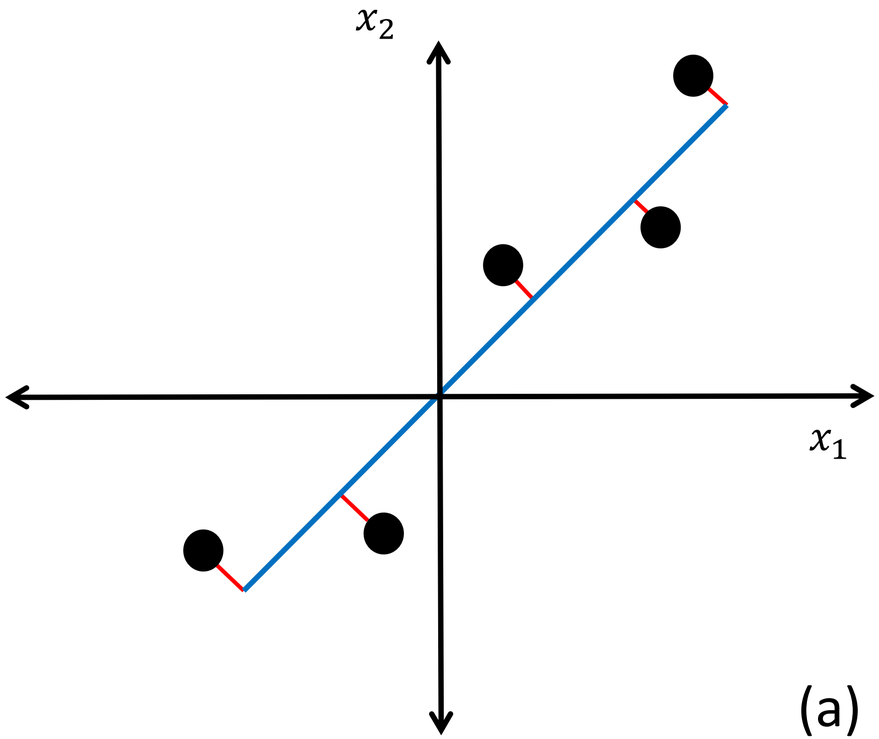
Principal Component Analysis (PCA)

- Simplest and most widely used form of feature extraction
- **Linear transformation** of the data into a **new coordinate system** where most of the variation in the data can be described with fewer dimensions than the initial data
- It finds a linear combination of features that **best summarizes the data**
- What is the best (linear) “summary”?
  - One that allows reconstructing the original data as well as possible
  - One that maximizes the variance of the low-dimensional data
    - Variance is information!
  - These are **equivalent** so PCA does both!

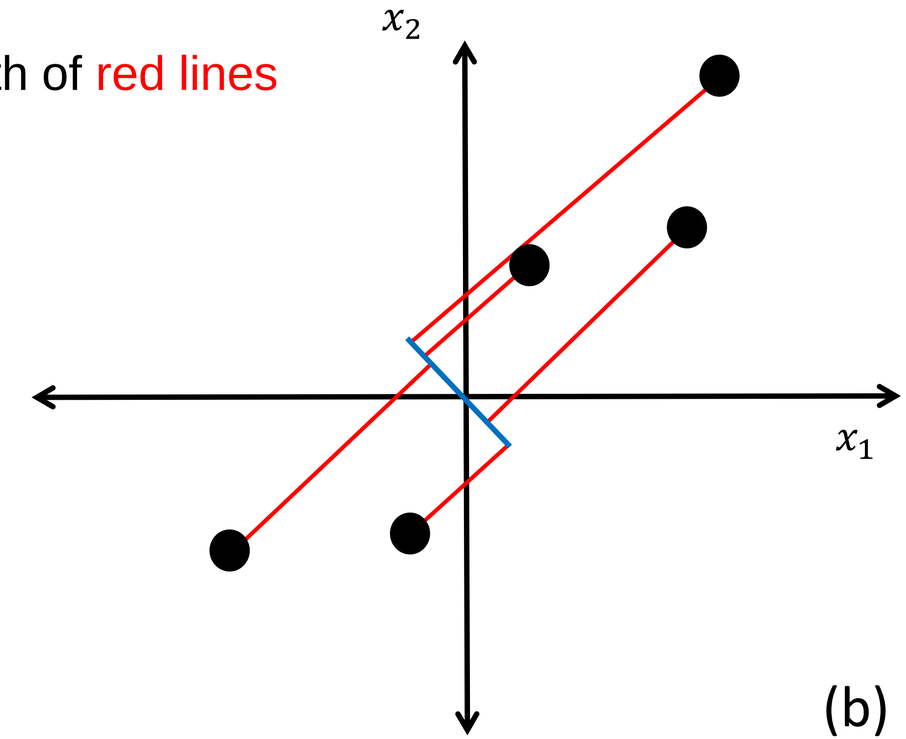
This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Which line has lower projection error?

Sum of the length of red lines



(a)

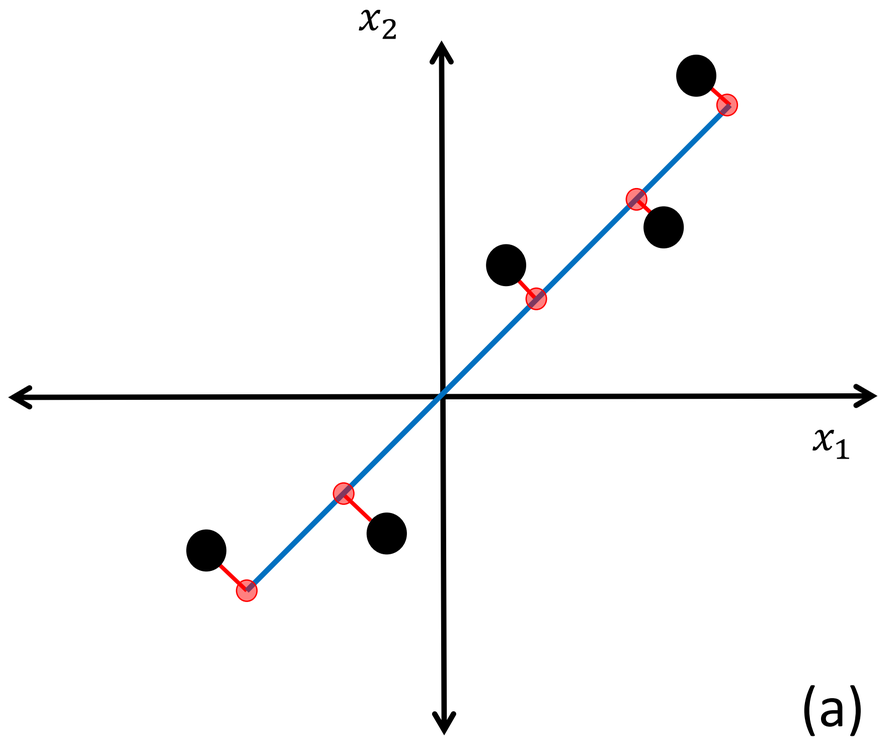


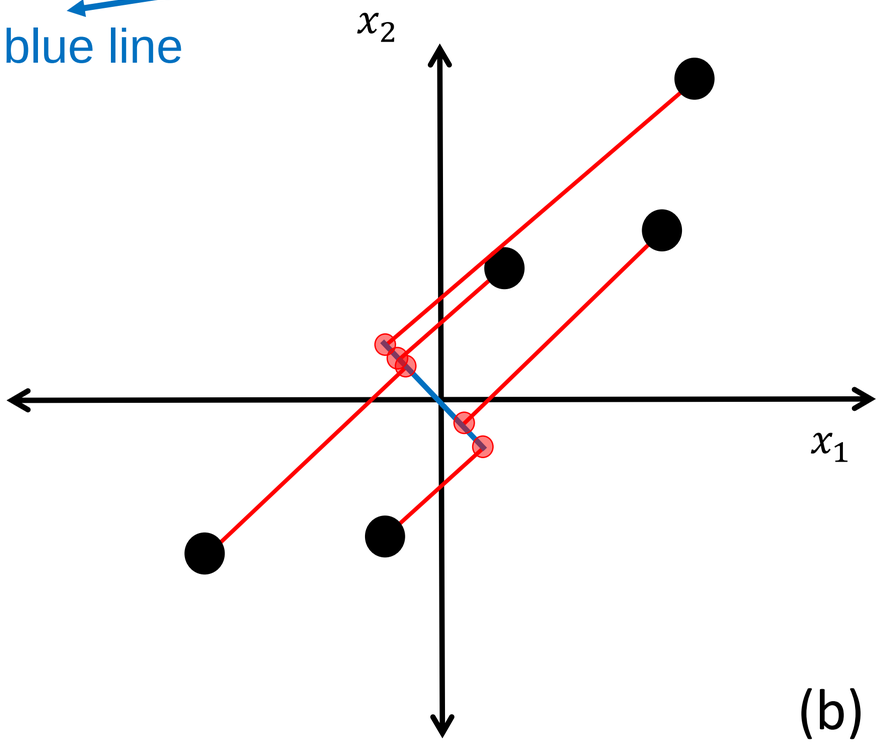
(b)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

In which case the projected points have more variance?

Size of the blue line

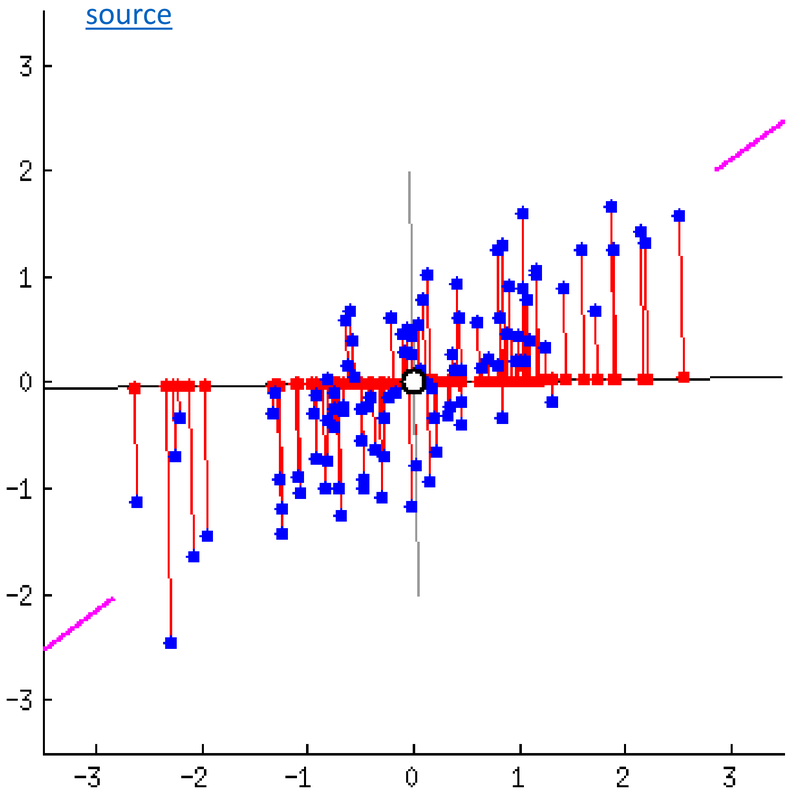




This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Principal Component Analysis

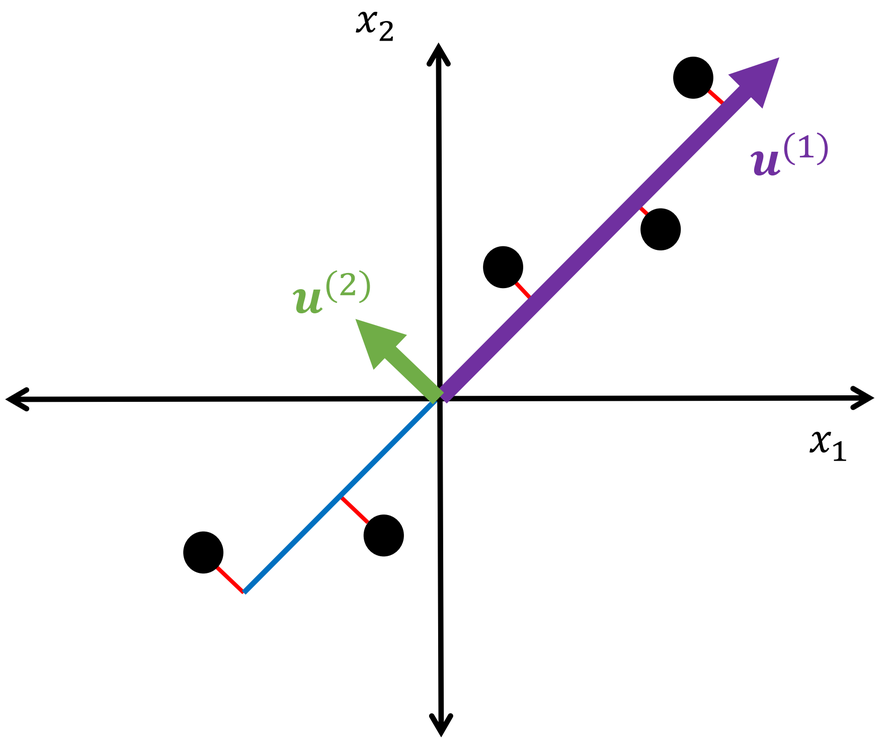
source



- Mean of the blue points is the white circle (the centre)
- Variance of the red points is the average squared distance from the centre
- Out of all the possible lines that rotate around the centre, PCA finds the one on which the projected points have maximal variance

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Principal Component Analysis



- We want to reduce data from 2D to 1D
$$\boldsymbol{x}^{(i)} \in \mathbb{R}^2 \rightarrow \boldsymbol{z}^{(i)} \in \mathbb{R}$$
- PCA finds a low dimensional subspace onto which to project the data, so as to minimize the sum of the squared projection errors
- This means that it finds a vector $\boldsymbol{u}^{(1)}$ which specifies that direction
- In general, it finds the n **orthogonal vectors**, called **principal components (or eigenvectors)**, where the first one explains most of the variance, the second one explains the most variance in what is left once the effect of the first component is removed, and so on until all the variance is explained.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Principal Components Analysis Algorithm

Reduces data from n dimensions to k dimensions

Step 0: **Use feature scaling** to ensure that every feature is normalized (zero-mean)

Step 1: Compute the covariance matrix:

$C = \frac{1}{m}\sum_{i=1}^{m}\left(\boldsymbol{x}^{(i)}\right)\left(\boldsymbol{x}^{(i)}\right)^{T} = (1/m) * X^{T} * X$ where $X \in \mathbb{R}^{m \times n}$

$$C = \begin{bmatrix} var(x_1) & cov(x_1, x_2) \\ cov(x_1, x_2) & var(x_2) \end{bmatrix}$$

Step 2: Compute the "eigenvectors" of matrix C:

$[U, S, V]$ = numpy.linalg.svd(C)

$U = \left[\boldsymbol{u}^{(1)}, \boldsymbol{u}^{(2)}, \ldots, \boldsymbol{u}^{(n)}\right] \in \mathbb{R}^{n \times n}$ where $\boldsymbol{u}^{(i)} \in \mathbb{R}^{n}$

Step 3: Take k first columns of matrix U:

W=U(:,1:k)

$W = \left[\boldsymbol{u}^{(1)}, \boldsymbol{u}^{(2)}, \ldots, \boldsymbol{u}^{(k)}\right] \in \mathbb{R}^{n \times k}$

Step 4: Transform data

$\boldsymbol{z}^{(i)} = W^{T}\boldsymbol{x}^{(i)}$ $\left(\boldsymbol{z}^{(i)} \in \mathbb{R}^{k}\right)$

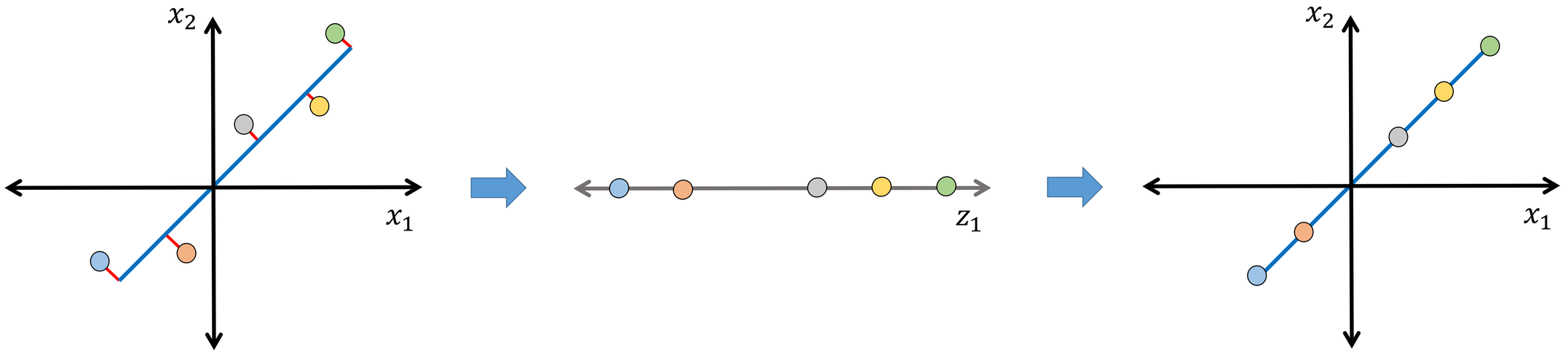
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Reconstruction from a compressed representation

Projection: encodes $\boldsymbol{x}$ to get $\boldsymbol{z} = W^T \boldsymbol{x}$

Reconstruction: decodes $\boldsymbol{z}$ to get $\hat{\boldsymbol{x}} = W\boldsymbol{z}$



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Choosing k (number of principal components)

Average squared projection error: $\frac{1}{m}\sum_{i=1}^{m}\left\|\boldsymbol{x}^{(i)}-\hat{\boldsymbol{x}}^{(i)}\right\|^2$

Total variation in the data: $\frac{1}{m}\sum_{i=1}^{m}\left\|\boldsymbol{x}^{(i)}\right\|^2$

Typically, choose k to be smallest value so that

$$\frac{\frac{1}{m}\sum_{i=1}^{m}\left\|\boldsymbol{x}^{(i)}-\hat{\boldsymbol{x}}^{(i)}\right\|^2}{\frac{1}{m}\sum_{i=1}^{m}\left\|\boldsymbol{x}^{(i)}\right\|^2} < 0.01 \qquad (1\%)$$

"99% of variance is retained"

Slide adapted from Andrew Ng – Machine Learning Course (Coursera)

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Choosing k (number of principal components)

Approach 1:

Set k=1

1. Try PCA(k) and compute $W, \boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}, \hat{\boldsymbol{x}}^{(1)}, \dots, \hat{\boldsymbol{x}}^{(m)}$
2. If $\frac{\frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)}-\hat{x}^{(i)}\right\|^2}{\frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)}\right\|^2} < 0.01$?

   return k

else

   k=k+1

   Go to 1.

Slide adapted from Andrew Ng – Machine Learning Course (Coursera)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Choosing k (number of principal components)

Approach 2:

$[U, S, V]$ = numpy.linalg.svd(Sigma)

$$S = \begin{bmatrix} S_{11} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & S_{nn} \end{bmatrix}$$ diagonal matrix

For k=1 to n:

if $\frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}} \geq 0.99$:

Ratio of explained variance

return k

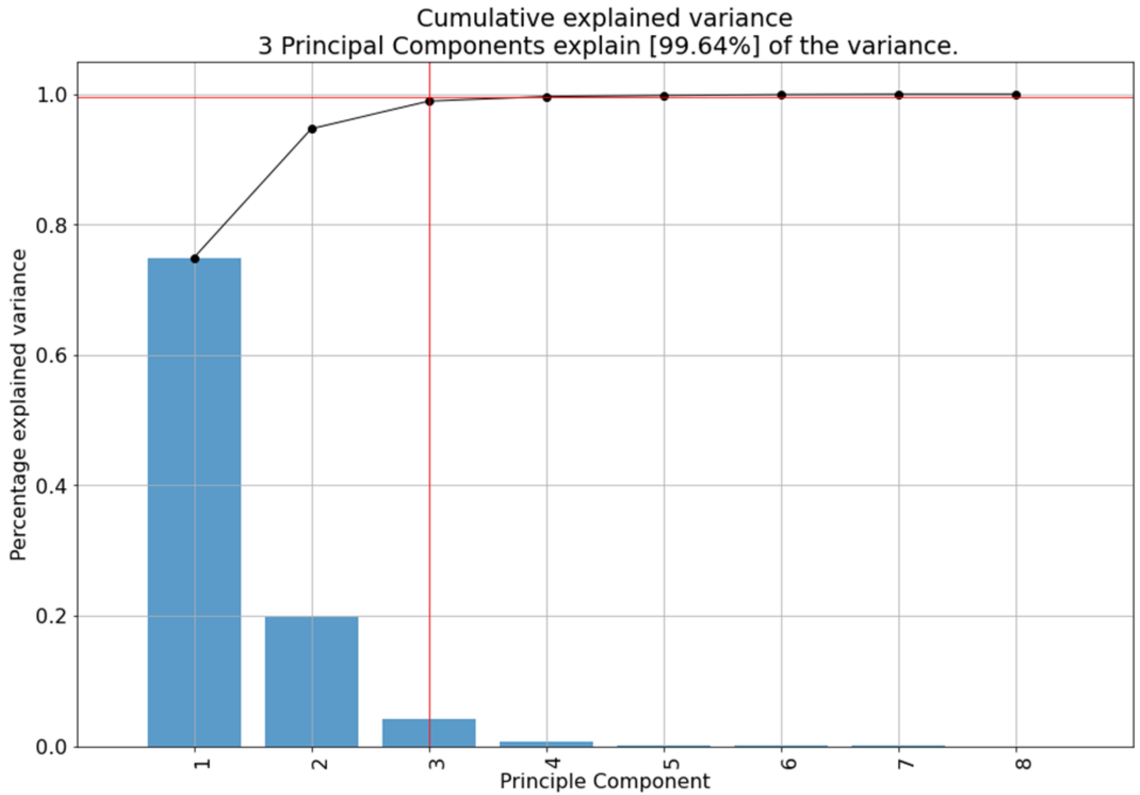
(99% of variance retained)

Slide adapted from Andrew Ng – Machine Learning Course (Coursera)

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Choosing k (number of principal components)



source

We can use a scree plot to help us visualize how much variance is retained

2 PCs cover more than 95% of the variation

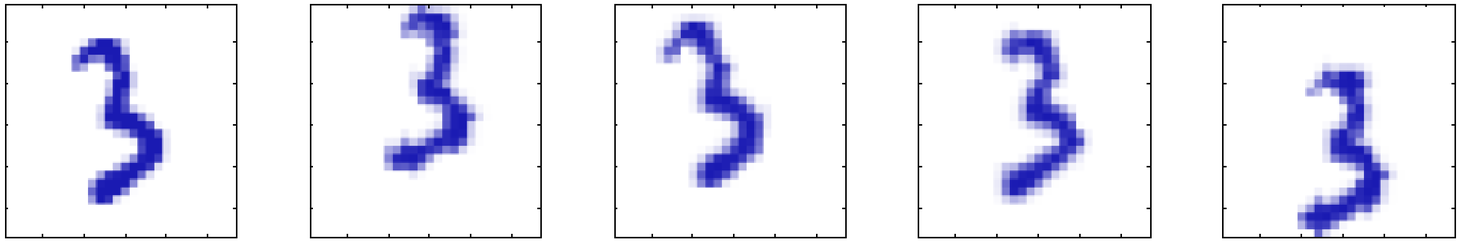
3 PCs cover more than 99% of the variation

→ Reduce the problem from 8D to 3D

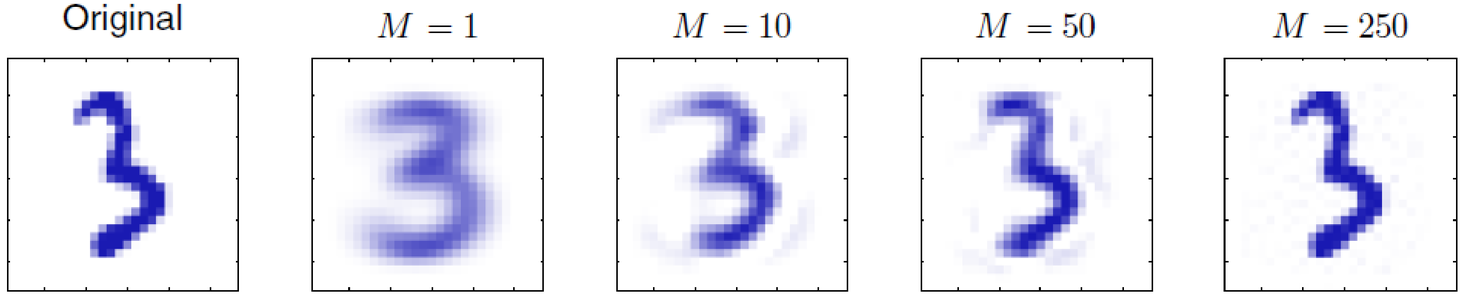
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example

Dataset of digits of 3 shifted and rotated (100x100 = 10,000 pixels)



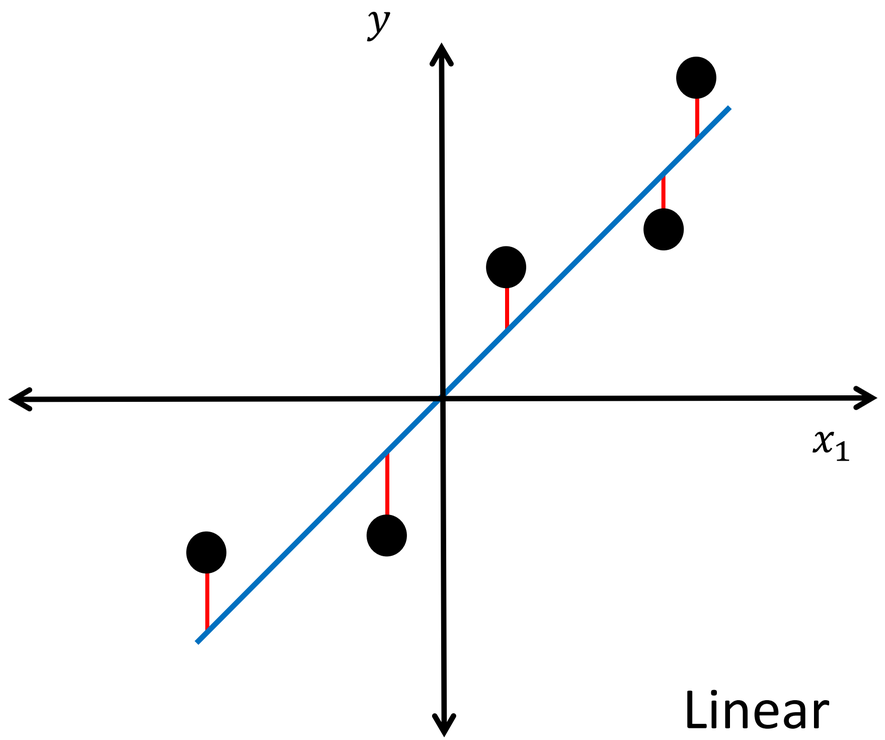
PCA reconstructions obtained by retaining M principal components



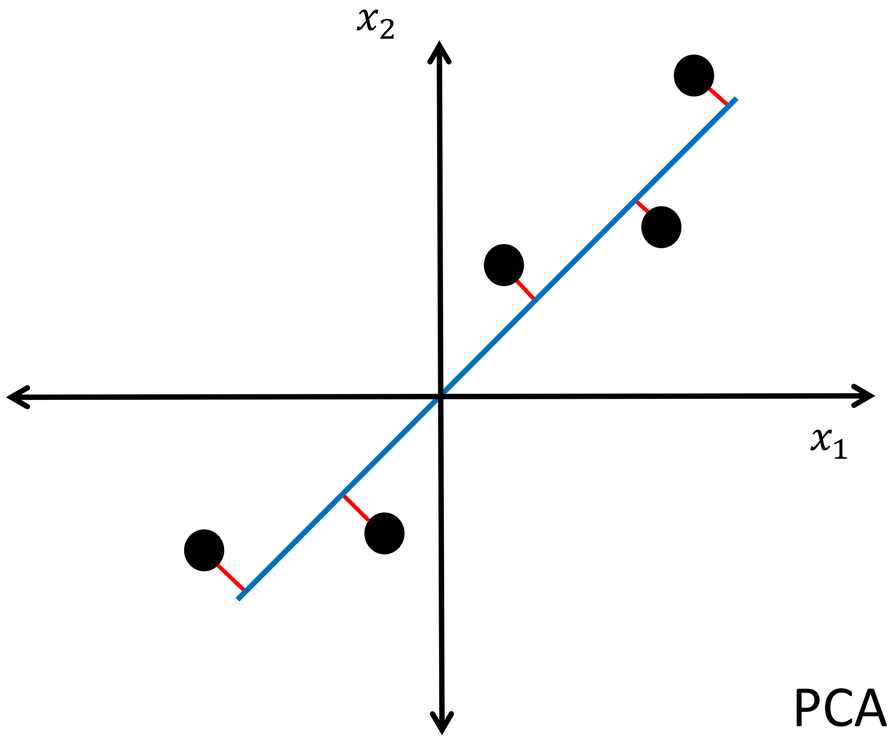
Adapted from: Bishop (2006). Pattern Recognition and Machine Learning.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

PCA is not linear regression



Linear Regression



PCA

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

PCA in scikit-learn

```
>>> import numpy as np
>>> from sklearn.decomposition import PCA
>>> X = np.array([[-1, -1], [-2, -1], [-3, -2], [1, 1], [2, 1], [3, 2]])
>>> pca = PCA(n_components=2)
>>> pca.fit(X)
PCA(n_components=2)
>>> print(pca.explained_variance_ratio_)
[0.9924... 0.0075...]
```

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Dimensionality Reduction can help Supervised Learning

$D = \{(\boldsymbol{x}^{(1)}, \boldsymbol{y}^{(1)}), \ldots, (\boldsymbol{x}^{(m)}, \boldsymbol{y}^{(m)})\}$ where $\boldsymbol{x}^{(i)}$ is high-dimensional, e.g., $\boldsymbol{x}^{(i)} \in \mathbb{R}^{10000}$

1. Extract inputs from the tuples and create an unlabeled dataset: $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)} \in \mathbb{R}^{10000}$

2. Apply PCA on the unlabeled dataset and get reduced dimensional representation: $\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)} \in \mathbb{R}^{500}$

Mapping $\boldsymbol{x}^{(i)} \rightarrow \boldsymbol{z}^{(i)}$ should be computed by running PCA only on the training set!

3. Create a new labeled training set: $D_{lowdim} = \{(\boldsymbol{z}^{(1)}, \boldsymbol{y}^{(1)}), \ldots, (\boldsymbol{z}^{(m)}, \boldsymbol{y}^{(m)})\}$

4. Use the new labeled dataset to train a supervised learning model (e.g., logistic regression): $f_{\boldsymbol{\theta}}\ (\boldsymbol{z})$

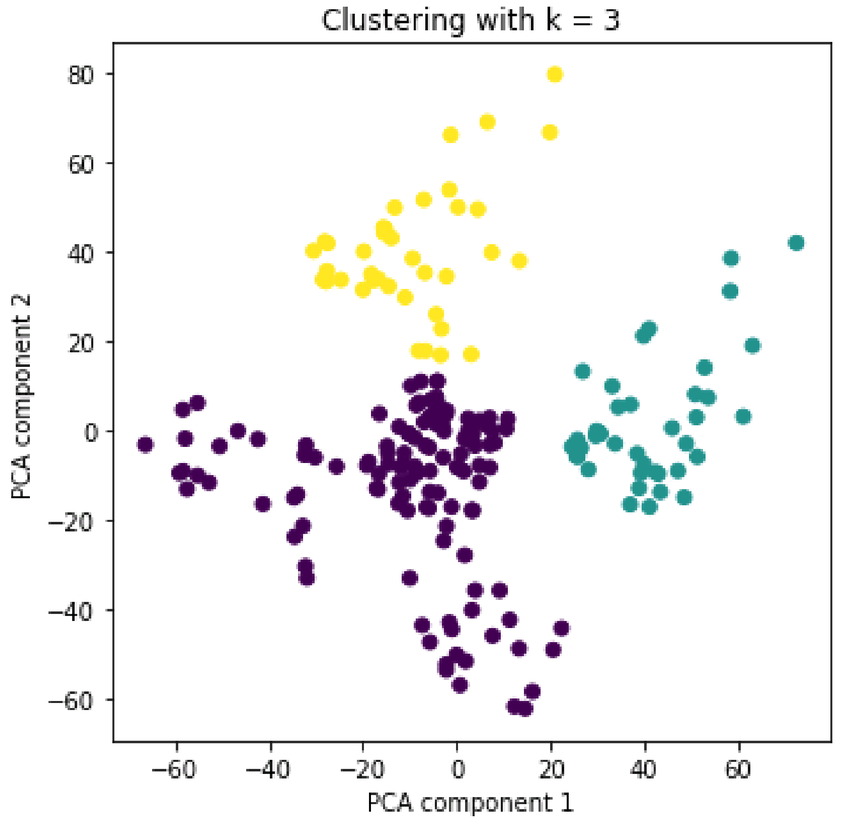
5. For a validation/test input, $\boldsymbol{x}^{(new)}$, we transform it to $\boldsymbol{z}^{(new)}$, and feed it to the model

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Dimensionality Reduction can help Clustering

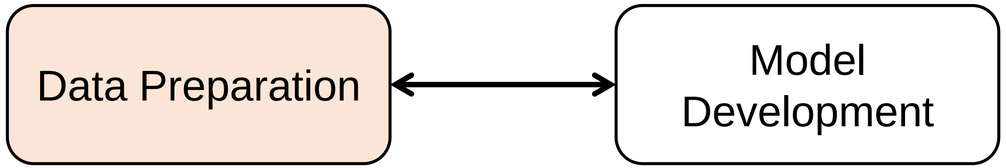
- Clustering algorithms rely on distance computations
- Distance computations in high dimensions is problematic
- When we want to cluster high-dimensional data we often perform dimensionality reduction to obtain the lower-dimensional representation, $\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}$, and perform clustering using these as input.



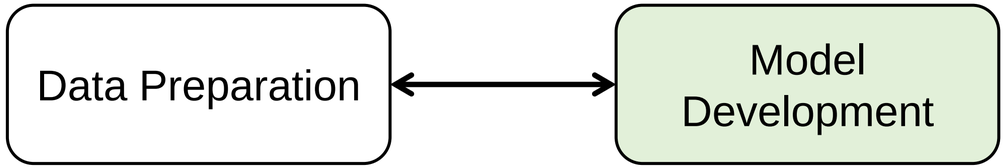
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Where dimensionality reduction lies in the ML project cycle?

- Depends on the usage:
  - Typical use is visualization or compression for helping other learning algorithms



  - However, a task might be about creating a model that reduces memory/disk needed to store certain data



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Nonlinear Dimensionality Reduction

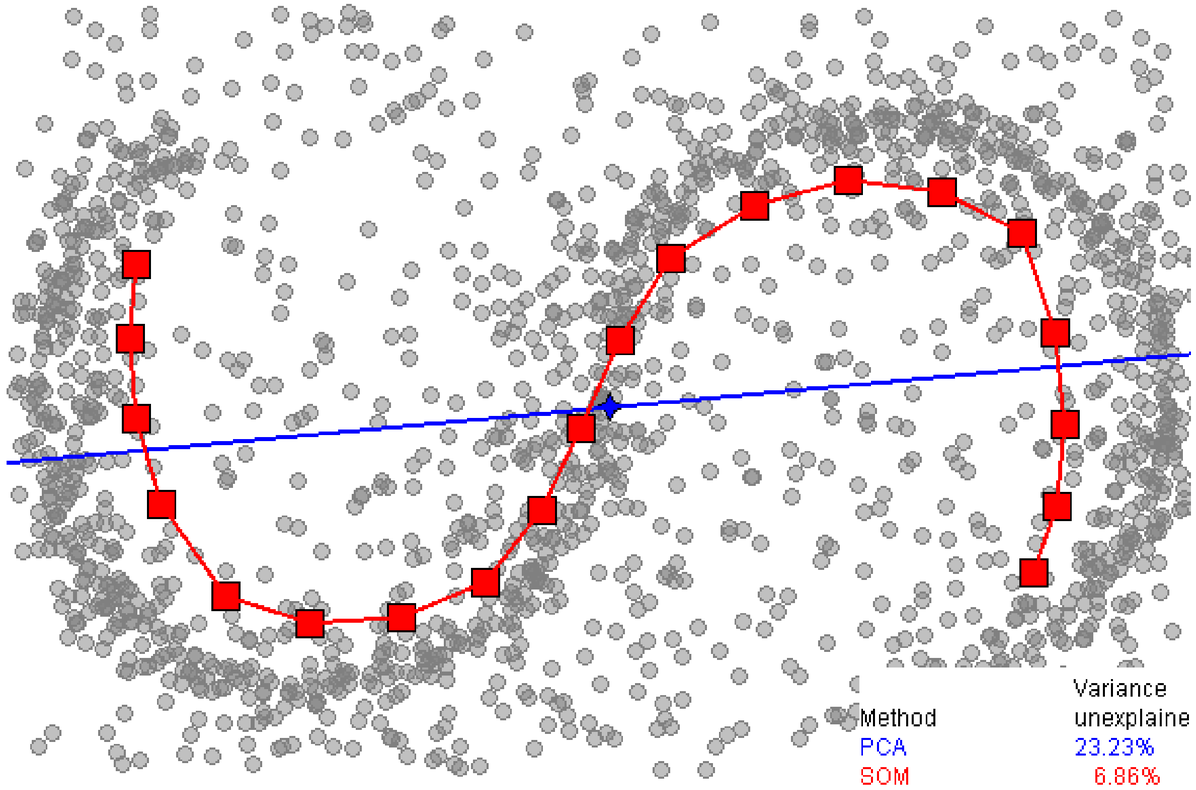
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Nonlinear Dimensionality Reduction

- PCA projects the data onto a lower dimensional **linear** subspace
- What if the data lies onto a low-dimensional **nonlinear** surface?

source



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel PCA

PCA computes the covariance matrix of $X \in \mathbb{R}^{m \times n}$ as follows:

$$C = \frac{1}{m} \sum_{i=1}^{m} \left(x^{(i)}\right) \left(x^{(i)}\right)^T$$

It then projects the data onto the first k eigenvectors of that matrix

Kernel PCA computes the covariance matrix of the data after being transformed into a higher-dimensional space using the kernel trick:

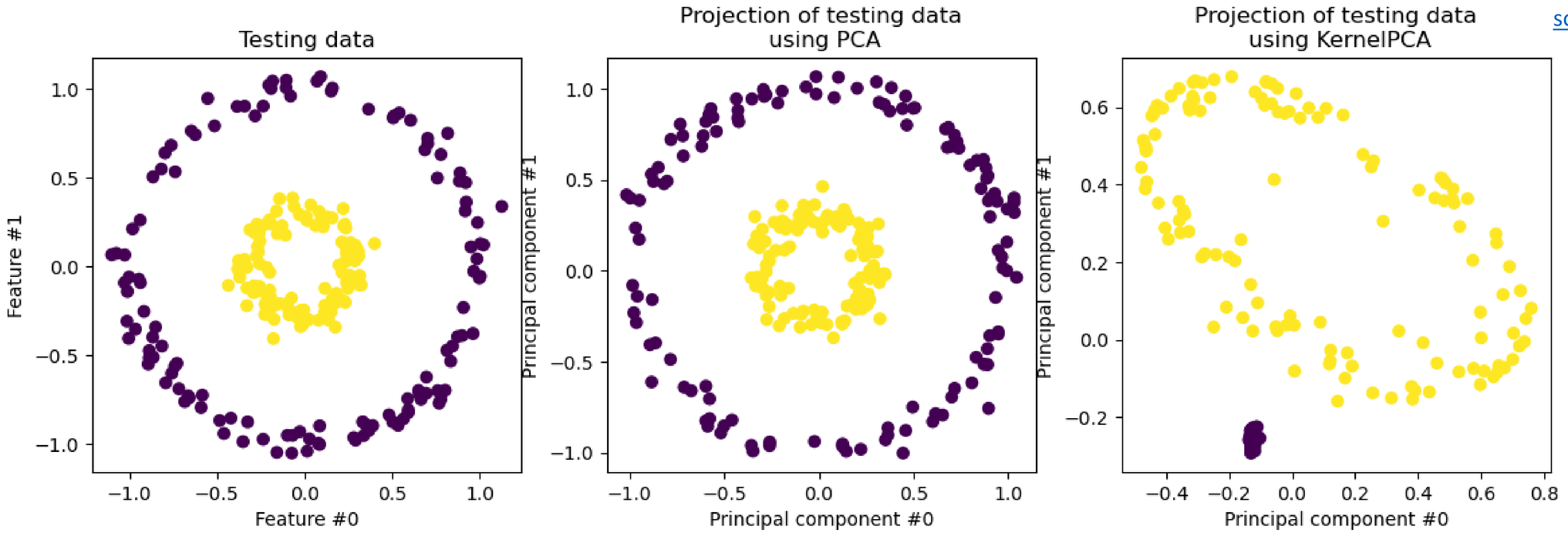
$$C = \frac{1}{m} \sum_{i=1}^{m} \varphi\left(x^{(i)}\right) \varphi\left(x^{(i)}\right)^T$$

It then projects the data onto the first k eigenvectors of that matrix just like PCA

Kernel trick reminder: $\varphi\left(x^{(i)}\right)$ is not explicitly computed

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel PCA



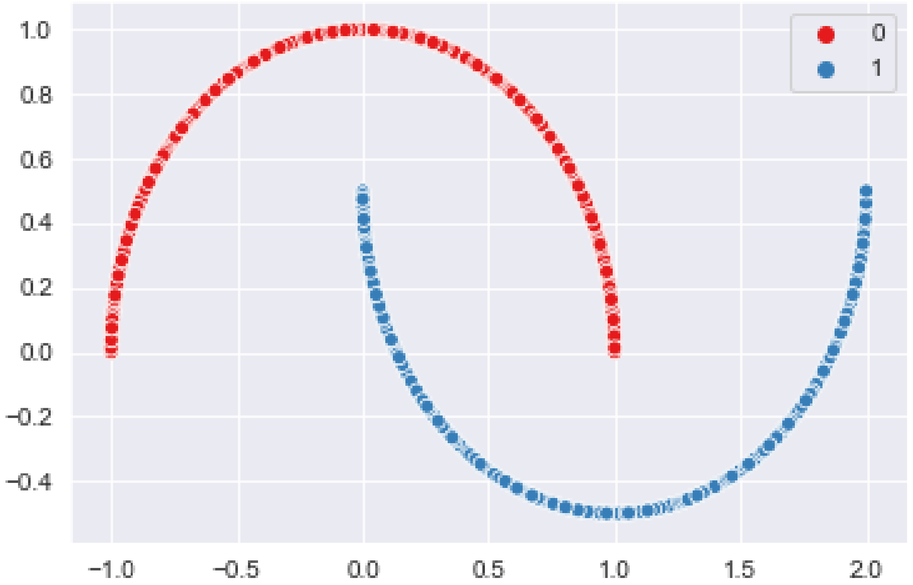
source

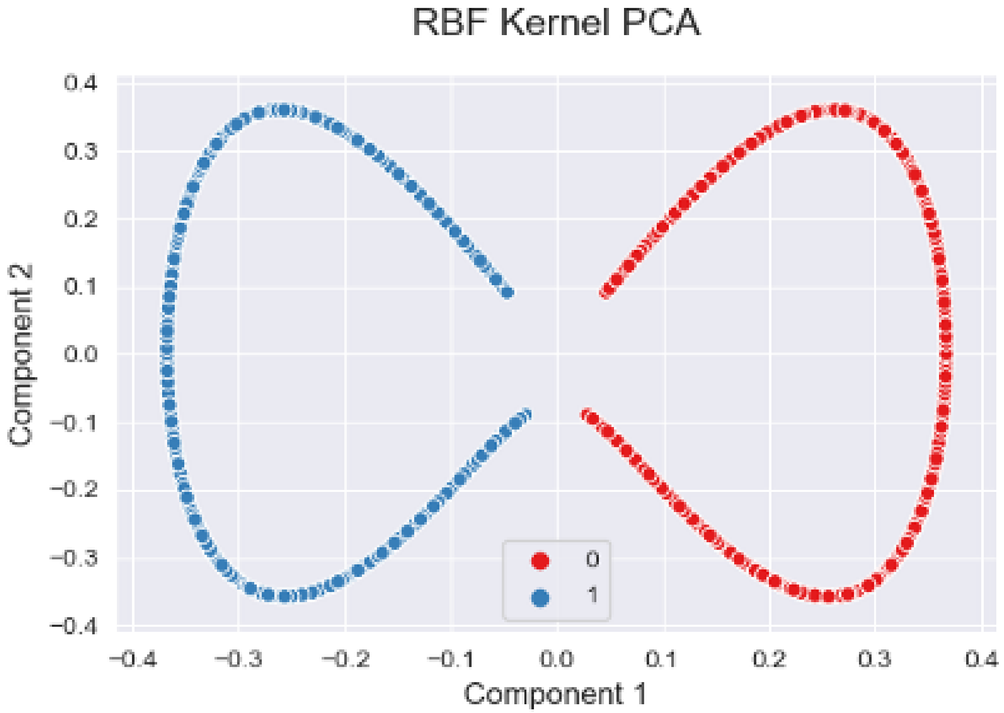
Using an RBF kernel allows to make a nonlinear projection, which unfolds the data while preserving the relative distances of pairs of data points that are close to one another in the original space

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel PCA

source





This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Autoencoders

- PCA can be thought of:
  - Learning a linear mapping $\boldsymbol{x} \rightarrow \boldsymbol{z}$, called the **encoder**, f_e, and learning another linear mapping $\boldsymbol{z} \rightarrow \boldsymbol{x}$, called the **decoder**, f_d.
  - The reconstruction function has the form $r(\boldsymbol{x}) = f_d\big(f_e(\boldsymbol{x})\big)$
  - The model is trained to minimize: $L(\boldsymbol{\theta}) = \|r(\boldsymbol{x}) - \boldsymbol{x}\|^2$

- Autoencoders are **neural networks** that use nonlinear mappings for the encoder and decoder
  - They are trained using backpropagation and gradient descent

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Autoencoders

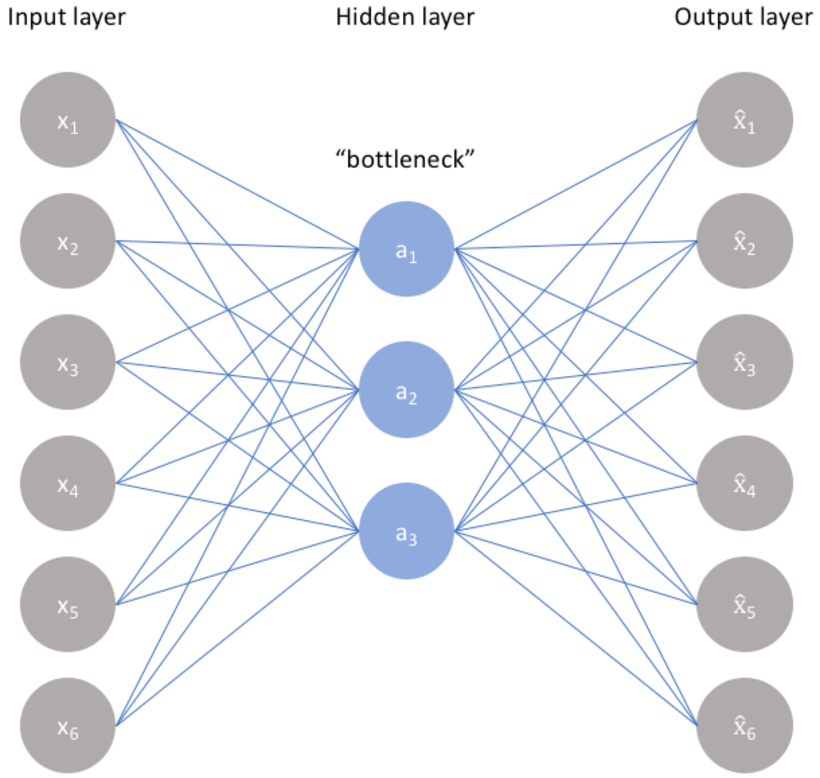


Image source

Hidden layer of size k

- If $k < n$: **undercomplete representation**:
  - low-dimensional bottleneck that forces to encode (compress) the information using less bits
- If $k \geq n$: **overcomplete representation**:
  - the model might learn the identify function!
  - need to impose some kind of regularization, such as adding noise to the inputs, forcing the activation of the hidden units to be sparse, etc.

What if we have 1 hidden layer of linear activation functions?

➢ this is equivalent to PCA

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Autoencoder variants

- **Bottleneck (undercomplete) autoencoder**: lower dimensional hidden layer than input layer
- **Denoising autoencoder**: reconstruct a clean signal from a noisy one
- **Sparse autoencoder**: more hidden nodes than inputs, but sparsity constraint on hidden units activation
- **Contractive autoencoder**: penalty on the derivatives of the hidden units
- **Convolutional autoencoder**: used for image data
- **Variational autoencoder**: generative model, can create new samples and interpolate between them
- …

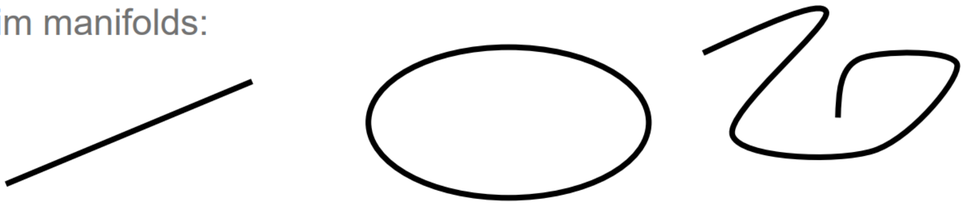
This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Manifold Learning

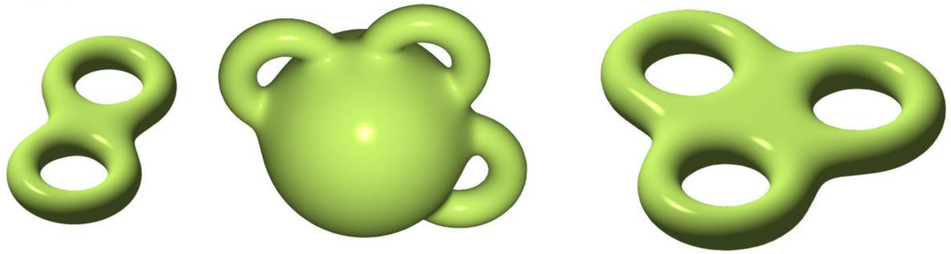
Manifold hypothesis: Many high-dimensional datasets that occur in the real-world actually lie along low-dimensional **manifolds** inside that high-dimensional space.

- **Manifold**: any object which is nearly “flat” on small scales

1dim manifolds:



2dim manifolds:



- Many datasets that appear to require many variables to describe, can actually be described by a smaller number of variables
- The lower dimensional representations of data are often referred to as “intrinsic variables”: values from which the data was produced

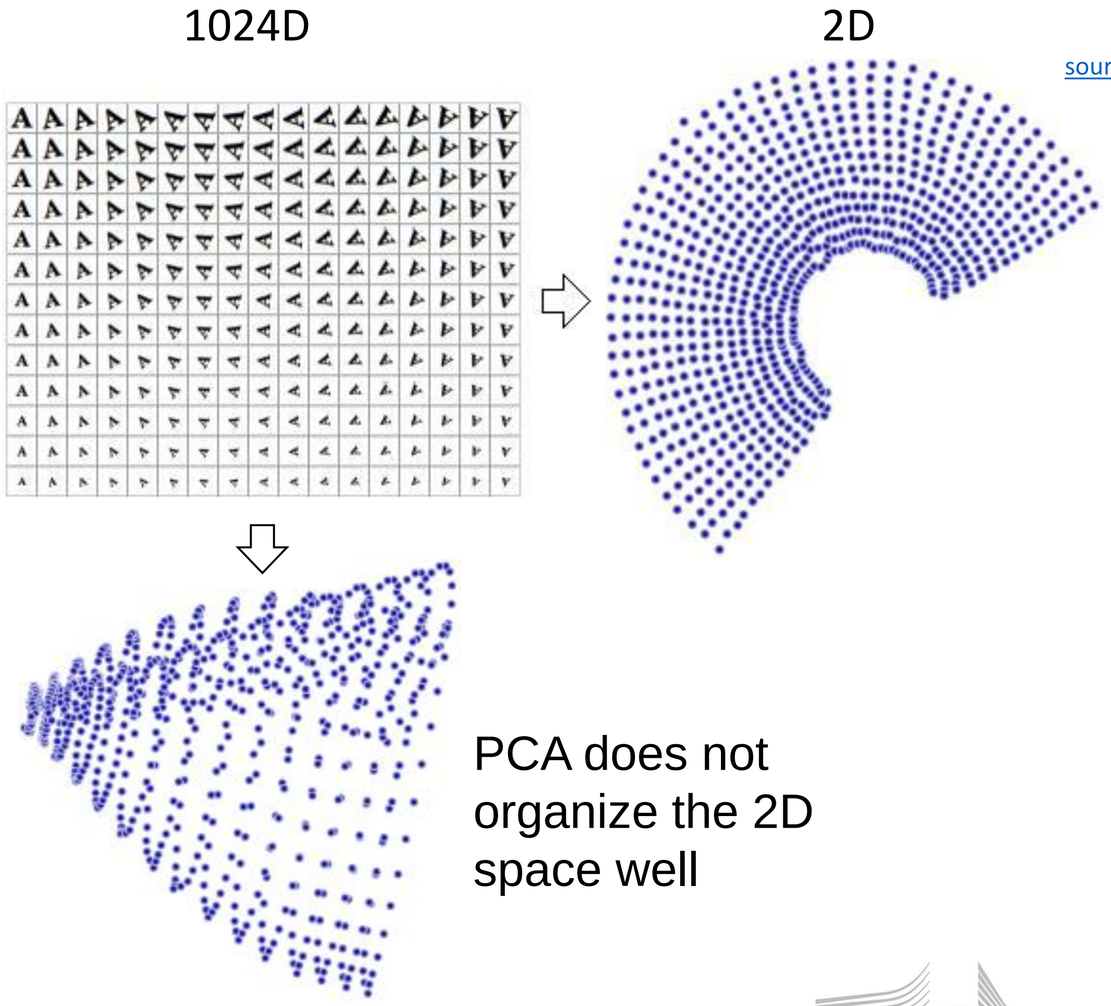
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example

Dataset that contains images of letter “A” which has been scaled and rotated by varying amounts

- Image: 32x32 pixels = 1024 pixel values (dimensions)
- Intrinsic dimensionality = 2 (rotation and scale)
- Information about shape or look of letter “A” is not part of the intrinsic variables
- Nonlinear dimensionality reduction algorithm can reduce the dataset in 2 dimensions and organize the points well



PCA does not organize the 2D space well

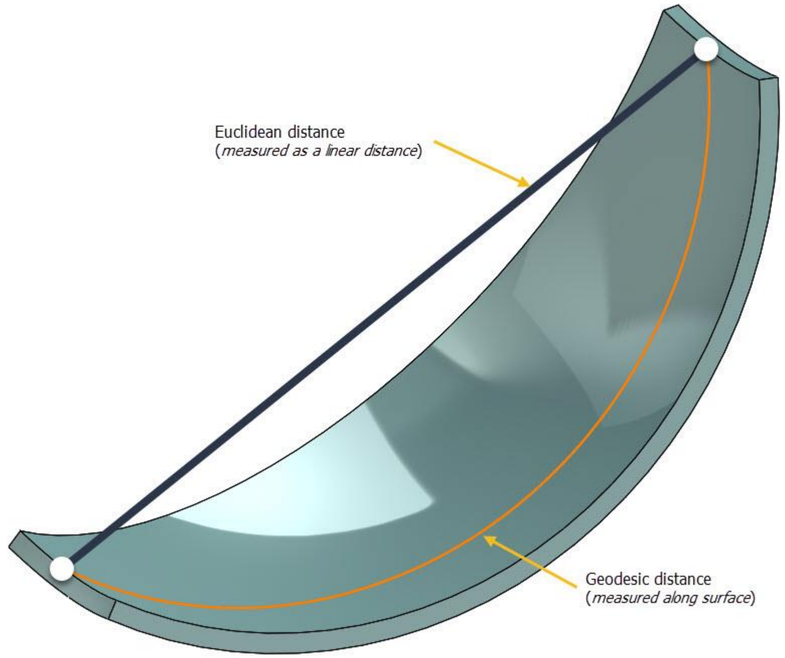
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Euclidean vs Geodesic distance

"The shortest distance between two points is a straight line"

… in **Euclidean geometry**

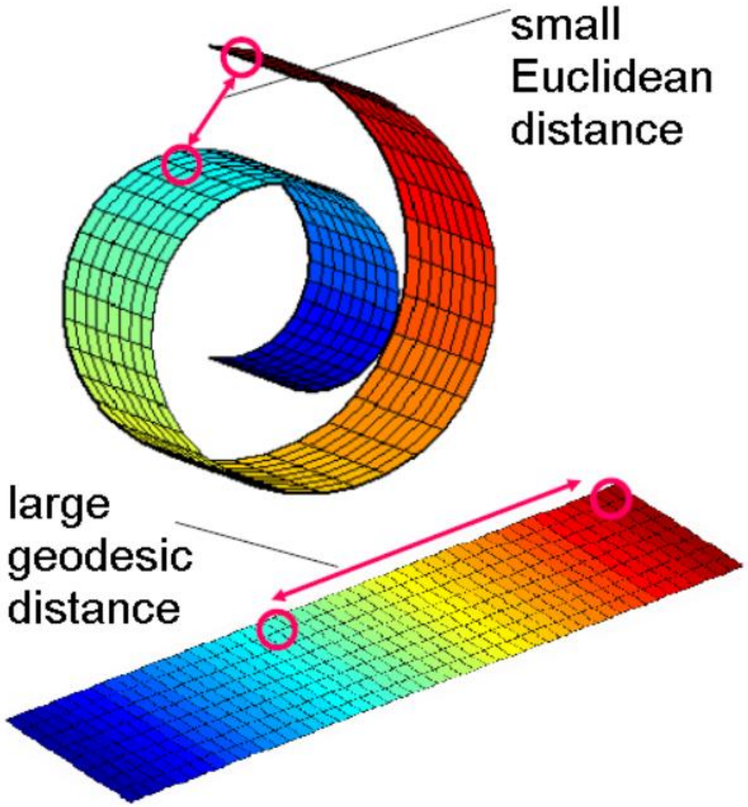
Geodesic distance: measured along the surface (takes into account the geometry)



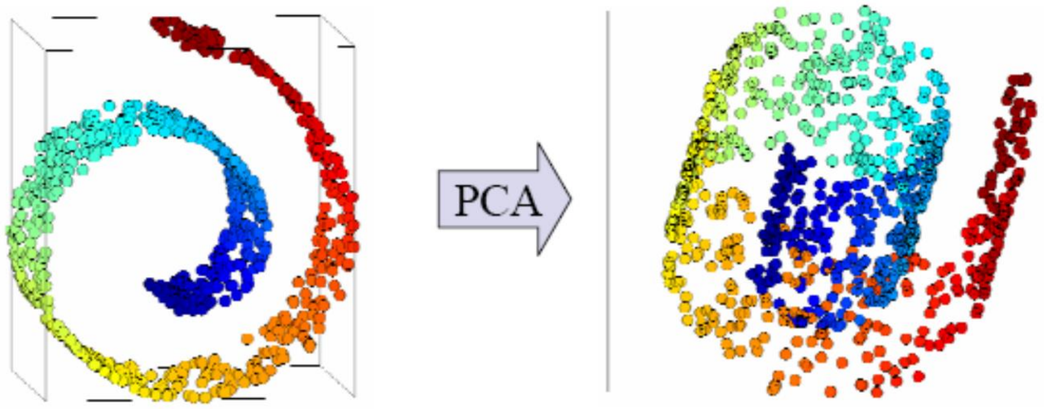
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

“Swiss roll” example: 2D manifold embedded in 3D space



PCA is a linear method: it fails to find the nonlinear structure in the data because it uses the Euclidean distance

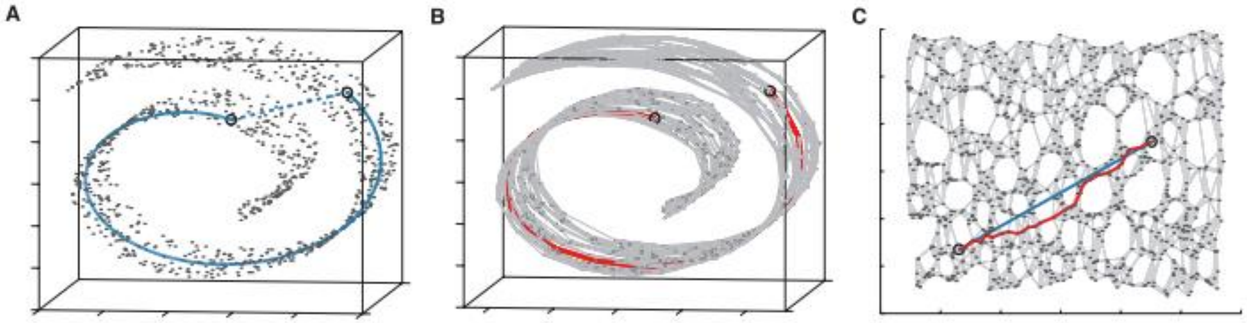


Co-financed by the European Union
Connecting Europe Facility

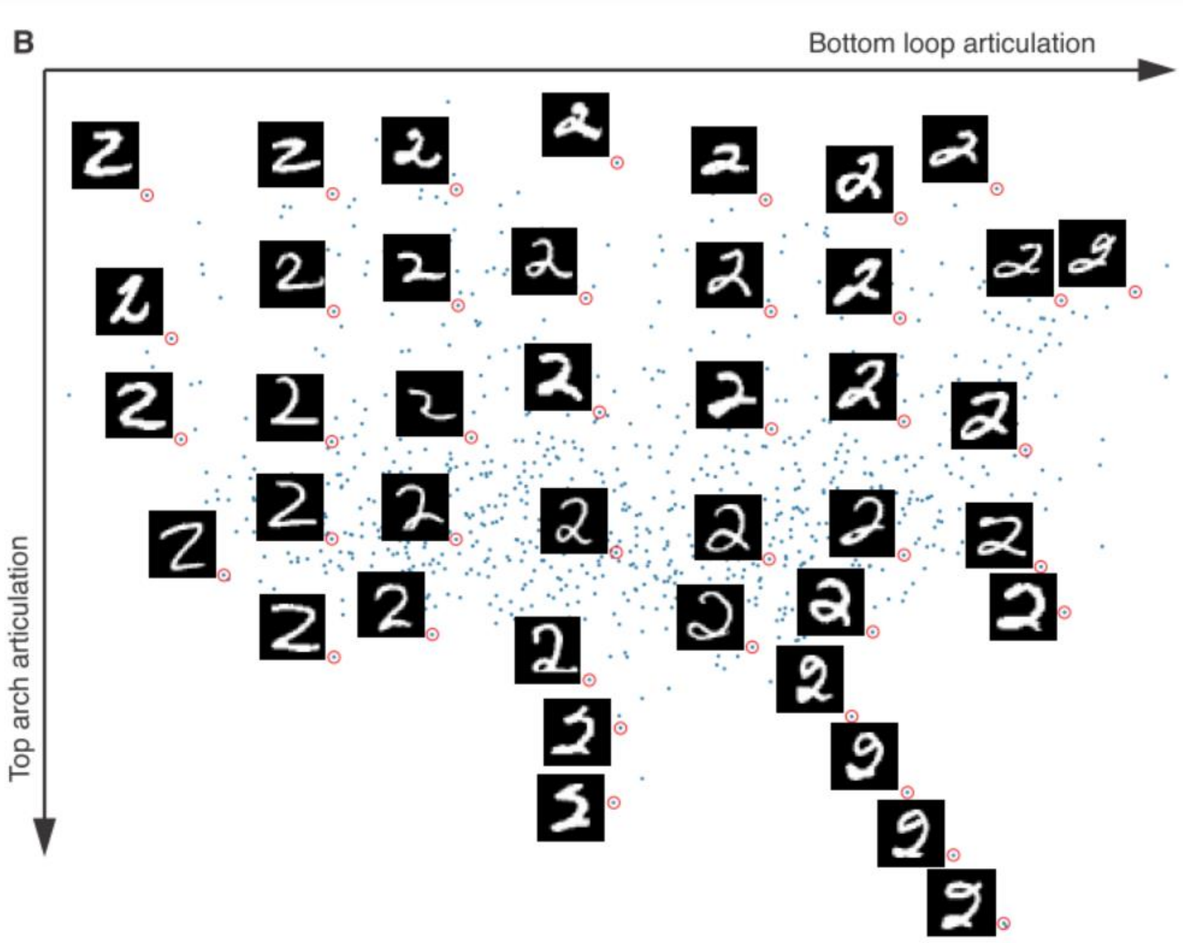
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Isometric mapping (Isomap)

Isomap seeks a lower-dimensional embedding which maintains geodesic distances between all points.



Learning the manifold allows to interpolate between the points (generate points not in the training set)



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

t-distributed stochastic neighbor embedding (t-SNE)

- Widely used method for **visualizing** high-dimensional datasets ($k = 2$ or $k = 3$)
- Computes the probability that pairs of datapoints in the high-dimensional space are related and chooses low-dimensional embeddings which produce a similar distribution
- Preserves only small pairwise distances (local similarities), whereas PCA preserves large pairwise distances to maximize variance
- Optimizes a non-convex cost function
  - Different initializations get different results (stochastic)
  - Can be slow
- Useful for visualization, but not recommended to use for clustering or outlier detection
  - Might show clusters where there are not
- Recommended: use PCA as an initialization step to reduce dimensions to reasonable amount (e.g. 50) to suppress noise and speed up computation

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

PCA vs t-SNE on MNIST digits

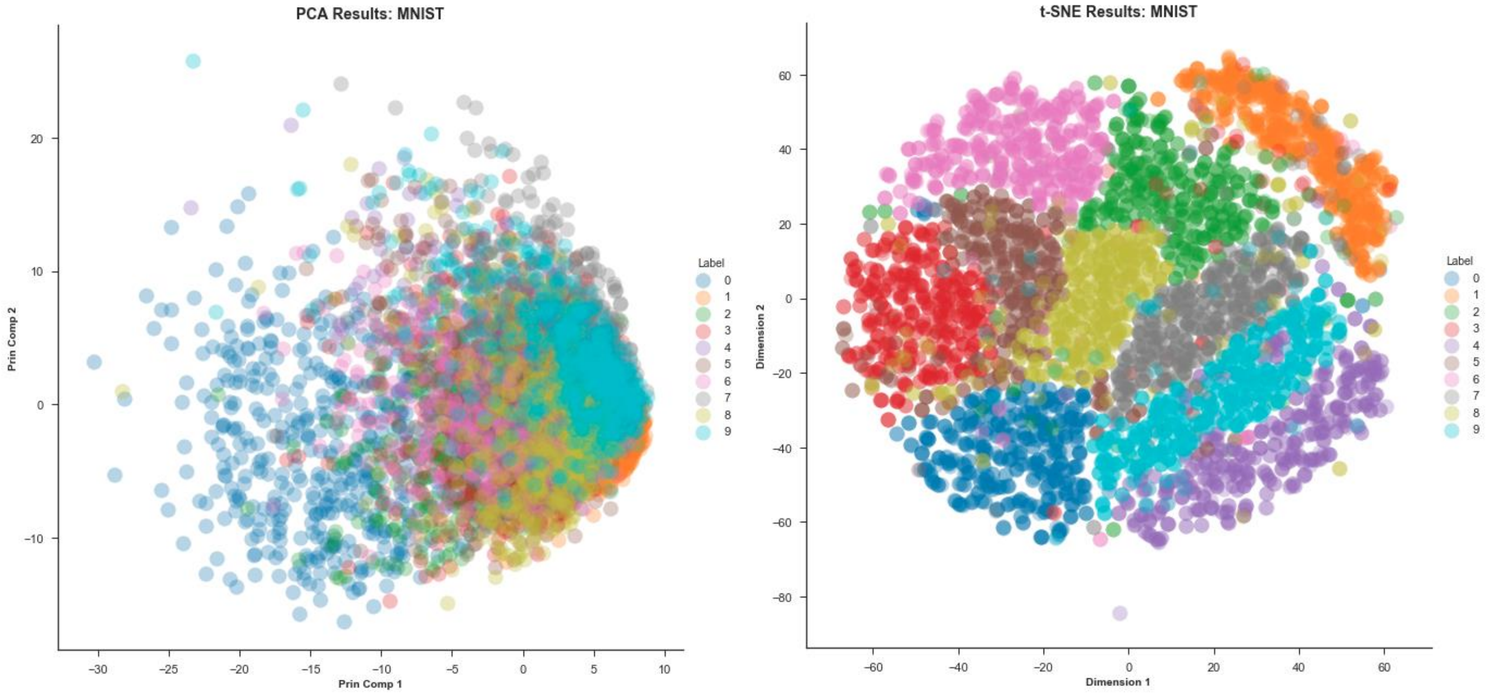
Images source

MNIST: dataset of handwritten digits

60000 training images

10000 test images

sample images:



PCA

t-SNE

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Dimensionality reduction methods

- PCA
- Probabilistic PCA
- Kernel PCA
- Factor analysis
- Kohonen Self-organizing Maps
- Independent Component Analysis
- Autoencoders
- Multi-dimensional scaling
- Maximum variance unfolding
- Locally-Linear Embedding
- Isomap
- Hessian Eigenmaps
- Manifold sculpting
- …

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Next Lectures

- Anomaly Detection
- Recommender Systems

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 14: Anomaly Detection

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Revision

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Dimensionality Reduction

- What is dimensionality reduction: transformation of high-dimensional data to low-dimensional space
- Why dimensionality reduction:
  - Data visualization
  - Data compression
    - Can help ML algorithms (supervised learning, clustering)
- Principal Component Analysis
  - Linear transformation into a new coordinate system
  - Maximize variance of low-dimensional data = minimizing reconstruction error
  - Finds n orthogonal vectors each ranked by how much it explains the variance in data
    - These are the principal components or eigenvectors
  - Projection = encoding in low dimensions
  - Reconstruction = decoding from low dimensions to high-dimensions
  - We can choose the number of components based on a desired ratio of explained variance (typically 90-99%)

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Dimensionality Reduction

- PCA works well in various problems but it is a linear method
- Nonlinear dimensionality reduction methods address this shortcoming
  - Kernel PCA uses the kernel trick
  - Autoencoders are NNs trained to encode and reconstruct the input
- Manifold learning approaches = nonlinear dimensionality reduction methods that explicitly consider that the data lie on low-dimensional structures embedded in high-dimensional space
  - Isomap: instead of Euclidean distance, considers the geodesic distance
    - Geodesic distance: distance on the manifold
    - Swiss roll example: isomap can unfold it
    - Allows better interpolation as the interpolated points expected to lie on the manifold
  - t-SNE
    - used for visualization
    - stochastic method that preserves local similarities
    - can give different results for different initializations

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 14: Anomaly Detection

Learning Outcomes

You will learn about:

1. The problem of anomaly detection and its difference with binary classification
2. The concept of density estimation and how to use it for anomaly detection
3. How to fit the parameters of a Gaussian probability density
4. Building an anomaly detection model using kernel density estimation, one-class SVMs, isolation forests and autoencoders
5. How to evaluate anomaly detection systems

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Anomaly Detection

Anomaly detection algorithms look at an unlabeled dataset of normal events and raise an alarm when an unusual or anomalous event occurs.

Fraud detection example:

Given a dataset $D = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$

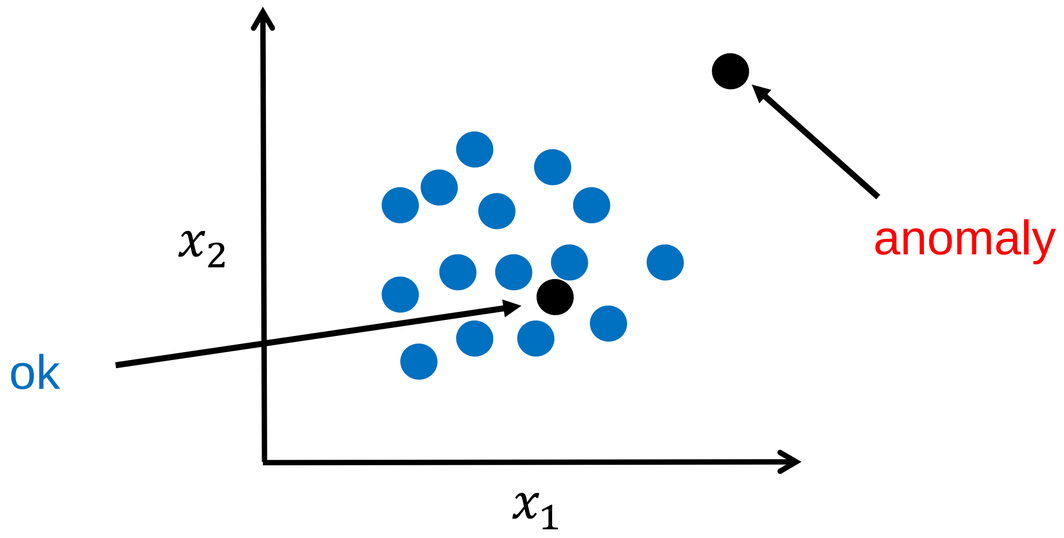
$\boldsymbol{x}^{(i)}$ features of user i's activities

Features:

x_1 = number of transactions

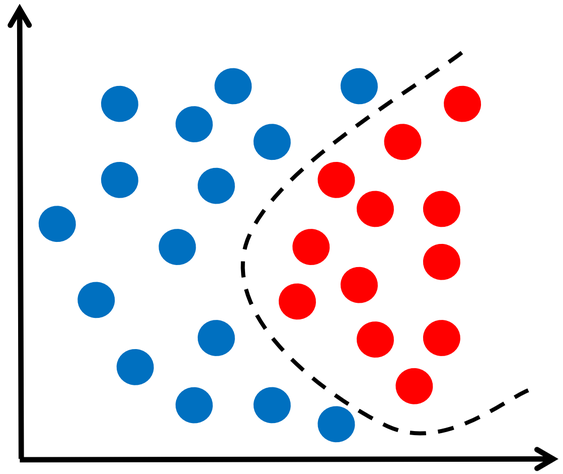
x_2 = typing speed

New user: $\boldsymbol{x}^{(new)}$

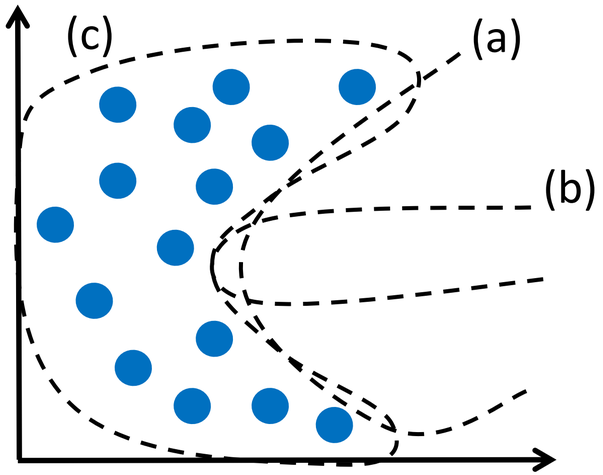


This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Supervised Binary Classification vs Anomaly Detection



- Availability of a large number of positive and negative examples
- Algorithm understands how to structure the decision boundary
- Future negative examples likely to be similar to ones in training set



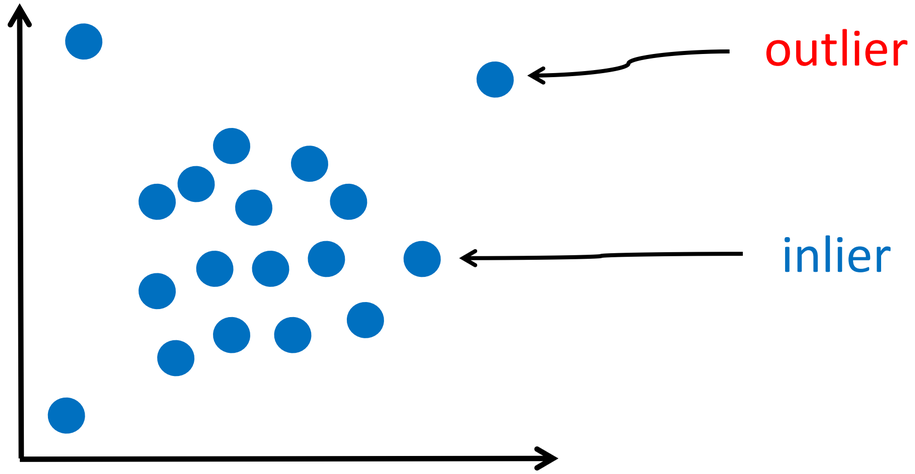
- If there are no negative examples available during training how should the decision boundary be?

- Examples:
  1. Fraud detection
  2. Machine monitoring

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

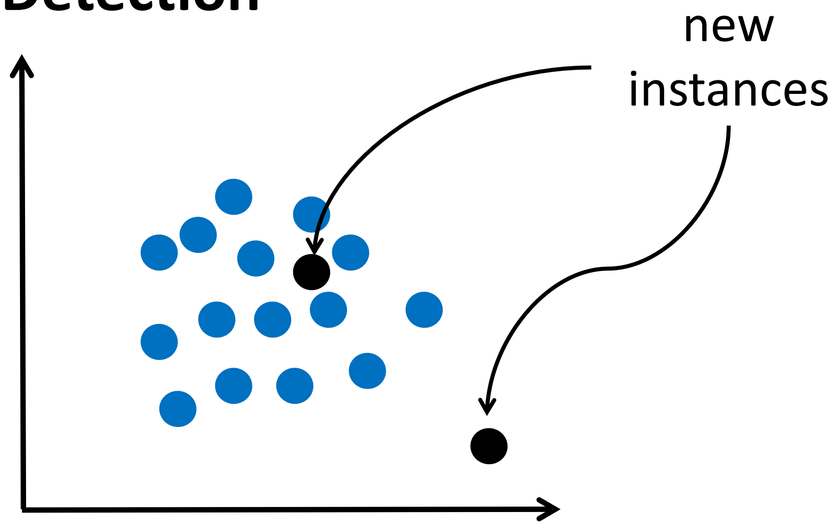
Anomaly Detection

Outlier Detection



- Training data polluted by outliers (instances far from the others)
- Fit regions of most concentrated data
- Assume outliers are not concentrated

Novelty Detection



- No outliers in training data
- Interested in detecting whether a **new** instance is an outlier (novelty)
- Novelties could be concentrated as long as they are in a **low-density** region of the training data

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Density Estimation

Random variable x sampled from **unknown** probability distribution $p(x)$

- x is a feature

Probability density: relationship between outcomes (values) of x and its probability

- Some outcomes of a random variable will have low probability density, others will have a high probability density

Probability distribution: the shape of the probability density across the domain of x

- E.g., uniform, Gaussian, exponential

Calculation of probabilities for specific outcomes of x is performed by a **probability density function** (PDF)

Useful to know the PDF for a sample of data in order to know whether a given observation is unlikely, so as to consider it an outlier or anomaly

(Probability) Density Estimation: we are using the observations from our sample (data) to estimate the density of probabilities beyond just the sample

source

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

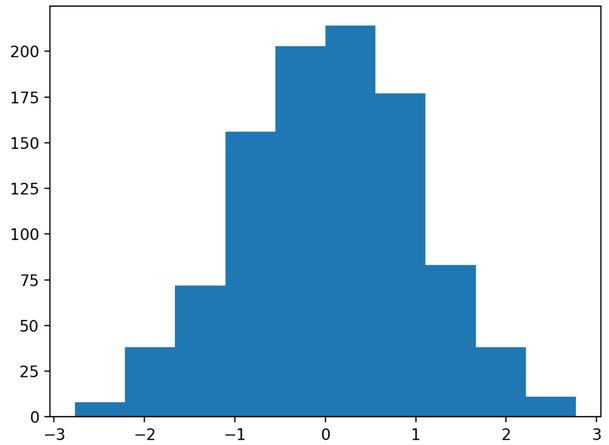
Density Estimation: Histogram

Histogram: plot that groups observations into bins and counting the number of events that fall in the bin

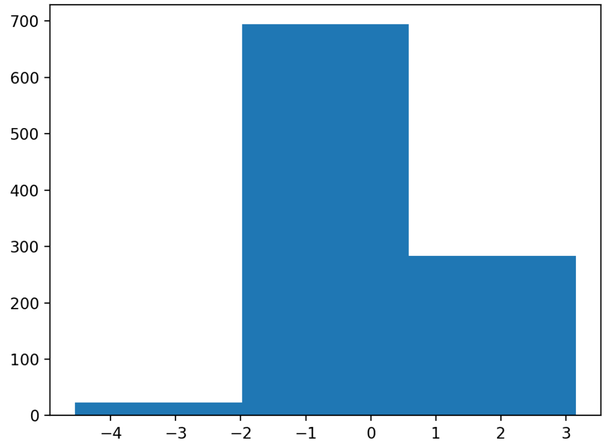
The counts (or frequencies) in each bin are plotted as a bar graph with bins on x-axis and frequency on y-axis

Choice of number of bins is important:
- controls the coarseness of the distribution
- how well the density of observations is plotted

Shape of the histogram often matches well-known probability distributions



10 bins



3 bins

source

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Parametric Density Estimation

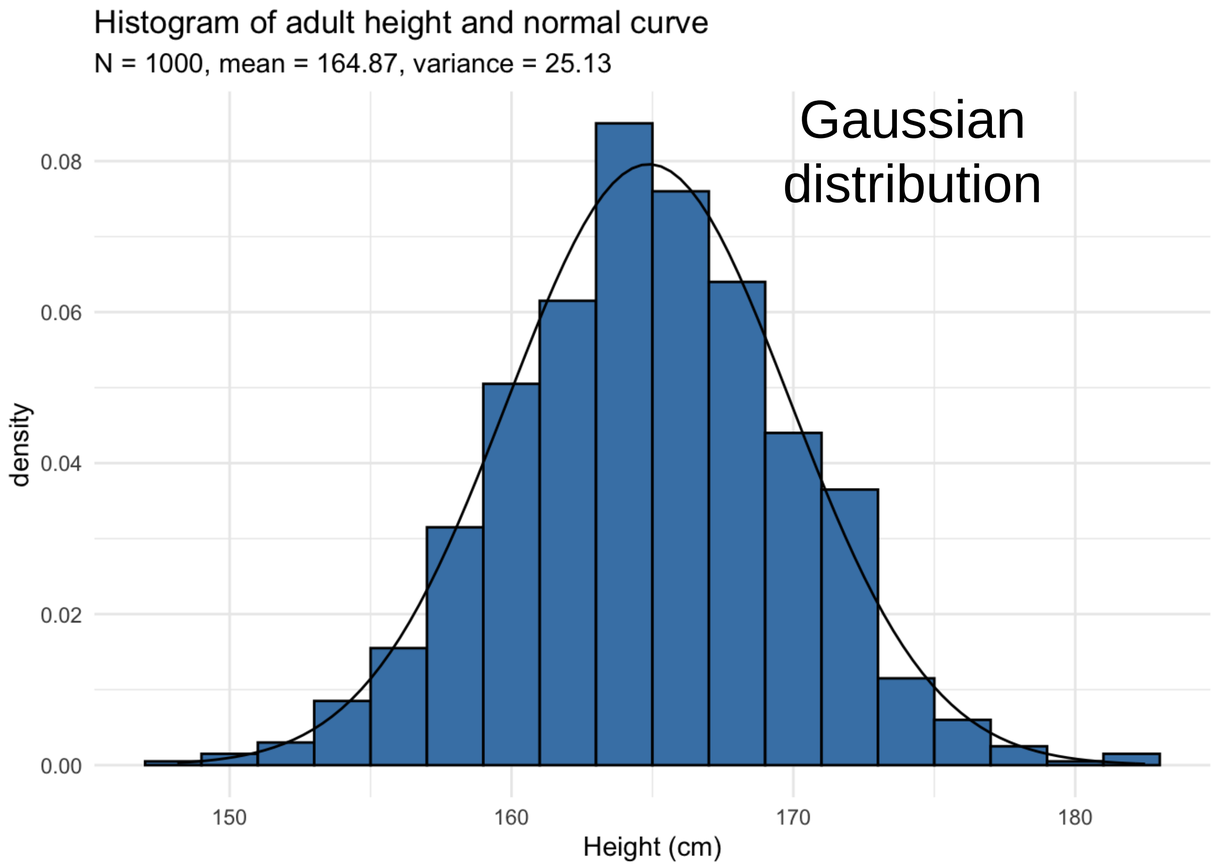
Example:

Take 1000 random people and measure their height

Create a histogram of the data

Sampled points follow a **Gaussian distribution**

It has parameters: mean and variance
- Mean: 164.87
- Variance: 25.13



Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Gaussian Distribution

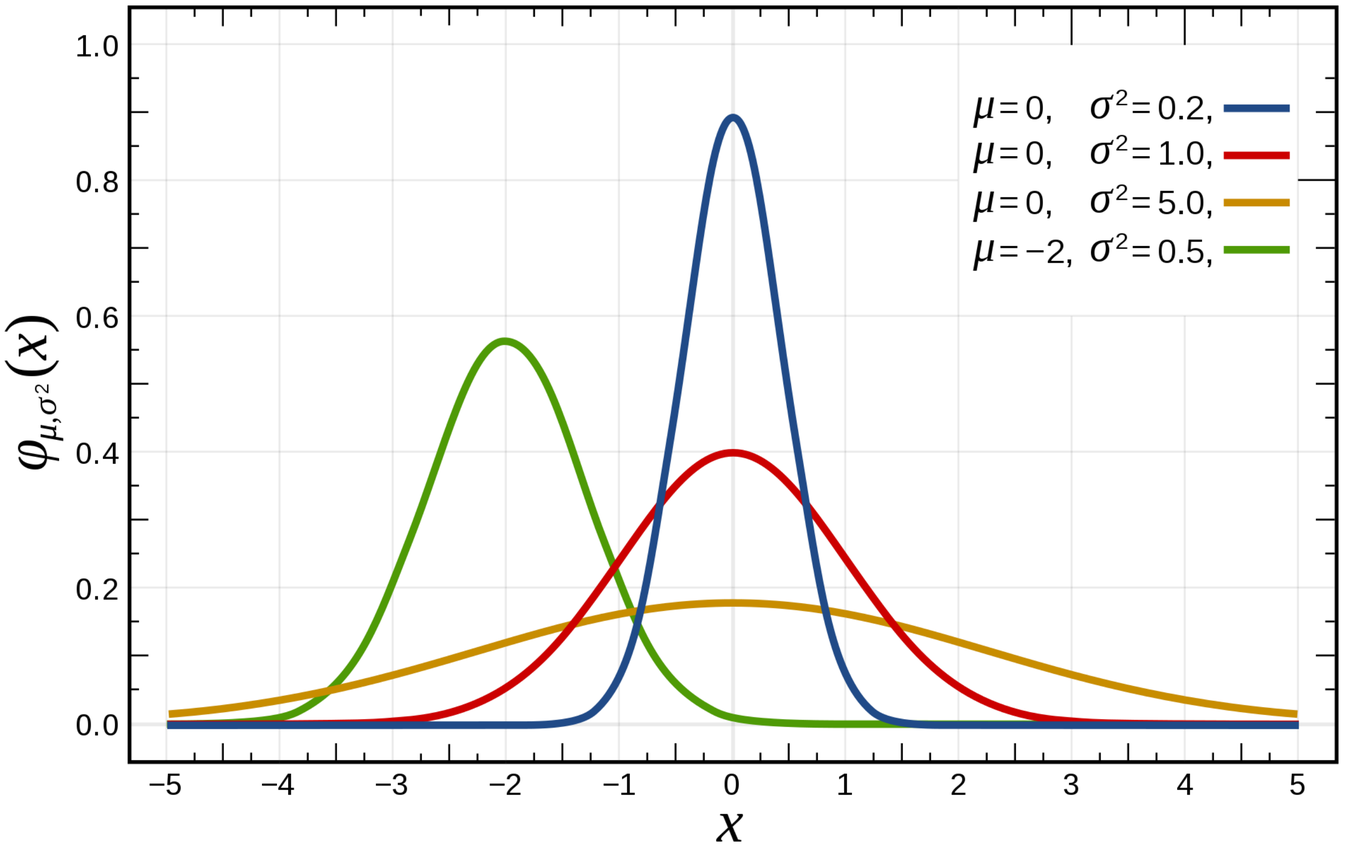
$$x \sim N(\mu, \sigma^2)$$

$$f_{\boldsymbol{\theta}}(x) \quad where \quad \boldsymbol{\theta} = (\mu, \sigma^2)$$

$$f_{\mu,\sigma^2}(x) = \frac{1}{\sigma\sqrt{2\pi}} exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right)$$

$$f_{\boldsymbol{\theta}}(x) = \frac{1}{\theta_2\sqrt{2\pi}} exp\left(\frac{-(x-\theta_1)^2}{2\theta_2^2}\right)$$

Linear regression: $f_{\boldsymbol{\theta}}(x) = \theta_1 x + \theta_2$



Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Parameter fitting

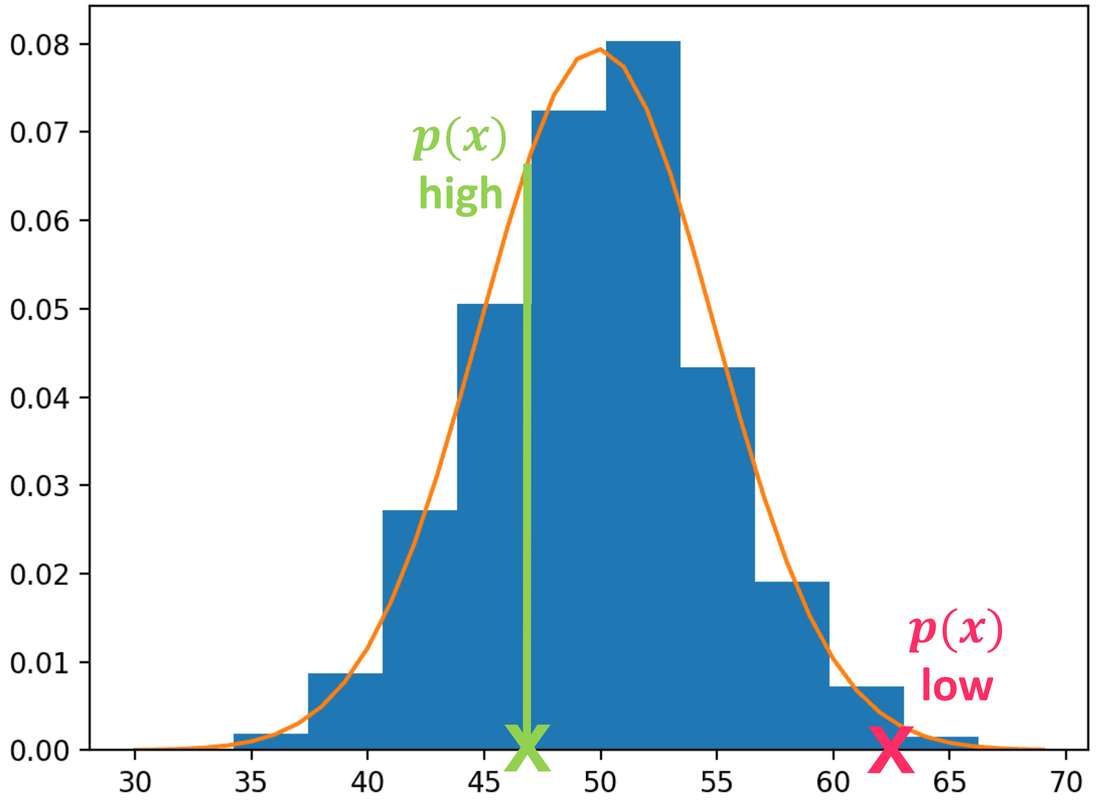
How do we estimate the parameters of this model?

Dataset: $D = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$$

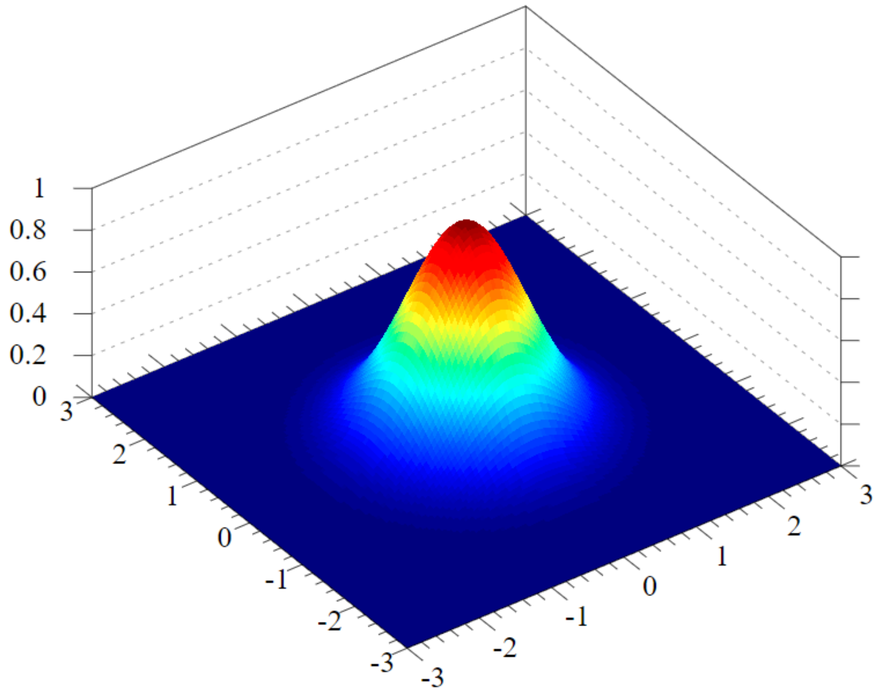
$$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m} \left(x^{(i)} - \mu\right)^2$$

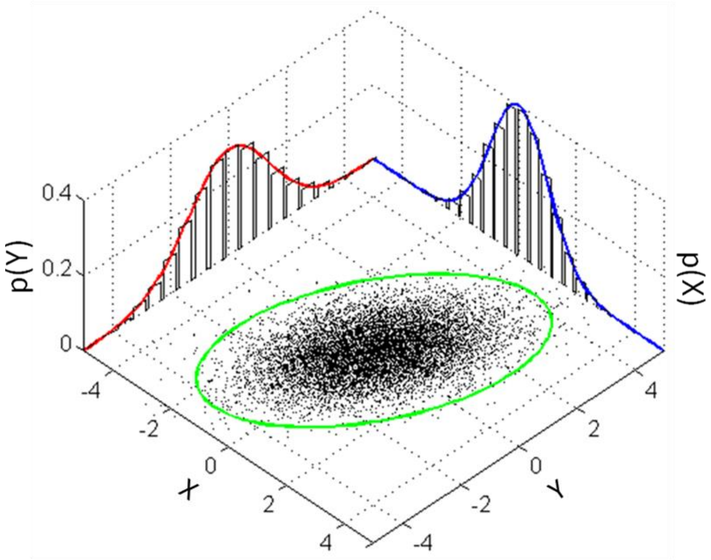
$$p(x) = f_{\mu,\sigma^2}(x) = \frac{1}{\sigma\sqrt{2\pi}} exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right)$$

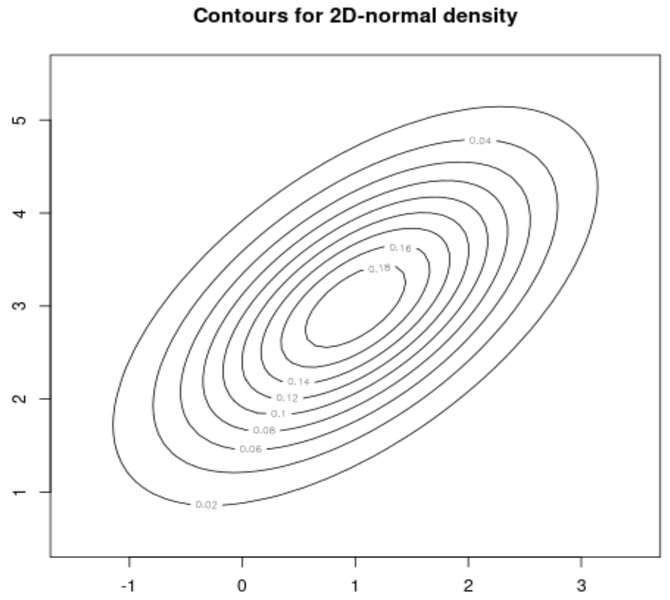


This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Multidimensional Gaussian Distributions







This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

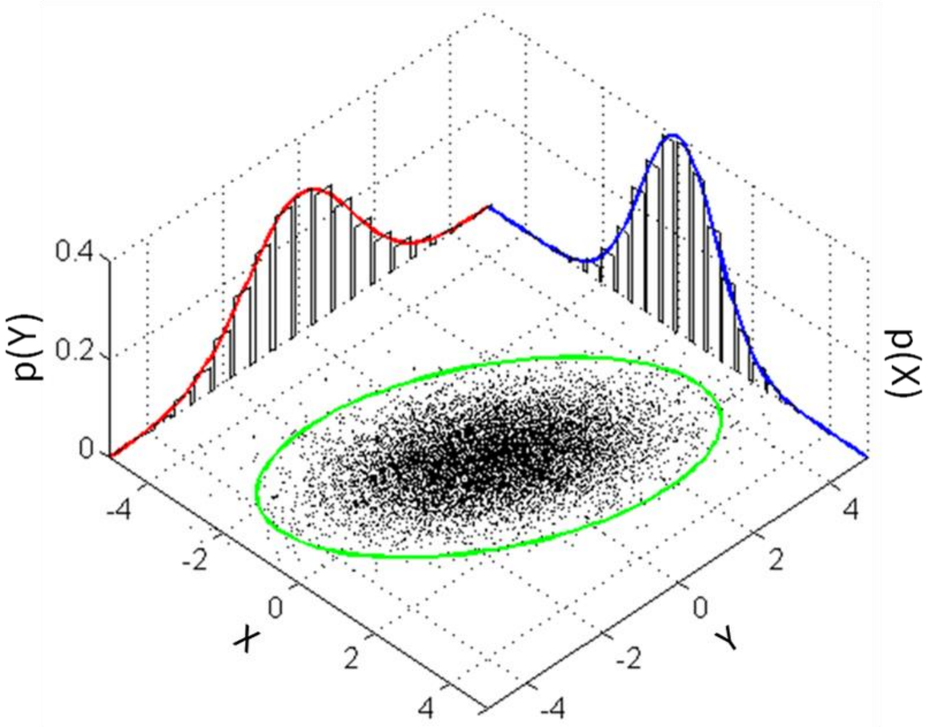
Density estimation in multidimensional case

Training set: $D = \{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}\}$
Each $\boldsymbol{x}^{(i)} \in \mathbb{R}^n$ (has n features)

$$p(\boldsymbol{x}) = p(x_1; \mu_1, \sigma_1^2) * p(x_2; \mu_2, \sigma_2^2) * \cdots * p(x_n; \mu_n, \sigma_n^2)$$

We multiply because the probability of x_1 taking a particular value AND x_2 taking a particular value becomes lower.

$$p(\boldsymbol{x}) = \prod_{j=1}^{n} p\left(x_j; \mu_j, \sigma_j^2\right)$$

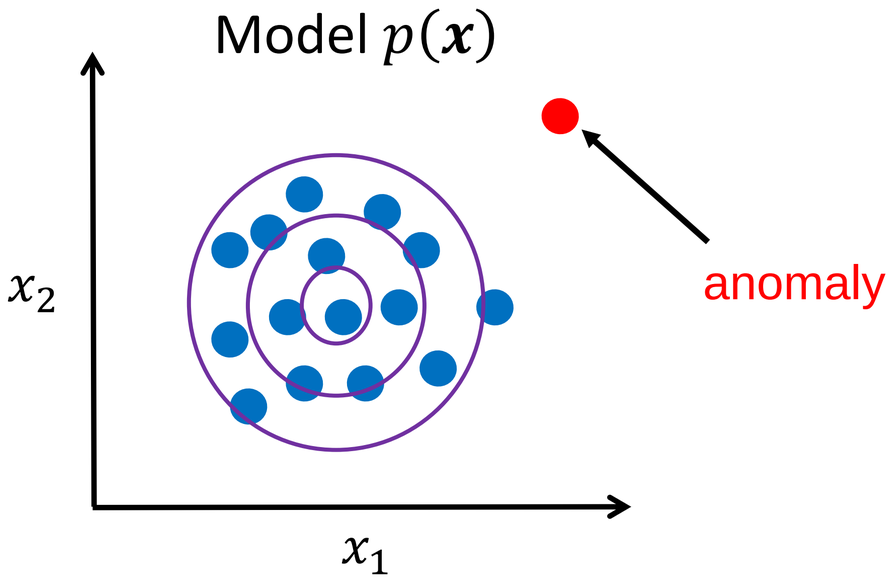


This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Anomaly detection using density estimation

Dataset $D = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$

Is $\boldsymbol{x}^{(new)}$ anomalous?



$p(\boldsymbol{x}^{(new)}) < \varepsilon$ anomaly

$p(\boldsymbol{x}^{(new)}) \geq \varepsilon$ normal

In order to model $p(x)$ we will use a Gaussian distribution

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Anomaly Detection Algorithm

Given dataset $D = \{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}\}$

1. Choose n features that might be indicative of anomalous examples

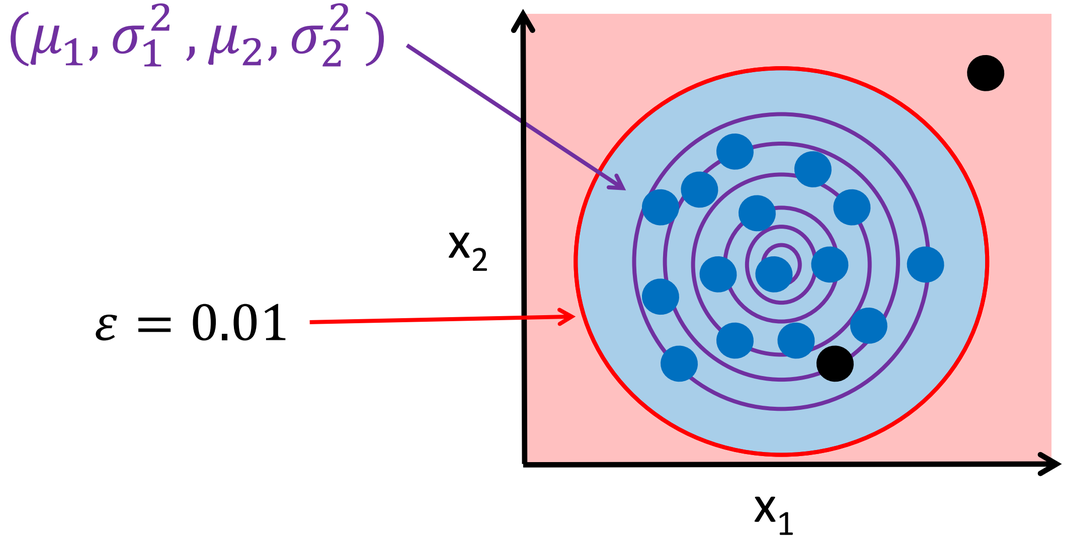
2. Fit parameters $\mu_1, \dots, \mu_n, \sigma_1^2, \dots, \sigma_n^2$

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)} \qquad \sigma_j^2 = \frac{1}{m}\sum_{i=1}^{m}\left(x_j^{(i)} - \mu_j\right)^2$$

3. Given new example $\boldsymbol{x}$, compute $p(\boldsymbol{x})$

$$p(\boldsymbol{x}) = \prod_{j=1}^{n} p\left(x_j; \mu_j, \sigma_j^2\right) = \prod_{j=1}^{n} \frac{1}{\sigma_j \sqrt{2\pi}} exp\left(\frac{-(x_j-\mu_j)^2}{2\sigma_j^2}\right)$$

4. If $p(\boldsymbol{x}) < \varepsilon$: x is an anomaly



$$p(\boldsymbol{x}) = f_{\mu_1\sigma_1^2}(x_1) f_{\mu_2\sigma_2^2}(x_2)$$

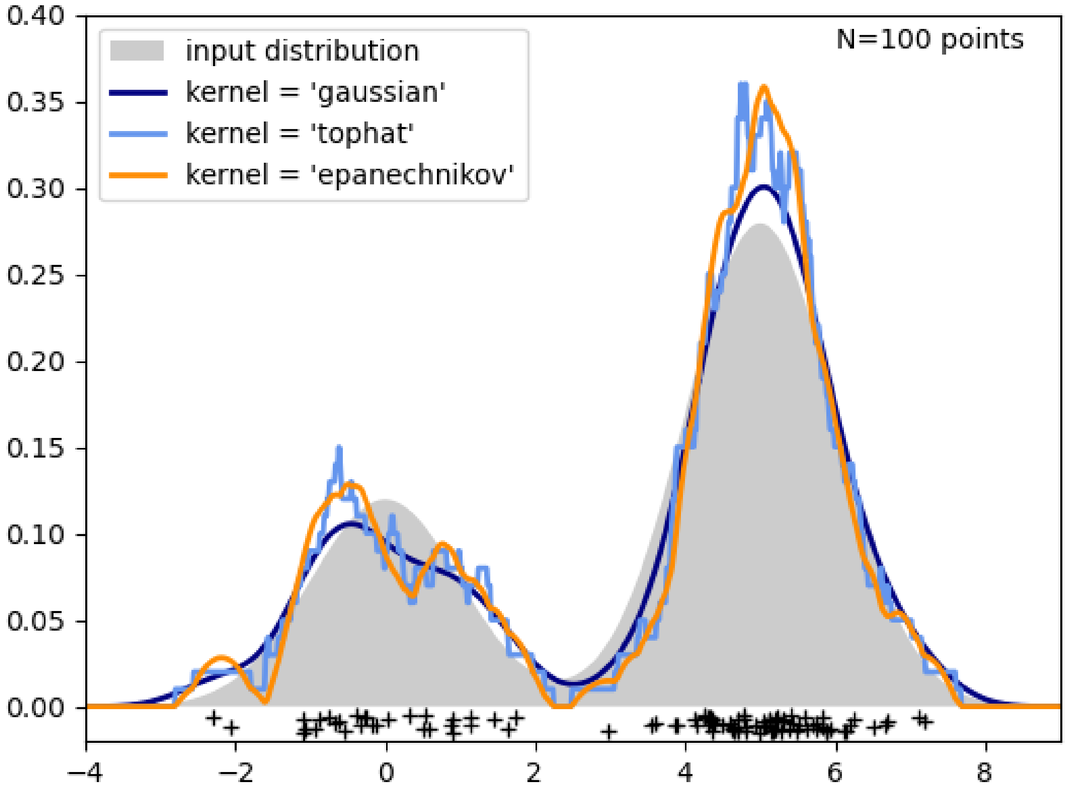
$p(\boldsymbol{x}) = 0.001 < \varepsilon$ anomaly

$p(\boldsymbol{x}) = 0.041 \geq \varepsilon$ normal

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Nonparametric Density Estimation

- What if the data are not Gaussian distributed?
- 100 points drawn from a bimodal distribution
- Parametric density estimation will not estimate the distribution of data well
  - Gaussian will be centred in the middle of the points
- Use **Kernel Density Estimation**



Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

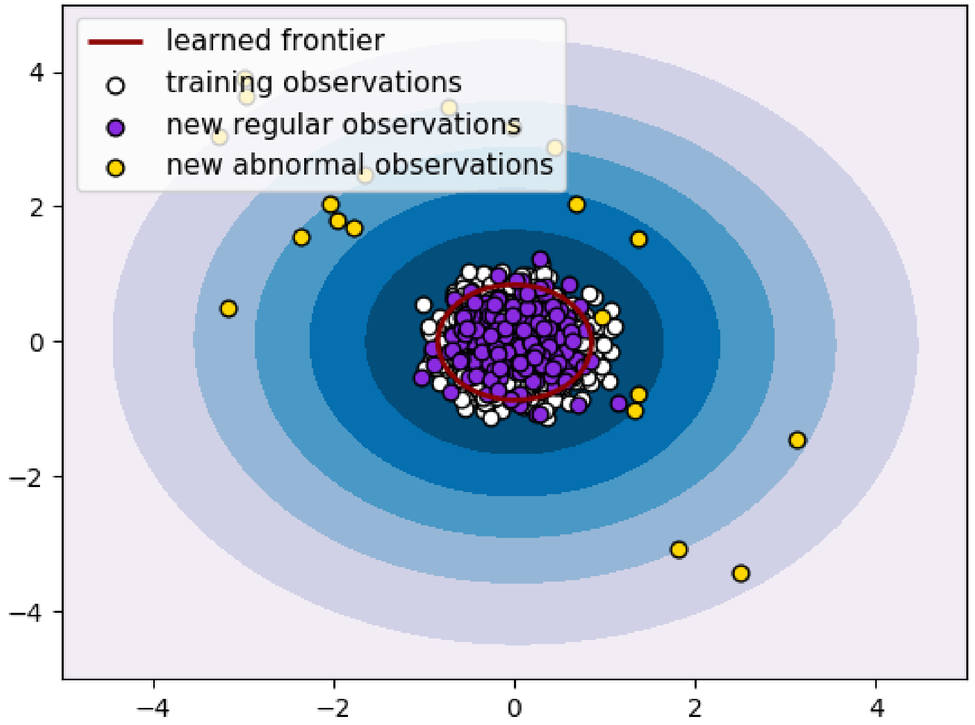
One-class classification

- Common approach for anomaly detection
- Instead of estimating the density we force a **discriminative** model to learn a conservative decision boundary that encompasses the normal points (points of the positive class)
- Typical choices:
  - One-class SVM
  - Isolation forests

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

One-class SVM

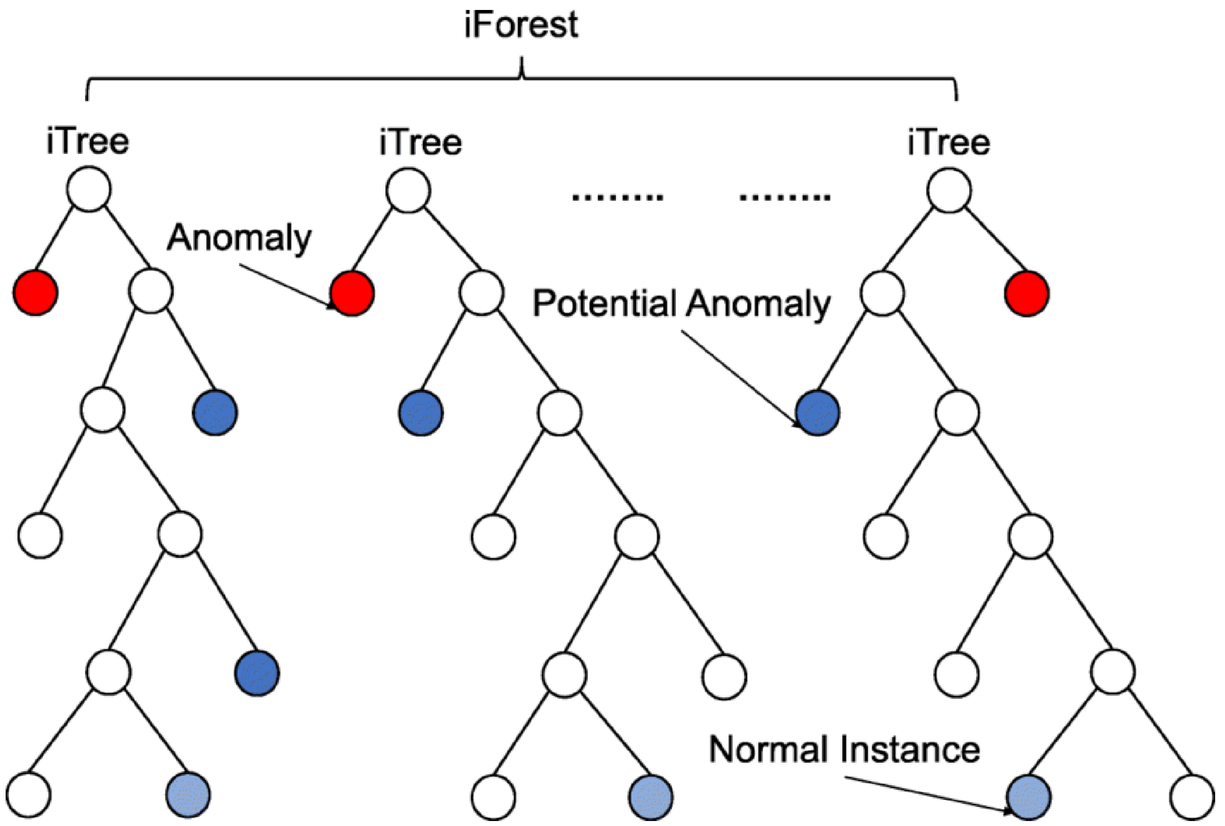
- Try to encompass all instances of the normal class using a **hypersphere**
- Create the smallest possible hypersphere
- Everything outside the hypersphere is considered an anomaly



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

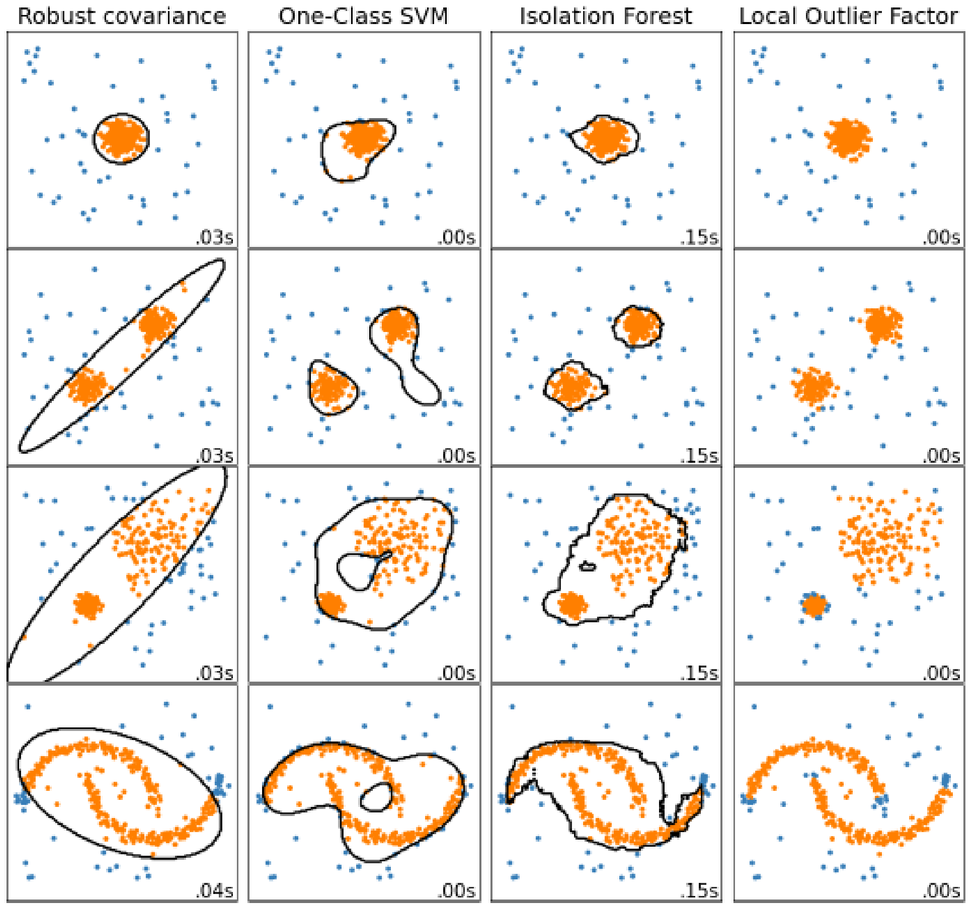
Isolation forests

- Like random forests
- Anomalies = observations with short average path lengths
- Outliers / Anomalies are easier to isolate (short path lengths)
- Inliers / Normal instances are harder to isolate (longer path lengths)



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Outlier detection in scikit-learn

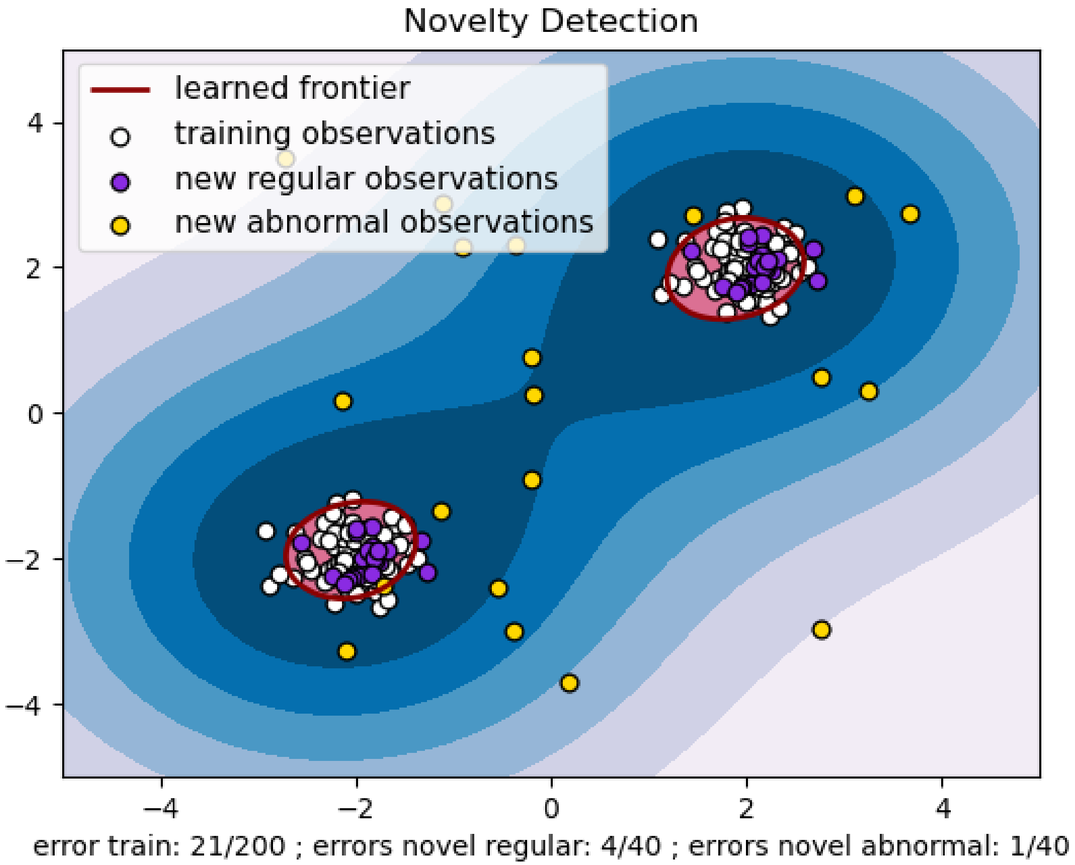


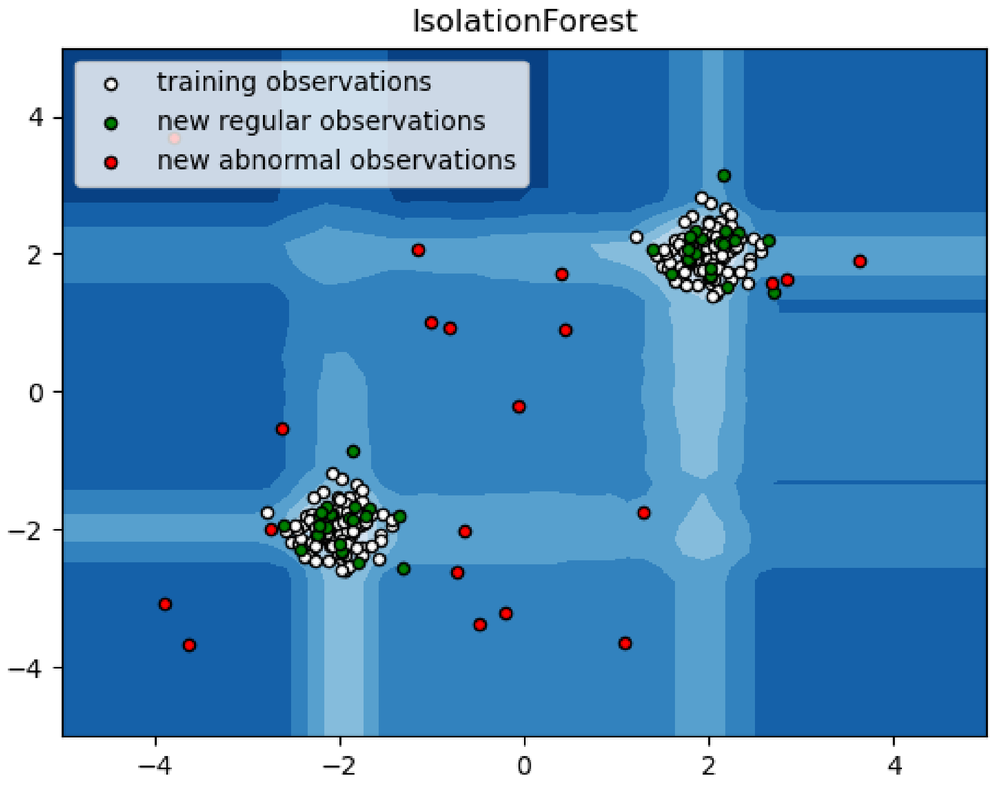
Typically need to set some outlier fraction parameter

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Novelty detection in scikit-learn

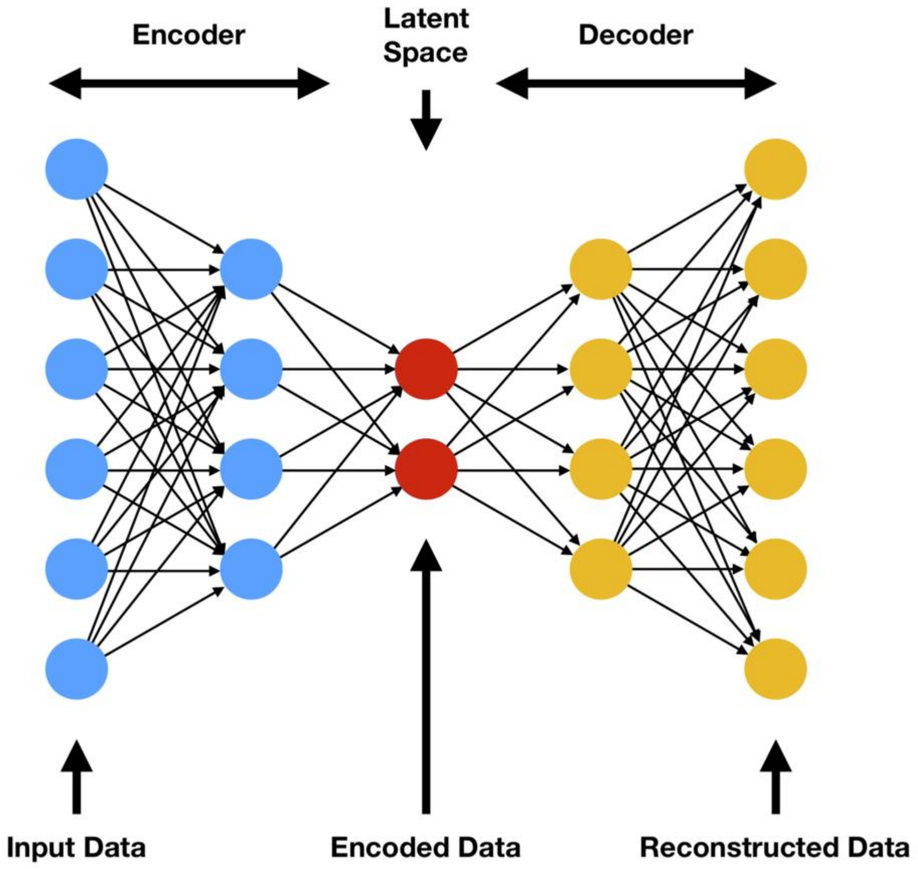




Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Anomaly Detection using Autoencoders



- Autoencoders provide a way of finding regularities in the data, by compressing the input and reconstructing it.
- They are trained to minimize the reconstruction error for a given dataset.
- Key idea behind using an autoencoder for anomaly detection: normal data will have low reconstruction error, while **abnormal data will have higher reconstruction error**.

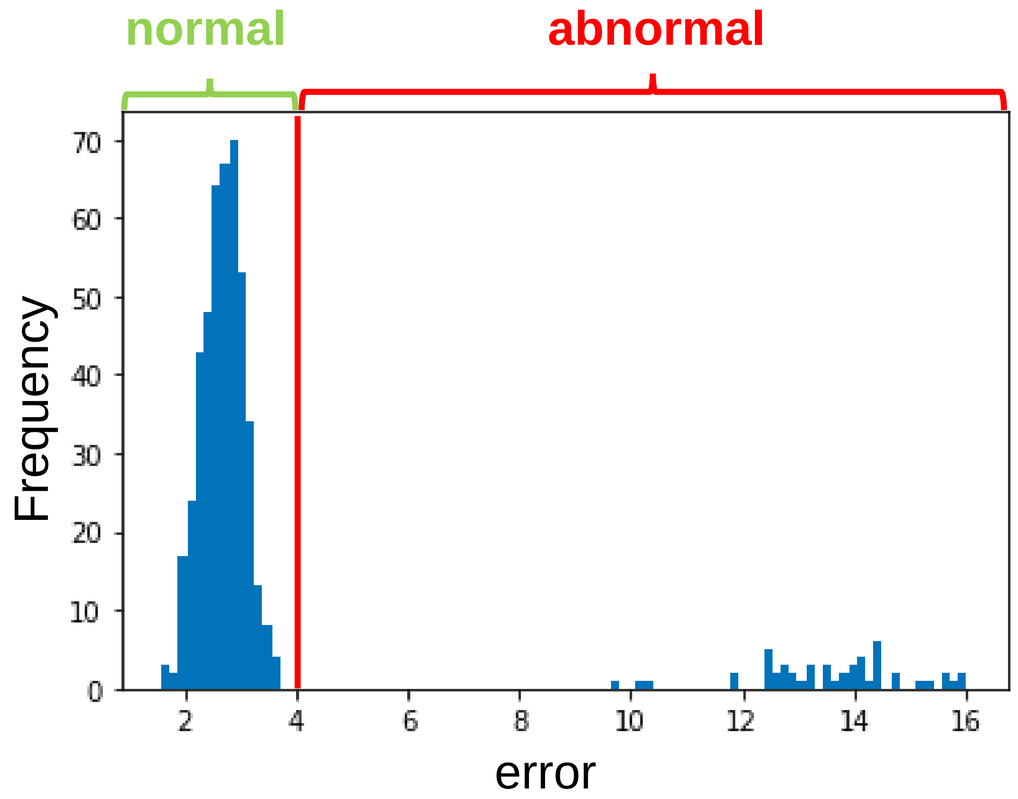
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Anomaly Detection using Autoencoders

Procedure:

- Given a dataset $D = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$
- Train an autoencoder to codify and reconstruct each input vector $\boldsymbol{x}^{(i)}$
- After training, use a histogram to visualize the frequency of the errors
- Establish the decision threshold based on the histogram (e.g., 4)
- Future data that have a reconstruction error:
  - smaller than the threshold are classified as **normal**
  - larger than the threshold are classified as **abnormal**



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Anomaly Detection Example

Transaction fraud detection: A fraudster will typically try to abuse the card as much as possible in a short period of time before the card is detected and suspended.

Merchant id	Merchant category code	Merchant city	Time	Transaction method	Transaction type	Amount
K2203	BC	LIMASSOL	9:02	Magnetic	Retail	100.10
L3425	GD	NICOSIA	9:10	Magnetic	Retail	40.10
F3928	VS	NICOSIA	10:20	Chip	Retail	5.10
W9843	TY	PAPHOS	13:20	Magnetic	Internet	200.00

Importance of feature engineering:

travel_speed = distance between cities / time difference between two adjacent transactions
travel_speed = 85 km / 8 min = 10.625 km/min = 637.5 km/h !!

Adapted from here

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Evaluating an anomaly detection system

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Evaluating an anomaly detection system

- When developing an anomaly detection system (e.g., features, models, algorithms) having a way to evaluate its performance would make decisions much easier

- Assume we have a small amount of labelled data
  - y=1: anomalous
  - y=0: normal

- Training set should have only normal (non-anomalous; y=0) data: $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(m)}$
- Cross-validation and test sets should contain labelled data:
  - $\left(x_{cv}^{(1)}, y_{cv}^{(1)}\right), \ldots, \left(x_{cv}^{(m_{cv})}, y_{cv}^{(m_{cv})}\right)$, $\left(x_{test}^{(1)}, y_{test}^{(1)}\right), \ldots, \left(x_{test}^{(m_{test})}, y_{test}^{(m_{test})}\right)$
  - Mostly normal examples (y=0)
  - A few anomalous examples (y=1)

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Engine monitoring example

Still unsupervised learning because training set contains unlabelled examples

10000 Good (normal) engines
20 Flawed (anomalous) engines

Training set: 6000 good engines
CV: 2000 good engines (y=0) 10 anomalous (y=1) → tune hyperparameters
Test: 2000 good engines (y=0) 10 anomalous (y=1) → evaluate

Alternative: No test set → Only use if very few labelled anomalous examples

Training set: 6000 good engines
CV: 4000 good engines (y=0) 20 anomalous (y=1) → tune hyperparameters, but higher risk of overfitting

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Anomaly detection system evaluation

- Fit model (e.g., $p(\boldsymbol{x})$) on training set $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(m)}$
- On a cross validation / test example $\boldsymbol{x}$, predict
  - y=1 if anomaly (e.g., if $p(\boldsymbol{x}) < \varepsilon$)
  - y=0 if normal (e.g., if $p(\boldsymbol{x}) \leq \varepsilon$)
- Like binary classification, so possible evaluation metrics:
  - True positive, false positive, false negative, true negative rates
  - Precision / recall
  - F1-score
  - AUC score
- Use cross validation set to tune hyperparameters

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 15: Recommender Systems

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 15: Recommender Systems

Learning Outcomes

You will learn about:

1. What recommender systems are and the problem they solve
2. The collaborative filtering algorithm
3. How to extend the algorithm in the case of binary labels
4. Implementation details that would make the algorithm work better in practice
5. Content-based filtering
6. How to find related items among a large set

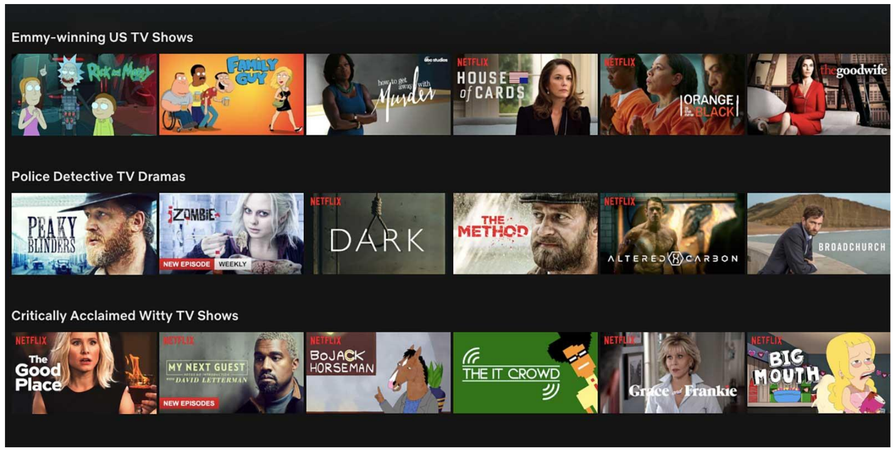
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Recommendations

Many companies drive their value by understanding user preferences and providing recommendations.

Recommendations may influence the news we read, the products we purchase and the entertainment we consume.



amazon.com **Recommended** for You

Amazon.com has new recommendations for you based on items you purchased or told us you own.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Recommender systems

A recommender system is a system that provides suggestions for items that are most relevant to a particular user

Particularly useful when a user needs to choose an item from a large catalogue of items a service may offer

Examples:

- Playlist generators for video and music services
- Product recommenders for online stores
- Content recommenders for social media platforms
- Travel recommenders
- Search results
- …

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Predicting movie ratings

User rates movies using zero to five stars

Movie	Kate (1)	John (2)	Annie (3)	Steve (4)
Alone in the forest	5	5	0	0
Afraid of the dark	5	?	?	0
The black cave	?	4	0	?
Billy's night off	0	0	5	4
The animal whisperer	0	0	5	?

Ratings				
★				
★	★			
★	★	★		
★	★	★	★	
★	★	★	★	★

n_u = no. of users
n_m = no. of movies
$r(i,j) = 1$ if user j has rated movie i

$n_u = 4$
$n_m = 5$

$r(1,1) = 1$
$r(3,1) = 0$

$y^{(3,2)} = 4$

$y^{(i,j)}$ = rating given by user j to movie i (defined only if $r(i,j) = 1$)

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

What if we have features of the movies?

$n_u = 4$
$n_m = 5$
$n = 2$

Movie	Kate (1)	John (2)	Annie (3)	Steve (4)	x_1 (thriller)	x_2 (comedy)
Alone in the forest	5	5	0	0	0.9	0
Afraid of the dark	5	?	?	0	1.0	0.01
The black cave	?	4	0	?	0.99	0
Billy's night off	0	0	5	4	0.1	1.0
The animal whisperer	0	0	5	?	0	0.9

$$\boldsymbol{x}^{(1)} = \begin{bmatrix} 1 \\ 0.9 \\ 0 \end{bmatrix} \qquad \boldsymbol{x}^{(4)} = \begin{bmatrix} 1 \\ 0.1 \\ 1.0 \end{bmatrix}$$

For user 1: Predict rating for movie i as: $\boldsymbol{\theta} \cdot \boldsymbol{x}^{(i)}$ ← just linear regression

$\boldsymbol{\theta}^{(1)} = [0, 5, 0]^T$ $\boldsymbol{x}^{(3)} = [1, 0.99, 0]^T$ $\boldsymbol{\theta}^{(1)} \cdot \boldsymbol{x}^{(3)} = 4.95$

Different linear regression model for each user

For user j: Predict user j's rating for movie i as: $\boldsymbol{\theta}^{(j)} \cdot \boldsymbol{x}^{(i)}$

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Cost function

Notation:

$r(i,j) = 1$ if user j has rated movie i (0 otherwise)
$y^{(i,j)}$ = rating given by user j to movie i (if defined)
$\boldsymbol{\theta}^{(j)}$ = parameters for user j $\quad \boldsymbol{\theta}^{(j)} \in \mathbb{R}^{n+1}$
$\boldsymbol{x}^{(i)}$ = feature vector for movie i

For user j and movie i, predict rating: $f_{\boldsymbol{\theta}^{(j)}}(\boldsymbol{x}^{(i)}) = \boldsymbol{\theta}^{(j)} \cdot \boldsymbol{x}^{(i)}$
$m^{(j)}$ = no. of movies rated by user j
To learn $\boldsymbol{\theta}^{(j)}$

$m^{(j)}$ is just a constant so if we remove it we still end up at the same solution for $\boldsymbol{\theta}^{(j)}$

number of features

$$\min_{\boldsymbol{\theta}^{(j)}} \; L(\boldsymbol{\theta}^{(j)}) = \frac{1}{2m^{(j)}} \sum_{i:r(i,j)=1} \left(f_{\boldsymbol{\theta}^{(j)}}(\boldsymbol{x}^{(i)}) - y^{(i,j)}\right)^2 + \frac{\lambda}{2m^{(j)}} \sum_{k=1}^{n} \left(\theta_k^{(j)}\right)^2$$

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Cost function

To learn $\boldsymbol{\theta}^{(j)}$ (parameters for user j):

$$L\left(\boldsymbol{\theta}^{(j)}\right) = \frac{1}{2} \sum_{i:r(i,j)=1} \left(f_{\boldsymbol{\theta}^{(j)}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{k=1}^{n} \left(\theta_k^{(j)}\right)^2$$

To learn parameters $\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \ldots, \boldsymbol{\theta}^{(n_u)}$ for all users:

$$L\left(\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \ldots, \boldsymbol{\theta}^{(n_u)}\right) = \frac{1}{2} \sum_{j=1}^{n_u} \sum_{i:r(i,j)=1} \left(f_{\boldsymbol{\theta}^{(j)}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^{n} \left(\theta_k^{(j)}\right)^2$$

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Optimization algorithm

$$\min_{\boldsymbol{\theta}^{(1)},\dots,\boldsymbol{\theta}^{(n_u)}} \underbrace{\frac{1}{2}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}\left(f_{\boldsymbol{\theta}^{(j)}}\left(\boldsymbol{x}^{(i)}\right)-y^{(i,j)}\right)^2+\frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(\theta_k^{(j)}\right)^2}_{L\left(\boldsymbol{\theta}^{(1)},\dots,\boldsymbol{\theta}^{(n_u)}\right)}$$

Gradient descent update:

$$\theta_k^{(j)} = \theta_k^{(j)} - a\left(\sum_{i:r(i,j)=1}\left(f_{\boldsymbol{\theta}^{(j)}}\left(\boldsymbol{x}^{(i)}\right)-y^{(i,j)}\right)\right) \quad \text{(for } k=0\text{)}$$

$$\theta_k^{(j)} = \theta_k^{(j)} - a\underbrace{\left(\sum_{i:r(i,j)=1}\left(f_{\boldsymbol{\theta}^{(j)}}\left(\boldsymbol{x}^{(i)}\right)-y^{(i,j)}\right)x_k^{(i)}+\lambda\theta_k^{(j)}\right)}_{\frac{\partial}{\partial\theta_k^{(j)}}L\left(\boldsymbol{\theta}^{(1)},\dots,\boldsymbol{\theta}^{(n_u)}\right)} \quad \text{(for } k\neq 0\text{)}$$

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Collaborative Filtering

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

What if we don't know the features?

Movie	Kate (1)	John (2)	Annie (3)	Steve (4)	x_1 ~~(thriller)~~	x_2 ~~(comedy)~~
Alone in the forest	5	5	0	0	?	?
Afraid of the dark	5	?	?	0	?	?
The black cave	?	4	0	?	?	?
Billy's night off	0	0	5	4	?	?
The animal whisperer	0	0	5	?	?	?

Assume that we somehow learned parameters for the users

Can we compute values for the features from data?

$$\boldsymbol{\theta}^{(1)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix} \quad \boldsymbol{\theta}^{(2)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix} \quad \boldsymbol{\theta}^{(3)} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix} \quad \boldsymbol{\theta}^{(4)} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}$$

Using $\boldsymbol{\theta}^{(j)} \cdot \boldsymbol{x}^{(i)}$

$$\left.\begin{aligned} \boldsymbol{\theta}^{(1)} \cdot \boldsymbol{x}^{(1)} &\approx 5 \\ \boldsymbol{\theta}^{(2)} \cdot \boldsymbol{x}^{(1)} &\approx 5 \\ \boldsymbol{\theta}^{(3)} \cdot \boldsymbol{x}^{(1)} &\approx 0 \\ \boldsymbol{\theta}^{(4)} \cdot \boldsymbol{x}^{(1)} &\approx 0 \end{aligned}\right\} \quad \boldsymbol{x}^{(1)} = \begin{bmatrix} 1 \\ 1.0 \\ 0.0 \end{bmatrix}$$

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Cost function

Given $\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \dots, \boldsymbol{\theta}^{(n_u)}$

To learn $\boldsymbol{x}^{(i)}$:

$$\min_{\boldsymbol{x}^{(i)}} \quad L\left(\boldsymbol{x}^{(i)}\right) = \frac{1}{2} \sum_{j:r(i,j)=1} \left(\boldsymbol{\theta}^{(j)} \cdot \boldsymbol{x}^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{k=1}^{n} \left(x_k^{(i)}\right)^2$$

To learn $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots, \boldsymbol{x}^{(n_m)}$:

$$\min_{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n_m)}} \quad L\left(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots, \boldsymbol{x}^{(n_m)}\right) = \frac{1}{2} \sum_{i=1}^{n_m} \sum_{j:r(i,j)=1} \left(\boldsymbol{\theta}^{(j)} \cdot \boldsymbol{x}^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^{n} \left(x_k^{(i)}\right)^2$$

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Collaborative filtering

Repeat:

Given $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots, \boldsymbol{x}^{(n_m)}$ and movie ratings ($r(i,j)$, $y^{(i,j)}$)

Estimate $\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \dots, \boldsymbol{\theta}^{(n_u)}$

Given $\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \dots, \boldsymbol{\theta}^{(n_u)}$

Estimate $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots, \boldsymbol{x}^{(n_m)}$

Guess $\boldsymbol{\theta} \rightarrow \boldsymbol{x} \rightarrow \boldsymbol{\theta} \rightarrow \boldsymbol{x} \rightarrow \boldsymbol{\theta} \rightarrow \boldsymbol{x} \rightarrow \boldsymbol{\theta} \rightarrow \boldsymbol{x} \rightarrow \cdots$

This works, but instead of going back and forth there is a more efficient way to solve for θ and x **simultaneously**

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Collaborative filtering

Cost function to learn $\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \dots, \boldsymbol{\theta}^{(n_u)}$:

$$L\left(\boldsymbol{\theta}^{(1)}, \dots, \boldsymbol{\theta}^{(n_u)}\right) = \frac{1}{2}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}\left(\boldsymbol{\theta}^{(j)} \cdot \boldsymbol{x}^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(\theta_k^{(j)}\right)^2$$

~~$x_0 = 1$~~

$\boldsymbol{x} \in \mathbb{R}^n$

$\boldsymbol{\theta} \in \mathbb{R}^n$

Cost function to learn $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots, \boldsymbol{x}^{(n_m)}$:

$$L\left(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n_m)}\right) = \frac{1}{2}\sum_{i=1}^{n_m}\sum_{j:r(i,j)=1}\left(\boldsymbol{\theta}^{(j)} \cdot \boldsymbol{x}^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}\left(x_k^{(i)}\right)^2$$

Combine them (minimize $\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n_m)}$ and $\boldsymbol{\theta}^{(1)}, \dots, \boldsymbol{\theta}^{(n_u)}$ simultaneously):

$$L\begin{pmatrix}\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n_m)}, \\ \boldsymbol{\theta}^{(1)}, \dots, \boldsymbol{\theta}^{(n_u)}\end{pmatrix} = \frac{1}{2}\sum_{(i,j):r(i,j)=1}\left(\boldsymbol{\theta}^{(j)} \cdot \boldsymbol{x}^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(\theta_k^{(j)}\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}\left(x_k^{(i)}\right)^2$$

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Collaborative filtering algorithm

1. Initialize $\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n_m)}, \boldsymbol{\theta}^{(1)}, \dots, \boldsymbol{\theta}^{(n_u)}$ to small random values
2. Minimize $L\left(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n_m)}, \boldsymbol{\theta}^{(1)}, \dots, \boldsymbol{\theta}^{(n_u)}\right)$ using gradient descent

   Repeat:

   For every $i = 1, \dots, n_m, and\ j = 1, \dots, n_u$:

$$x_k^{(i)} = x_k^{(j)} - a \frac{\partial}{\partial x_k^{(i)}} L\left(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n_m)}, \boldsymbol{\theta}^{(1)}, \dots, \boldsymbol{\theta}^{(n_u)}\right)$$

$$\theta_k^{(j)} = \theta_k^{(j)} - a \frac{\partial}{\partial \theta_k^{(j)}} L\left(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n_m)}, \boldsymbol{\theta}^{(1)}, \dots, \boldsymbol{\theta}^{(n_u)}\right)$$

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Collaborative filtering algorithm

1. Initialize $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n_m)}, \boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(n_u)}$ to small random values
2. Minimize $L\left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n_m)}, \boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(n_u)}\right)$ using gradient descent

   Repeat:

   For every $i = 1, \ldots, n_m, and\ j = 1, \ldots, n_u$:

$$x_k^{(i)} = x_k^{(i)} - a\left(\sum_{j:r(i,j)=1}\left(\boldsymbol{\theta}^{(j)} \cdot \boldsymbol{x}^{(i)} - y^{(i,j)}\right)\theta_k^{(j)} + \lambda x_k^{(i)}\right)$$
$$\theta_k^{(j)} = \theta_k^{(j)} - a\left(\sum_{i:r(i,j)=1}\left(\boldsymbol{\theta}^{(j)} \cdot \boldsymbol{x}^{(i)} - y^{(i,j)}\right)x_k^{(i)} + \lambda \theta_k^{(j)}\right)$$

3. For a user with parameters $\boldsymbol{\theta}$ and a movie with (learned) features $\boldsymbol{x}$, predict a rating of $\boldsymbol{\theta}^T \boldsymbol{x}$

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Quiz

Why do we consider collaborative filtering as an unsupervised learning algorithm?

Because the features are not provided by a teacher.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Binary Labels

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Binary labels

Movie	Kate (1)	John (2)	Annie (3)	Steve (4)
Alone in the forest	1	1	0	0
Afraid of the dark	1	?	?	0
The black cave	?	1	0	?
Billy's night off	0	0	1	1
The animal whisperer	0	0	1	?

- How can we extend the collaborative filtering algorithm to work for binary labels?
- We want to predict whether the users will like a particular item they have not yet rated.
- We would like to estimate the probability of liking an item in order to decide how much to recommend these items.

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example applications

1. Online shopping website
   - Did user j purchase an item after being shown?
2. Social media
   - Did user j like an item?
3. Online advertising
   - Did user j click on an item?
4. Use the user behavior, instead of asking for explicit rating
   - Did user j spend at least 30sec with an item?

Meaning of ratings:
1: engaged after being shown item
0: did not engage after being shown item
?: item not yet shown

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

From regression to binary classification

Previously:

Predict $y^{(i,j)}$ as $\boldsymbol{\theta}^{(j)} \cdot \boldsymbol{x}^{(i)}$

For binary labels:

Predict that the probability of $y^{(i,j)} = 1$ is given by $g\left(\boldsymbol{\theta}^{(j)} \cdot \boldsymbol{x}^{(i)}\right)$

Logistic regression model

where $g(z) = \frac{1}{1+e^{-z}}$

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Cost function for binary application

Need to modify the cost function from the squared error to the binary cross-entropy error

Previous cost function:

$$L\left(\begin{matrix}\boldsymbol{x}^{(1)},\dots,\boldsymbol{x}^{(n_m)},\\ \boldsymbol{\theta}^{(1)},\dots,\boldsymbol{\theta}^{(n_u)}\end{matrix}\right)=\frac{1}{2}\sum_{(i,j):r(i,j)=1}\left(\overbrace{\boldsymbol{\theta}^{(j)}\cdot\boldsymbol{x}^{(i)}}^{\hat{y}^{(i,j)}}-y^{(i,j)}\right)^2+\frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(\theta_k^{(j)}\right)^2+\frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}\left(x_k^{(i)}\right)^2$$

Loss for binary labels:

$$\hat{y}^{(i,j)}=g\left(\boldsymbol{\theta}^{(j)}\cdot\boldsymbol{x}^{(i)}\right)$$

$$loss\left(\hat{y}^{(i,j)},y^{(i,j)}\right)=-y^{(i,j)}\log\left(\hat{y}^{(i,j)}\right)-\left(1-y^{(i,j)}\right)\log\left(1-\hat{y}^{(i,j)}\right)$$ ← Loss for single example

$$L\left(\begin{matrix}\boldsymbol{x}^{(1)},\dots,\boldsymbol{x}^{(n_m)},\\ \boldsymbol{\theta}^{(1)},\dots,\boldsymbol{\theta}^{(n_u)}\end{matrix}\right)=\sum_{(i,j):r(i,j)=1}loss\left(\hat{y}^{(i,j)},y^{(i,j)}\right)+\frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(\theta_k^{(j)}\right)^2+\frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}\left(x_k^{(i)}\right)^2$$

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Implementation Details

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Mean Normalization

Movie	Kate (1)	John (2)	Annie (3)	Steve (4)	Emily (5)
Alone in the forest	5	5	0	0	?
Afraid of the dark	5	?	?	0	?
The black cave	?	4	0	?	?
Billy's night off	0	0	5	4	?
The animal whisperer	0	0	5	0	?

- Suppose that a new user arrives who has not rated any movies
- The optimization objective would most likely predict a rating of 0 for all movies, because the new user's ratings do not have any contribution in the squared error function.

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Mean Normalization

$$\text{Ratings} = \begin{bmatrix} 5 & 5 & 0 & 0 & ? \\ 5 & ? & ? & 0 & ? \\ ? & 4 & 0 & ? & ? \\ 0 & 0 & 5 & 4 & ? \\ 0 & 0 & 5 & 0 & ? \end{bmatrix} \quad \mu = \begin{bmatrix} 2.5 \\ 2.5 \\ 2 \\ 2.25 \\ 1.25 \end{bmatrix}$$

$$\text{Normalized Ratings} = \begin{bmatrix} 2.5 & 2.5 & -2.5 & -2.5 & ? \\ 2.5 & ? & ? & -2.5 & ? \\ ? & 2 & -2 & ? & ? \\ -2.25 & -2.25 & 2.75 & 1.75 & ? \\ -1.25 & -1.25 & 3.75 & -1.25 & ? \end{bmatrix}$$

Ratings - μ

For user j on movie i predict: $\boldsymbol{\theta}^{(j)} \cdot \boldsymbol{x}^{(i)} + \mu_i$

New user's predicted ratings would be the mean of the ratings

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Mean Normalization

- Normalizing the rows is the most common case for this application: when a new user arrives, provide them with reasonable ratings
- We could normalize the columns, which is what we would do when a new movie arrives with no ratings
- However, such movies should not be recommended initially to many users, as we do not know much about them, so it makes more sense to normalize the rows.

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Vectorized implementation of predicted ratings

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 & ? \\ 5 & ? & ? & 0 & ? \\ ? & 4 & 0 & ? & ? \\ 0 & 0 & 5 & 4 & ? \\ 0 & 0 & 5 & 0 & ? \end{bmatrix} \in \mathbb{R}^{n_m \times n_u}$$

Predicted ratings: $\begin{bmatrix} \boldsymbol{\theta}^{(1)} \cdot \boldsymbol{x}^{(1)} & \dots & \boldsymbol{\theta}^{(n_u)} \cdot \boldsymbol{x}^{(1)} \\ \vdots & \ddots & \vdots \\ \boldsymbol{\theta}^{(1)} \cdot \boldsymbol{x}^{(n_m)} & \dots & \boldsymbol{\theta}^{(n_u)} \cdot \boldsymbol{x}^{(n_m)} \end{bmatrix}$

$$X\Theta \in \mathbb{R}^{n_m \times n_u}$$

$$\boldsymbol{x}^{(i)} \in \mathbb{R}^{n \times 1}$$

$$\boldsymbol{\theta}^{(j)} \in \mathbb{R}^{n \times 1}$$

$$X = \begin{bmatrix} \left(\boldsymbol{x}^{(1)}\right)^T \\ \vdots \\ \left(\boldsymbol{x}^{(n_m)}\right)^T \end{bmatrix} \in \mathbb{R}^{n_m \times n}$$

$$\Theta = \begin{bmatrix} \boldsymbol{\theta}^{(1)} & \dots & \boldsymbol{\theta}^{(n_u)} \end{bmatrix} \in \mathbb{R}^{n \times n_u}$$

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Finding related items

- When using collaborative filtering, the features $\boldsymbol{x}^{(i)}$ of item i are hard to interpret
  - In the example, we had features such as whether a movie is a thriller or a comedy, however, we cannot easily look at individual features, e.g., x_1 or x_2 and understand what they are about
  - Nonetheless, collectively the features convey some meaning about the item

- To find other items related to an item of interest, we need to rank the items based on their distance to $\boldsymbol{x}^{(i)}$.
  - This is achieved using the k-nearest neighbors of $\boldsymbol{x}^{(i)}$, where k indicates the number of items similar to $\boldsymbol{x}^{(i)}$ we want to retrieve

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Content-based Filtering

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Collaborative filtering vs Content-based filtering

Collaborative filtering:

- Recommend items to you based on ratings of users who gave similar ratings as you

Content-based filtering:

- Recommend items to you based on features of user and item to find good match
- Still have some ratings

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Examples of user and item features

User features:

- Age
- Gender
- Country
- Movies watched
- Average rating per genre
- …

Movie features:

- Year
- Genre(s)
- Reviews
- Average rating
- …

$\boldsymbol{x}_u^{(j)}$ for user j

$\boldsymbol{x}_m^{(i)}$ for movie i

Vector size could be different

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Content-based filtering: learning to match

Compute embeddings from feature vectors:

$$\boldsymbol{v}_u^{(j)} = f_u\left(\boldsymbol{x}_u^{(j)}\right) \qquad \boldsymbol{v}_u^{(j)} \in \mathbb{R}^n$$

$$\boldsymbol{v}_m^{(i)} = f_m\left(\boldsymbol{x}_m^{(i)}\right) \qquad \boldsymbol{v}_m^{(i)} \in \mathbb{R}^n$$

Needs to be of the same size
e.g., $n = 32$

Predict rating of user j on movie i as:

$$\boldsymbol{v}_u^{(j)} \cdot \boldsymbol{v}_m^{(i)}$$

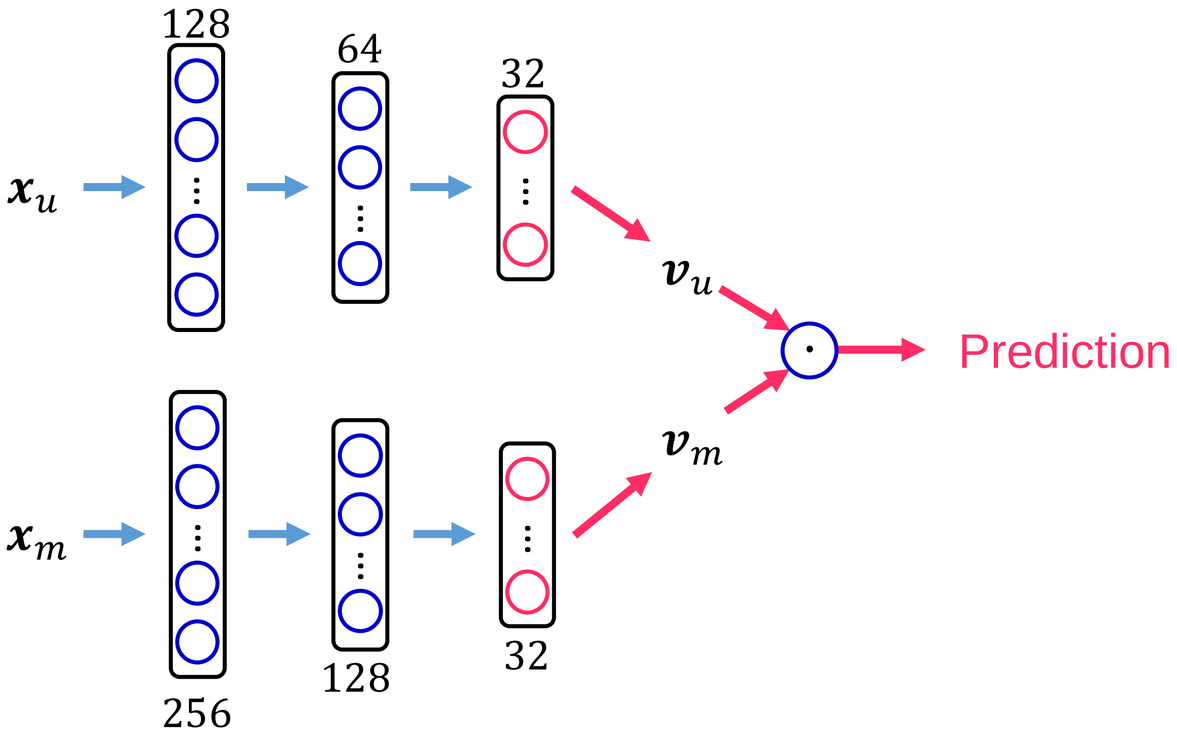
Problem:
How to come up with an appropriate choice of embeddings in order for their **dot product** to be a good prediction of the rating that user j gives to movie i

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Deep Learning for content-based filtering



Cost function

$$L(\ldots) = \sum_{(i,j):r(i,j)=1} \left(\boldsymbol{v}_u^{(j)} \cdot \boldsymbol{v}_m^{(i)} - y^{(i,j)} \right)^2$$

+ NN regularization term

Training

Using backprop and gradient descent

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Finding related items

To find movies related to movie i, find the k-nearest neighbors of $\boldsymbol{v}_m^{(i)}$

- This can be pre-computed

How to efficiently find recommendation from a large set of items (e.g., millions)?

1. Retrieval:
   - Generate large and broad list of plausible item candidates
     - 10 most similar movies to last 10 movies watched by user
     - Top 10 movies for most viewed 3 genres by user
     - Top 20 movies in the country
   - Combine retrieved items into list, remove duplicates and items already watched

2. Ranking:
   - Take list retrieved and rank using the learned model ($\boldsymbol{v}_u^{(j)} \cdot \boldsymbol{v}_m^{(i)}$)
   - Display ranked items to user (e.g., top 10)

Slides based on Andrew Ng's Machine Learning course - Coursera

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Recommendations as a sequential decision making problem

- What we mentioned so far does not explicitly consider the sequential nature of user interaction with a system
  - This is done typically by creating features that try to capture this interaction (e.g., last watched movie)

- However, if we explicitly consider this sequential nature we expect to be able to maximize engagement

- The problem of Recommender Systems can be viewed under the prism of Reinforcement Learning!

- Active research area

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Next Lectures

Part 4: Reinforcement Learning

- Introduction to RL
- Markov Decision Processes & Dynamic Programming

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 16: Introduction to Reinforcement Learning

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Revision

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Anomaly Detection

- Anomaly detection is the problem of modeling a dataset of normal events and raising an alarm when an unusual event occurs in the future.
  - Normal events are assumed to be concentrated
  - Outlier detection: outliers exist in the training set
  - Novelty detection: no outliers in the training set
- Approaches:
  - Density estimation
  - One-class classification using discriminative models
  - Autoencoders
- Density estimation: Parametric and Non-parametric
  - Build a model of the probability of points
  - Use a threshold (ε) on the probability to classify an anomaly (unlikely point) from a normal point
  - Parametric: fit a Gaussian (Parameters: mean and variance)
  - Non-parametric: kernel density estimation

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Anomaly Detection

- One-class classification using discriminative models:
  - Create a conservative decision boundary
  - One-class SVM: try to encompass all (normal) training data using the smallest hypersphere
  - Isolation Forests: anomalies are data points that have short path lengths in a tree
- Autoencoders:
  - Normal data have low reconstruction error
  - Abnormal data have higher reconstruction error
  - Use a histogram of the errors and decide an anomaly threshold
- Feature engineering can be very important in anomaly detection systems
- Supervised anomaly detection: way to evaluate anomaly detection systems
  - Have small amount of labelled data
  - Training set: normal data, no labels
  - CV and test sets: normal+abnormal labelled data
  - Evaluation metrics like in binary classification (TP rate, precision, AUC score,…)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Recommender Systems

- Recommender systems are systems that provide suggestions for items that are most relevant to a particular user
- Matrix completion problem:
  - sparse matrix with lots of missing values
  - predict missing values from the others
- Example: predicting movie ratings (0-5)
  - Rows: movies
  - Columns: users
  - Use different linear regression model for each user (e.g., if 1M users, we have 1M models)
  - When we have the features for each movie we can formulate a cost function for learning the parameters of all users using gradient descent
    - Supervised regression problem using the squared error loss

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Recommender Systems

- Collaborative filtering:
  - Recommend items based on ratings of users who gave similar ratings
  - Unsupervised because it does not assume knowledge of features
  - Formulates a cost function that can be used to simultaneously learn both the features and the parameters for each user using gradient descent
  - For binary labels we can use a logistic regression prediction model and a binary cross-entropy loss
  - When a new user arrives we can use mean normalization so that the predicted ratings of the new user will equal the mean of all ratings for each movie
- Content-based filtering:
  - Recommendation based on features of user and item to find good match
  - Compute embeddings (e.g., using a NN) from features which need to be of the same size
  - Predicted rating: dot product of embeddings
- Finding related items (e.g., movies related to movie i) can be done using a k-nearest neighbor search (in feature or embedding space)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 16: Introduction to Reinforcement Learning

Learning Outcomes

You will learn about:

1. what reinforcement learning (RL) is and how it differs from other types of machine learning
2. how to formalize the RL problem
3. the different agent components and agent categories
4. the different applications of RL

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Brief history of automation

- First, automation of repeated physical solutions
  - Industrial revolution (1750 - 1850) and Machine Age (1870 - 1940)
- Second, automation of repeated mental solutions
  - Digital revolution (1950 - now) and Information Age
- Next step: allow machines to find solutions themselves
  - Artificial Intelligence
  - Only need to specify a problem and/or goal
  - Requires learning autonomously how to make decisions

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

What is artificial intelligence?

- We will use the following definition of intelligence:

To be able to learn to make decisions to achieve goals

- **Learning**, **decisions**, and **goals** are all central

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

What is reinforcement learning?

- People and animals learn by interacting with our environment
- This differs from certain other types of learning
  - It is active rather than passive
  - Interactions are often sequential — future interactions can depend on earlier ones
- We are goal-directed
- We can learn without examples of optimal behaviour
- Instead, we optimise some reward signal
  - Trial-and-error learning

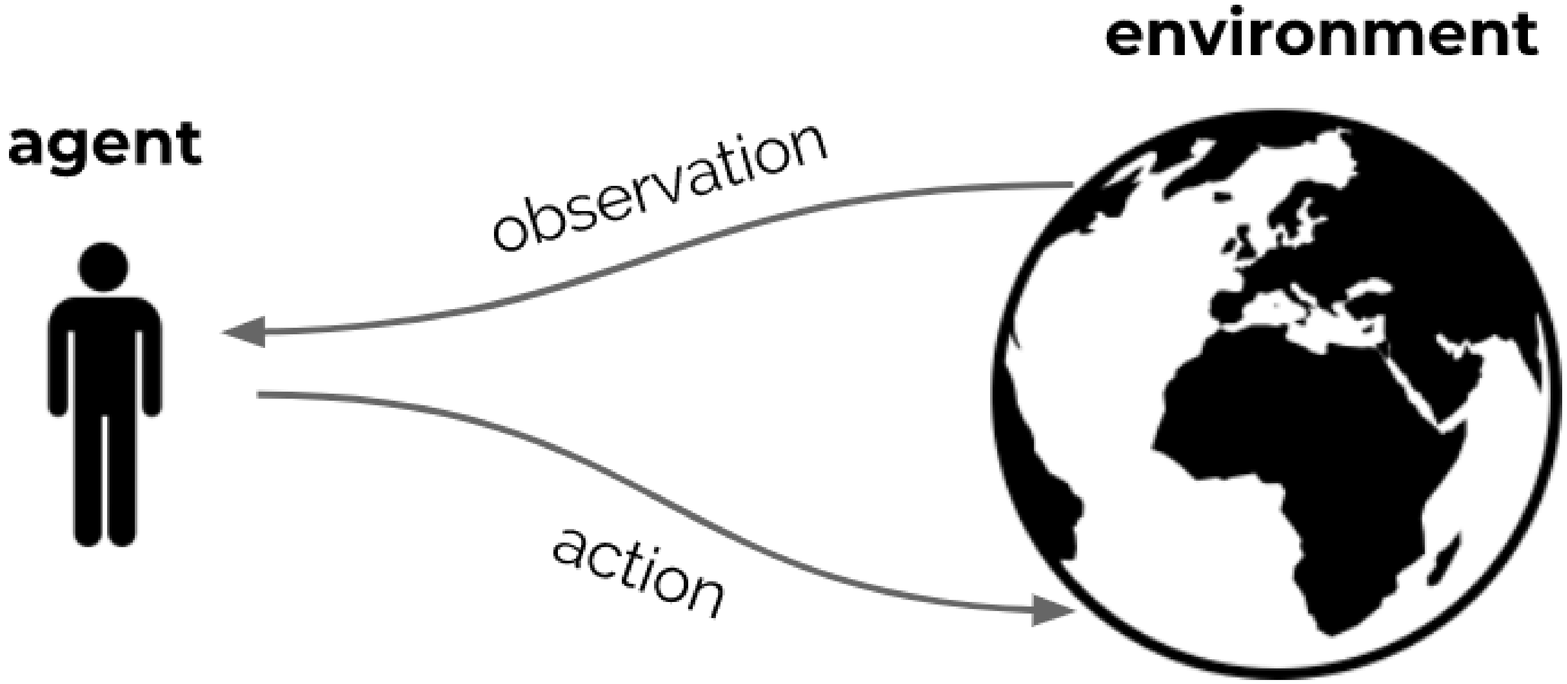
Image Source: ***UC Berkeley CS188 – Intro to AI course***

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

The interaction loop



Goal: optimize sum of rewards, through repeated interaction

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

The reward hypothesis

Reinforcement Learning is based on the reward hypothesis

Any goal can be formalized as the outcome of maximizing a cumulative reward

Can you find a counterexample?

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Examples of RL problems

- Fly a helicopter — Reward: air time, inverse distance, ...
- Manage an investment portfolio — Reward: gains, gains minus risk, ...
- Control a power station — Reward: efficiency, ...
- Make a robot walk — Reward: distance, speed, ...
- Play video or board games — Reward: win, maximize score, ...

If the goal is to learn via interaction, these are all reinforcement learning problems (Irrespective of which solution you use)

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

What is reinforcement learning?

There are distinct reasons to learn:

1. Find solutions
   - A program that plays chess really well
   - A manufacturing robot with a specific purpose
2. Adapt online, deal with unforeseen circumstances
   - A chess program that can learn to adapt to you
   - A robot that can learn to navigate unknown terrains

- Reinforcement learning can provide algorithms for both cases
- Note that the second point is not (just) about generalization — it is about continuing to learn efficiently online, during operation

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

What is reinforcement learning?

- Science and framework of learning to make decisions from interaction
- This requires us to think about
  - time
  - (long-term) consequences of actions
  - actively gathering experience
  - predicting the future
  - dealing with uncertainty

- Huge potential scope
- A formalization of the AI problem

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: Learning to play Atari

Video source

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

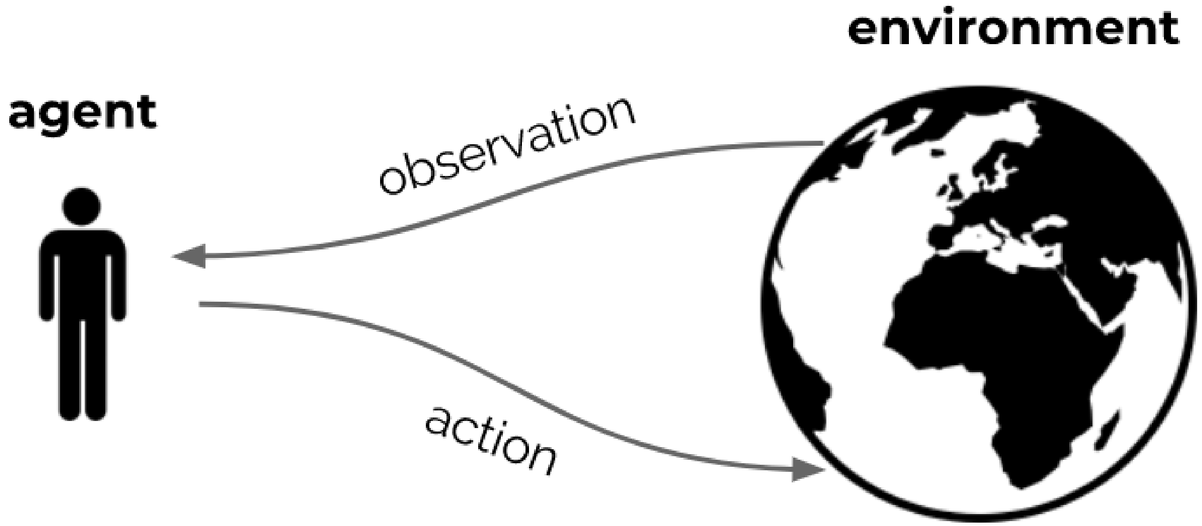
Master programmes in Artificial Intelligence 4 Careers in Europe

Formalizing the RL problem

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Agent and Environment



- At each step *t* the agent:
  - Receives observation O_t (and reward R_t)
  - Executes action A_t
- The environment:
  - Receives action A_t
  - Emits observation O_{t+1} (and reward R_{t+1})

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Rewards

- A **reward** R_t is a scalar feedback signal
- Indicates how well agent is doing at step t — defines the goal
- The agent's job is to maximize cumulative reward

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + \ldots$$

- We call this the **return**

Reinforcement learning is based on the **reward hypothesis**:

Any goal can be formalized as the outcome of maximizing a cumulative reward

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Values

- We call the expected cumulative reward, from a state s , the **value**

$$v(s) = \mathbb{E}\left[G_t \mid S_t = s\right]$$
$$= \mathbb{E}\left[R_{t+1} + R_{t+2} + R_{t+3} + \ldots \mid S_t = s\right]$$

- The value depends on the actions the agent takes
- Goal is to **maximize value**, by picking suitable actions
- Rewards and values define **utility** of states and action (no supervised feedback)
- Returns and values can be defined recursively

$$G_t = R_{t+1} + G_{t+1}$$
$$v(s) = \mathbb{E}\left[R_{t+1} + v(S_{t+1}) \mid S_t = s\right]$$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Maximizing value by taking actions

- Goal: **select actions to maximise value**
- Actions may have long term consequences
- Reward may be delayed
- It may be better to sacrifice immediate reward to gain more long-term reward
- Examples:
  - Refueling a helicopter (might prevent a crash in several hours)
  - Defensive moves in a game (may help chances of winning later)
  - Learning a new skill (can be costly & time-consuming at first)
- A mapping from states to actions is called a **policy**

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Action values

- It is also possible to condition the value on **actions**:

$$q(s,a) = \mathbb{E}\left[G_t \mid S_t = s, A_t = a\right]$$
$$= \mathbb{E}\left[R_{t+1} + R_{t+2} + R_{t+3} + \ldots \mid S_t = s, A_t = a\right]$$

- We will talk in depth about state and action values later

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Core concepts

The reinforcement learning formalism includes

- **Environment** (dynamics of the problem)
- **Reward** signal (specifies the goal)
- **Agent**, containing:
  - Agent state
  - Policy
  - Value function estimate?
  - Model?
- We will now go into the agent

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Inside the Agent

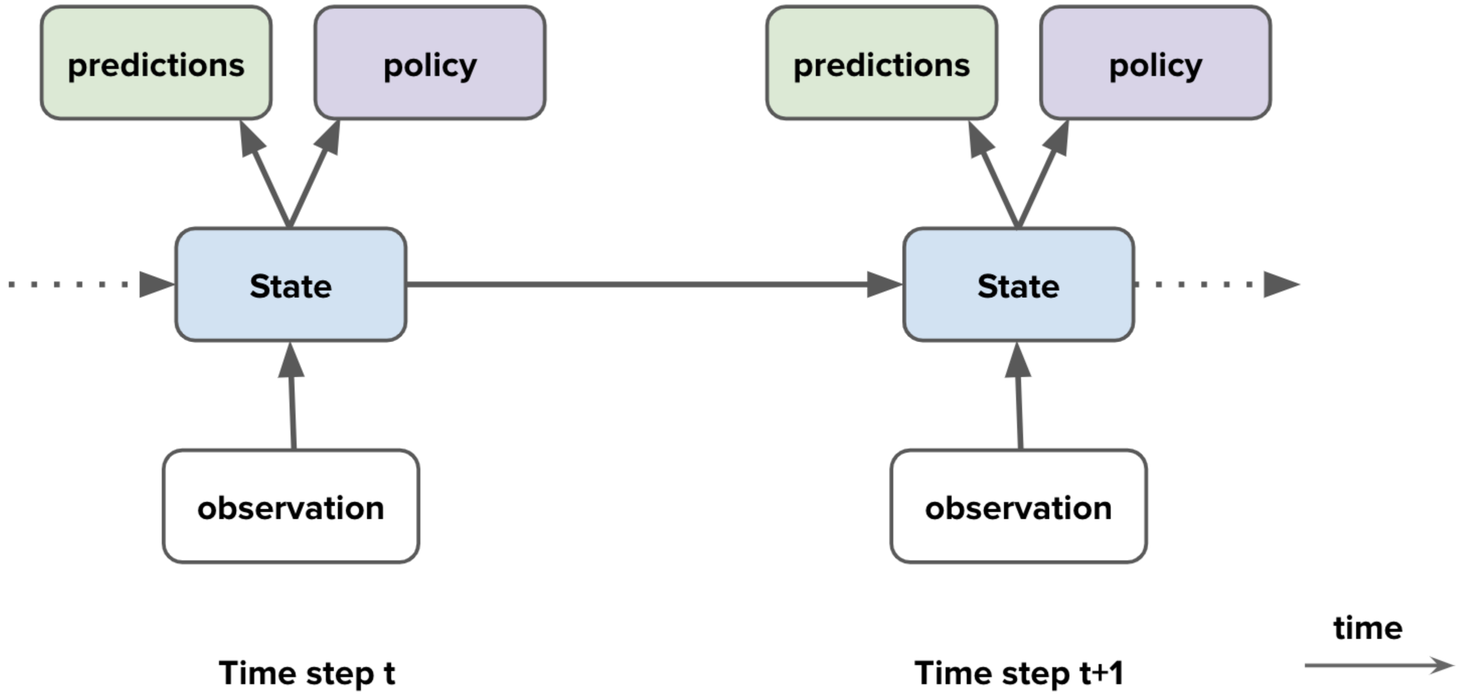
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Agent components

Agent components

- **Agent state**
- Policy
- Value function
- Model

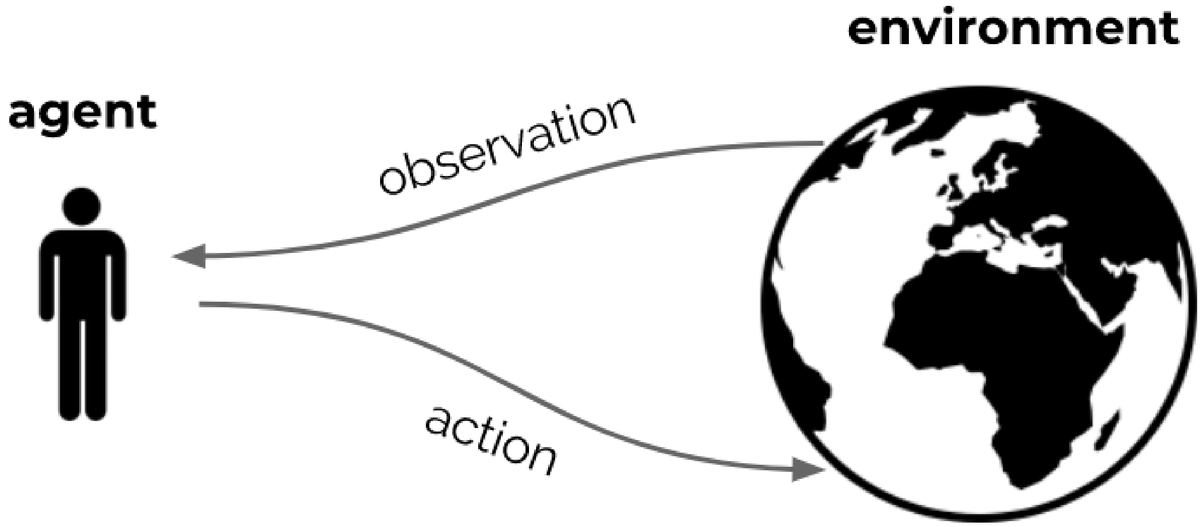


Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Environment state



- The **environment state** is the environment's internal state
- It is usually invisible to the agent
- Even if it is visible, it may contain lots of irrelevant information

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Agent state

- The **history** is the full sequence of observations, actions, rewards

$$\mathcal{H}_t = O_0, A_0, R_1, O_1, \ldots, O_{t-1}, A_{t-1}, R_t, O_t$$

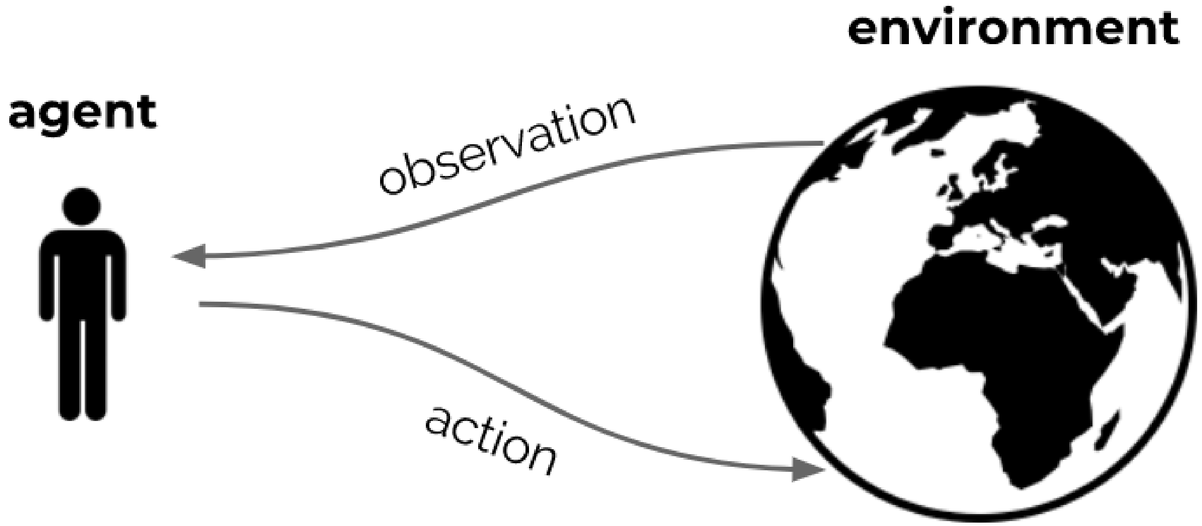
- For instance, the sensorimotor stream of a robot
- This history is used to construct the **agent state** S_t

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Fully Observable Environments



Full observability

Suppose the agent sees the full environment state

- observation = environment state
- The agent state could just be this observation:

$$S_t = O_t = \text{environment state}$$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Markov Decision Processes

Markov decision processes (MDPs) are a useful mathematical framework

Definition: A decision process is Markov if: $p(r, s \mid S_t, A_t) = p(r, s \mid \mathcal{H}_t, A_t)$

- This means that the state contains all we need to know from the history
- Doesn't mean it contains everything, just that adding more history doesn't help
  - Once the state is known, the history may be thrown away

- Typically, the agent state S_t is some compression of H_t
- Note: we use S_t to denote the **agent state**, not the **environment state**

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Partially Observable Environments

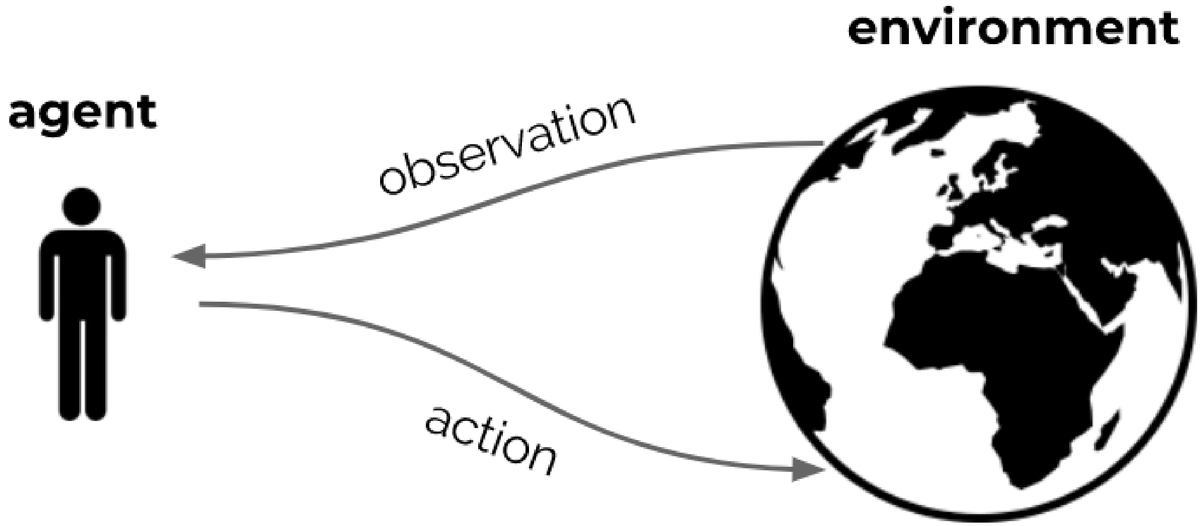
- **Partial observability:** The observations are not Markovian
  - A robot with camera vision isn't told its absolute location
  - A poker playing agent only observes public cards
- Now using the observation as state would not be Markovian
- This is called a **partially observable Markov decision process** (POMDP)
- The **environment state** can still be Markov, but the agent does not know it
- We might still be able to construct a Markov agent state

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Agent State



- The agent's actions depend on its state
- The **agent state** is a function of the history
- For instance, $S_t = O_t$
- More generally:

$$S_{t+1} = u(S_t, A_t, R_{t+1}, O_{t+1})$$

where *u* is a 'state update function'

- The agent state is often **much** smaller than the environment state

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Agent State

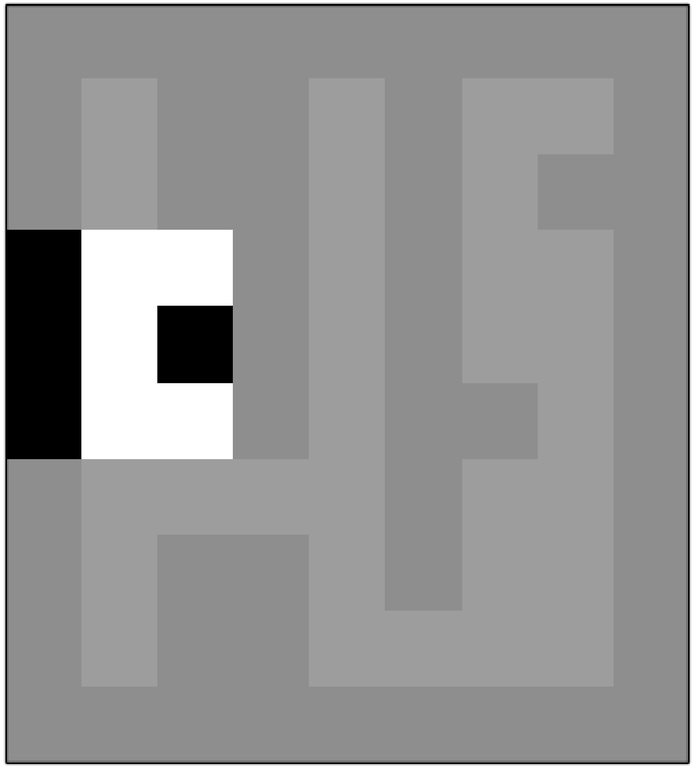
The full environment state of a maze

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Agent State



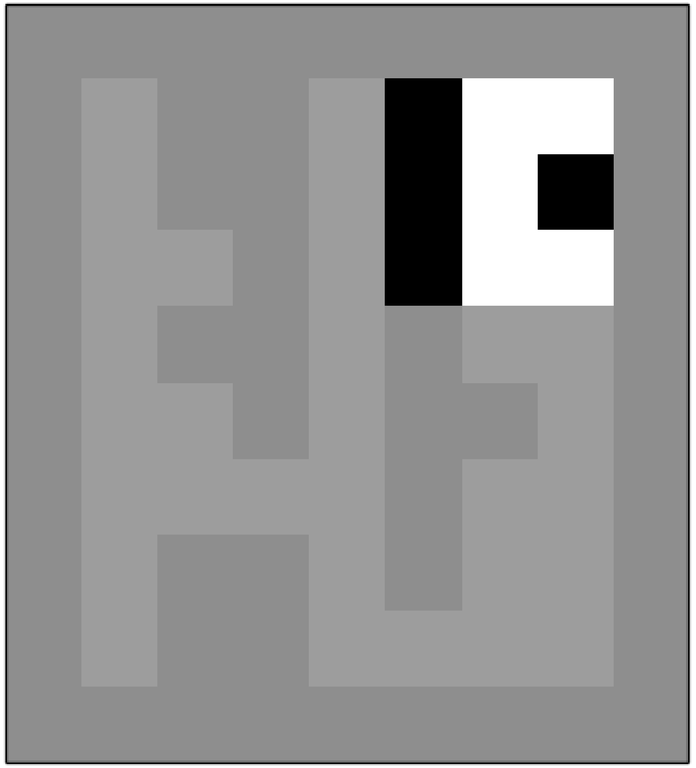
A potential observation

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Agent State



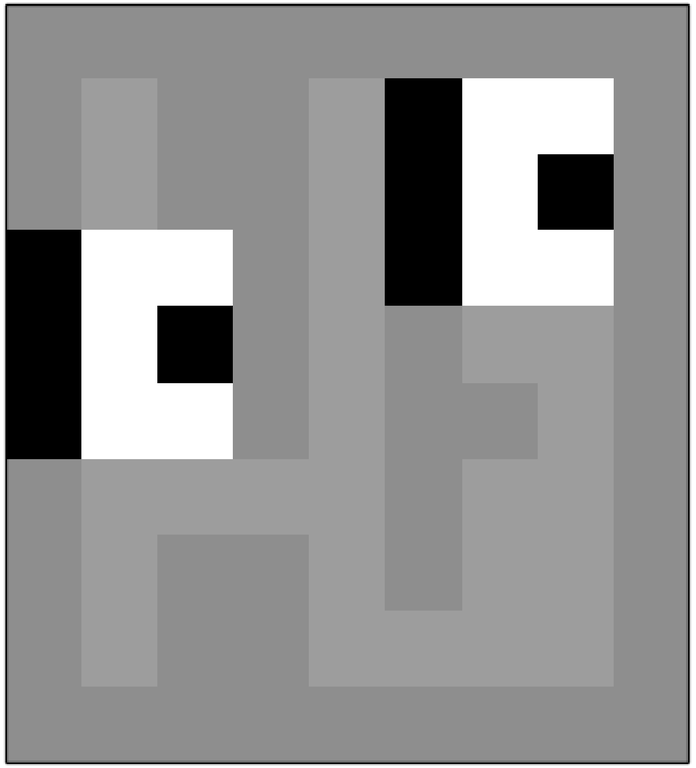
An observation at a different location

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Agent State



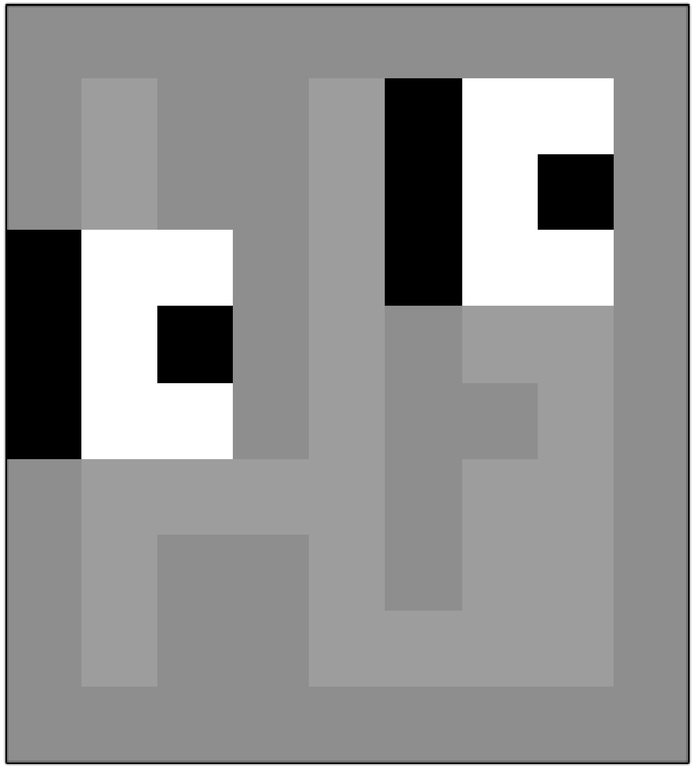
The two observations are indistinguishable

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Agent State



These two states are not Markov

How could you construct a Markov agent state in this maze (for any reward signal)?

What if we take the previous observation into account?

Depends on policy and state transitions (whether they are probabilistic).

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Partially Observable Environments

- To deal with partial observability, agent can construct suitable state representations
- Examples of agent states:
  - Last observation: $S_t = O_t$ (might not be enough)
  - Complete history: $S_t = H_t$ (might be too large)
  - A generic update: $S_t = u(S_{t-1}, A_{t-1}, R_t, O_t)$ (but how to pick/learn u ?)
- Constructing a fully Markovian agent state is often not feasible
- More importantly, the state should allow good policies and value predictions

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Agent components

Agent components

- Agent state
- **Policy**
- Value function
- Model

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Policy

- A **policy** defines the agent's behaviour
- It is a map from agent state to action
- Deterministic policy: $A = \pi(S)$
- Stochastic policy: $\pi(A \mid S) = p(A \mid S)$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Agent components

Agent components

- Agent state
- Policy
- **Value function**
- Model

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Value Function

- The actual value function is the expected return

$$v_\pi(s) = \mathbb{E}\left[G_t \mid S_t = s, \pi\right]$$
$$= \mathbb{E}\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \mid S_t = s, \pi\right]$$

- We introduced a **discount factor** $\gamma \in [0, 1]$
  - Trades off importance of immediate vs long-term rewards
- The value depends on a policy
- Can be used to evaluate the desirability of states
- Can be used to select between actions

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Value Function

- The return has a recursive form $G_t = R_{t+1} + \gamma G_{t+1}$
- Therefore, the value has as well

$$v_\pi(s) = \mathbb{E}\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t \sim \pi(s)\right]$$
$$= \mathbb{E}\left[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t \sim \pi(s)\right]$$

- Here $A_t \sim \pi(s)$ means A_t is chosen by policy π in state s (even if π is deterministic)
- This is known as a **Bellman equation** (Bellman 1957)
- A similar equation holds for the optimal (=highest possible) value:

$$v_*(s) = \max_a \mathbb{E}\left[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a\right]$$

- This does **not** depend on a policy

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Value Function approximations

- Agents often approximate value functions
- We will discuss algorithms to learn these efficiently
- With an accurate value function, we can behave optimally
- With suitable approximations, we can behave well, even in intractably big domains

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Agent components

Agent components

- Agent state
- Policy
- Value function
- **Model**

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model

- A **model** predicts what the environment will do next
- E.g., $\mathcal{P}$ predicts the next state

$$\mathcal{P}(s, a, s') \approx p\left(S_{t+1} = s' \mid S_t = s, A_t = a\right)$$

- E.g., $\mathcal{R}$ predicts the next (immediate) reward

$$\mathcal{R}(s, a) \approx \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right]$$

- A model does not immediately give us a good policy - we would still need to plan
- We could also consider **stochastic (generative)** models

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

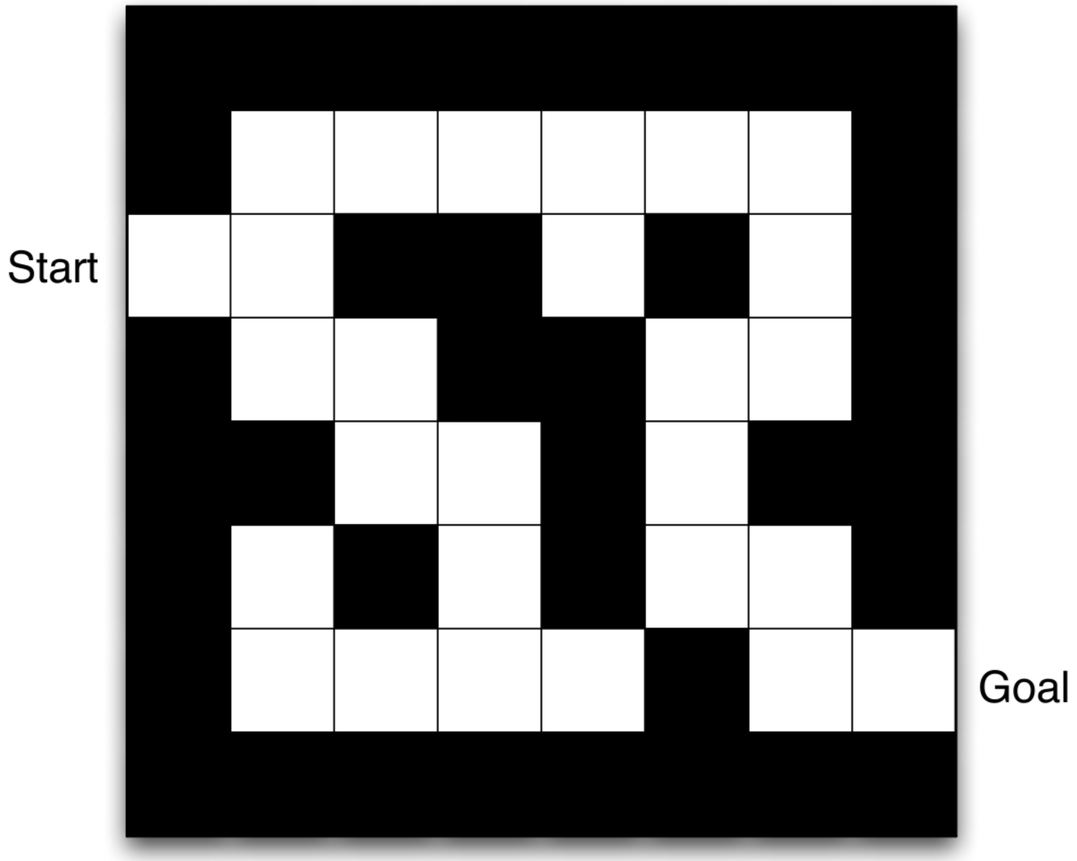
Master programmes in Artificial Intelligence 4 Careers in Europe

An Example

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Maze Example



- Rewards: -1 per time-step
- Actions: N, E, S, W
- States: Agent's location

Tip: When using a negative reward per time step and a reward of 0 at the goal, we define a shortest-path problem.

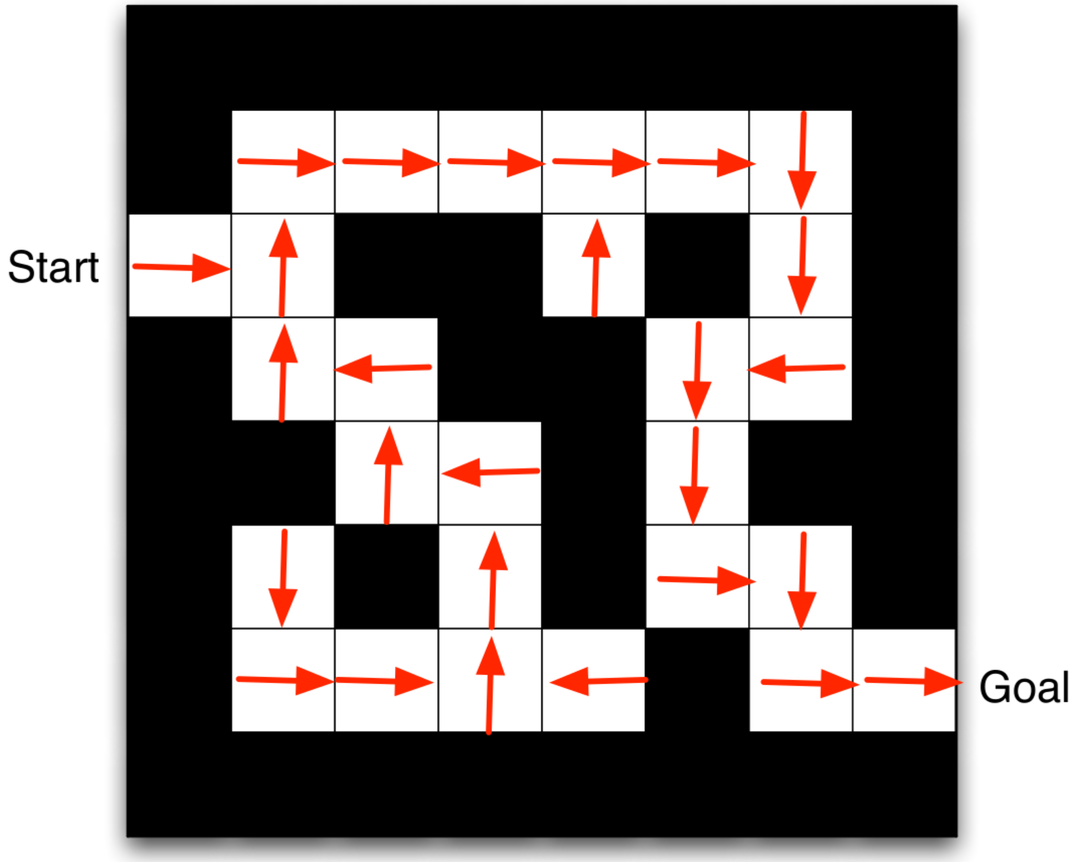
That is, we incentivize the agent to go to the goal as quickly as possible because this would give it the lowest amount of -1s

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Maze Example: Policy



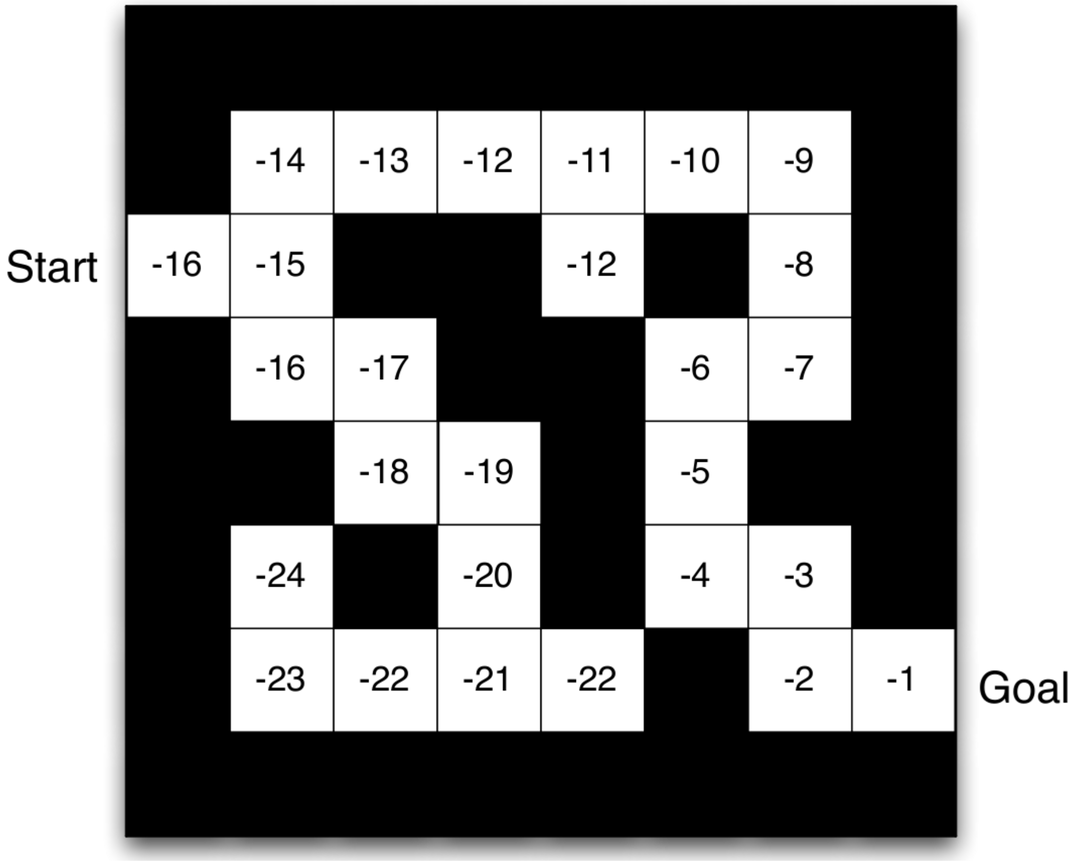
- Arrows represent policy $\pi(s)$ for each state s

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Maze Example: Value Function



- Numbers represent value $v_\pi(s)$ of each state s

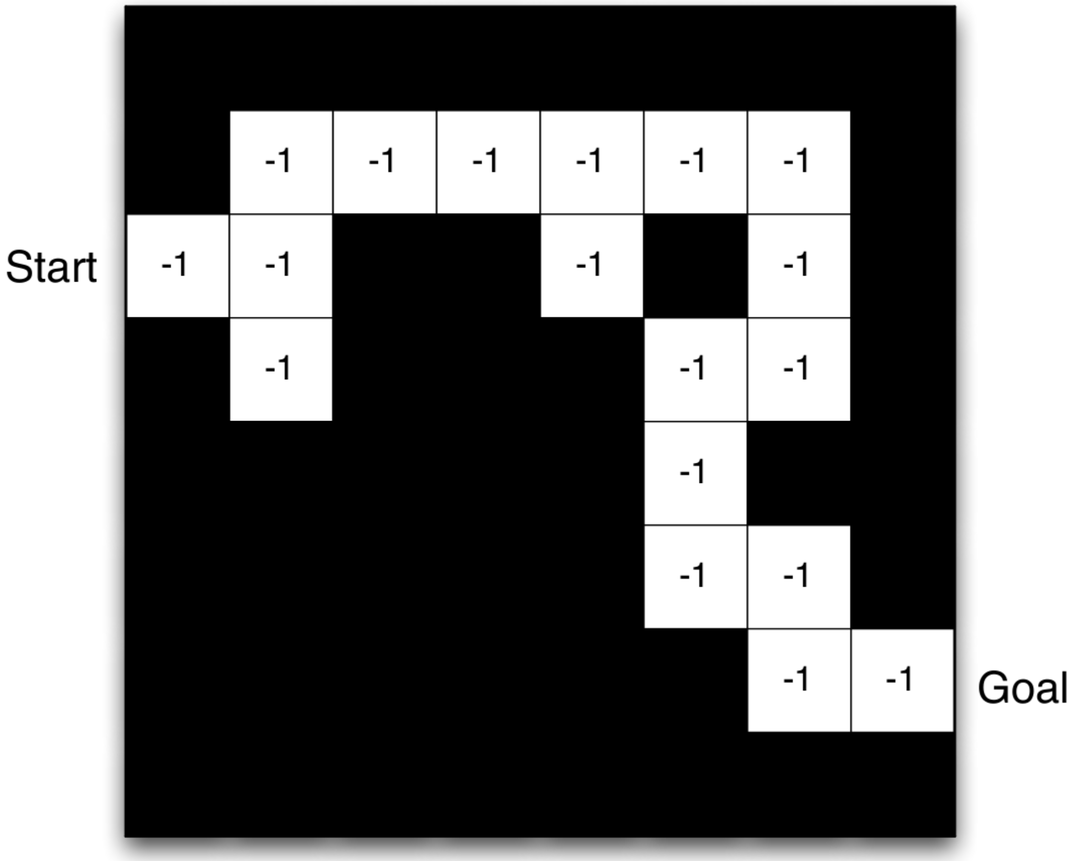
Here since we used a reward of -1 per time-step each value reflects the number of steps to the goal.

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Maze Example: Model



- Grid layout represents partial transition model $\mathcal{P}^{a}_{ss'}$
- Numbers represent immediate reward $\mathcal{R}^{a}_{ss'}$ from each state *s* (same for all *a* and *s'* in this case)

The agent learned this inaccurate model because it might not have explored the bottom part of the maze

The agent can use this model to plan its actions and arrive to the optimal solution for the states it has seen

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Agent Categories

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Agent Categories

- Value Based
  - No Policy (Implicit)
  - Value Function
- Policy Based
  - Policy
  - No Value Function
- Actor Critic
  - Policy
  - Value Function

- Model Free
  - Policy and/or Value Function
  - No Model
- Model Based
  - Optionally Policy and/or Value Function
  - Model

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU
Master programmes in Artificial Intelligence 4 Careers in Europe

Subproblems of the RL problem

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Prediction and Control

- **Prediction**: evaluate the future (for a given policy), e.g., learning a value function
- **Control**: optimise the future (find the best policy)
- These can be strongly related:

$$\pi_*(s) = \underset{\pi}{\operatorname{argmax}}\ v_\pi(s)$$

- If we could predict **everything** do we need anything else?

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Learning and Planning

Two fundamental problems in reinforcement learning

- **Learning**:
  - The environment is initially unknown
  - The agent interacts with the environment
- **Planning**:
  - A model of the environment is given (or learnt)
  - The agent plans in this model (without external interaction)
  - a.k.a. reasoning, pondering, thought, search, planning

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Learning Agent Components

- All components are functions
  - Policies: $\pi : S \rightarrow A$ (or to probabilities over A)
  - Value functions: $v : S \rightarrow R$
  - Models: $m : S \rightarrow S$ and/or $r : S \rightarrow R$
  - State update: $u : S \times O \rightarrow S$
- E.g., we can use neural networks, and use deep learning techniques to learn
- Take care: we do often violate assumptions from supervised learning (iid, stationarity)
- Deep learning is an important tool
- Deep reinforcement learning is a rich and active research field

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

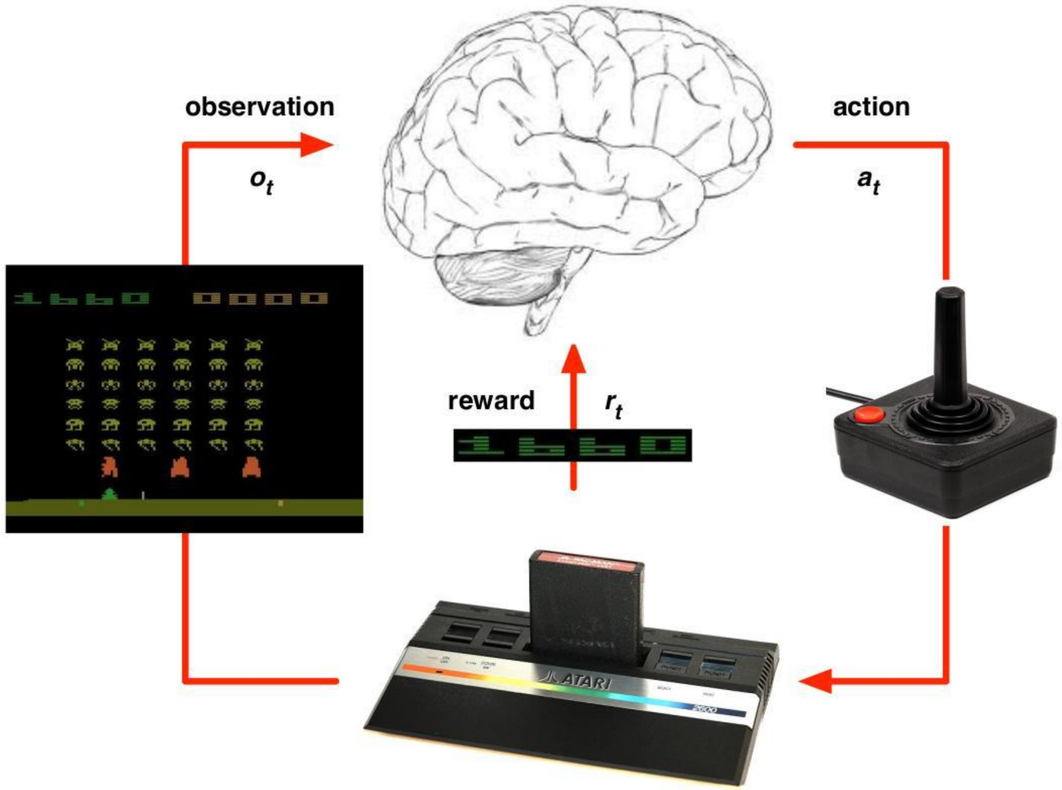
Master programmes in Artificial Intelligence 4 Careers in Europe

Examples

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Atari Games



- Rules of the game are unknown (model-free RL)
- Learn directly from interactive game-play
- Pick actions on joystick, see pixels and scores

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

AlphaGo

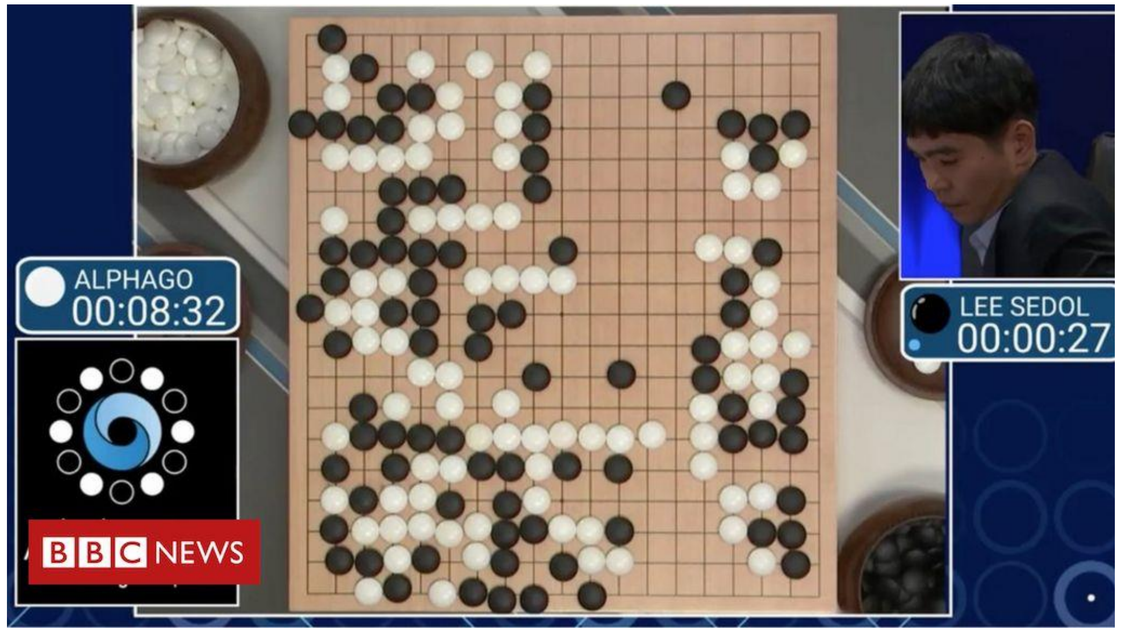


Image Source

- Go: ~ 10^{170} number of board configurations
  - Chess: ~ 10^{44}
  - Number of atoms in the universe: ~ 10^{82}
- Rules of the game are known (model-based RL)
- Combines planning (tree-search) with deep neural networks
- Policy network: selects next move
- Value network: predicts the winner of the game
- Defeated world champions

- See documentary on Netflix or YouTube

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Robotic Quadrupedal Locomotion

YouTube Video

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Learning robust perceptive locomotion for quadrupedal robots in the wild

Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen,
Vladlen Koltun, Marco Hutter

Other RL applications / successes

- TD-Gammon computer backgammon program
- Flying helicopters for aerobatic maneuvers
- Flying stratospheric balloons
- AlphaZero: mastering the games of chess, shogi and go without expert knowledge
- MuZero: master games without knowing their rules
- AlphaStar: Mastering the real-time strategy game StarCraft 2
- Controlling nuclear fusion reactors
- Designing hardware chips
- Discovering more efficient matrix multiplication algorithms
- Solving Rubik's cube with a robot hand
- …

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 17: Markov Decision Processes and Dynamic Programming

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 17: Markov Decision Processes and Dynamic Programming

Learning Outcomes

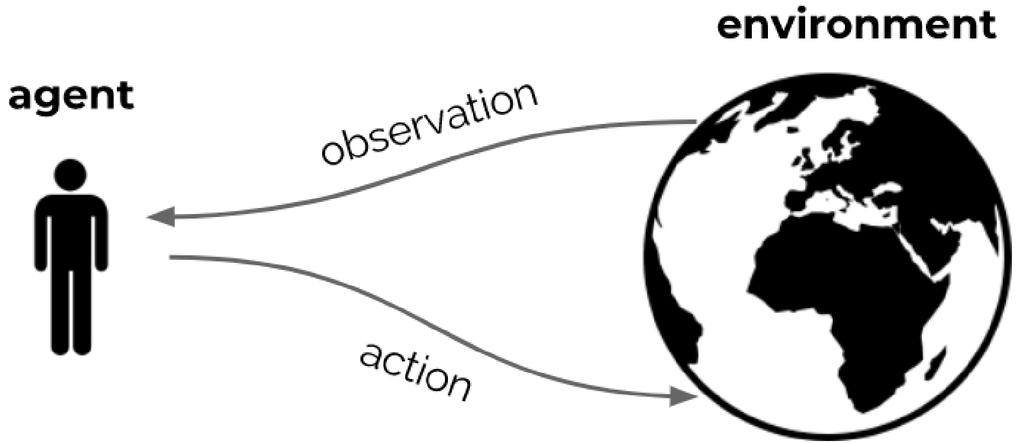
You will understand:

1. How to formalize the RL problem using the framework of Markov Decision Processes (MDPs).
2. What value functions are and how they relate with the objectives in an MDP.
3. Two classes of RL problems: prediction and control.
4. How to use the Bellman Equations to solve RL problems.
5. Dynamic programming and the value iteration and policy iteration algorithms for finding solutions to MDPs.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Recap



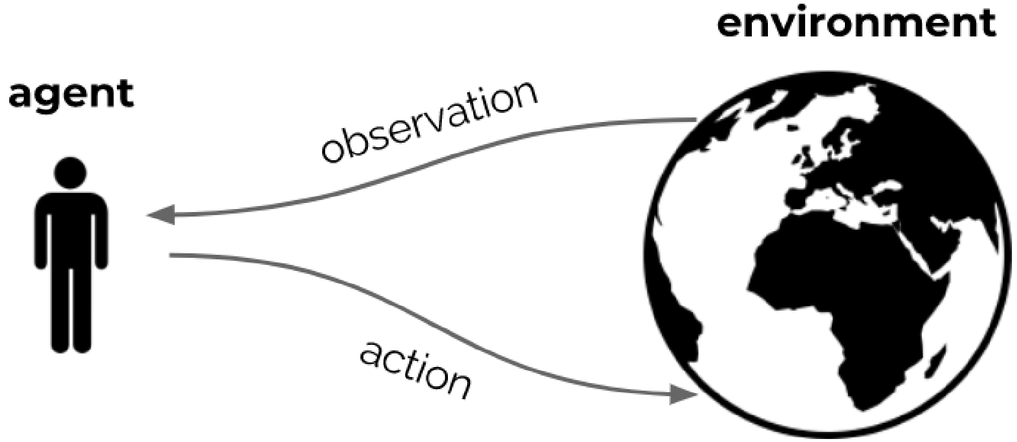
- Reinforcement learning is the science of learning to make decisions
- Agents can learn a **policy**, **value function** and/or a **model**
- The general problem involves taking into account **time** and **consequences**
- Decisions affect the **reward**, the **agent state**, and **environment state**
- Learning is **active**: decisions impact data

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Formalizing the RL interface



- We will discuss a mathematical formulation of the agent-environment interaction
- This is called a **Markov Decision Process (MDP)**
- Enables us to talk clearly about the **objective** and **how to achieve it**

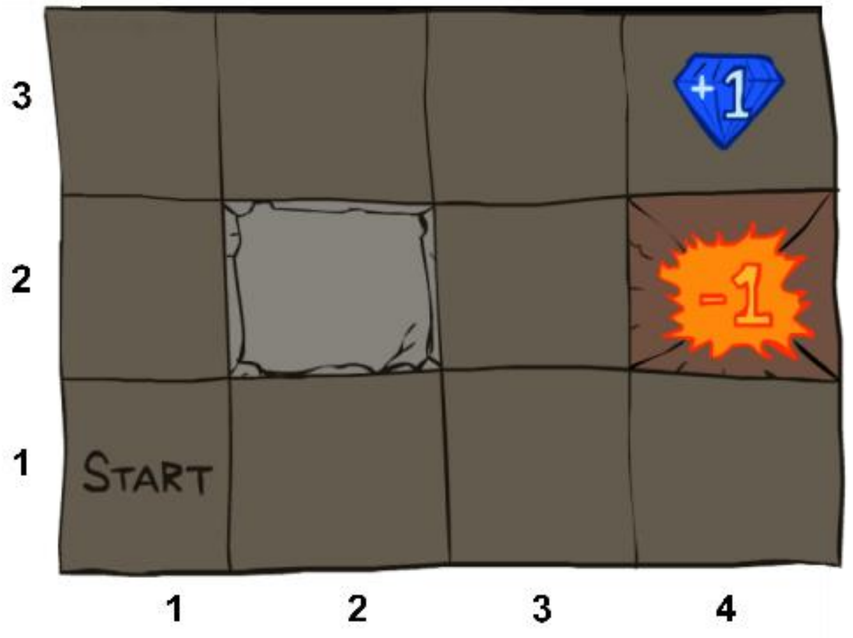
Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: Grid World

- A maze-like problem
  - The agent lives in a grid
  - Walls block the agent's path
- Noisy movement: actions do not always go as planned
  - 80% of the time, the action North takes the agent North (if there is no wall there)
  - 10% of the time, North takes the agent West; 10% East
  - If there is a wall in the direction the agent would have been taken, the agent stays put
- The agent receives rewards each time step
  - Small “living” reward each step (can be negative)
  - Big rewards come at the end (good or bad)
- Goal: maximize sum of rewards



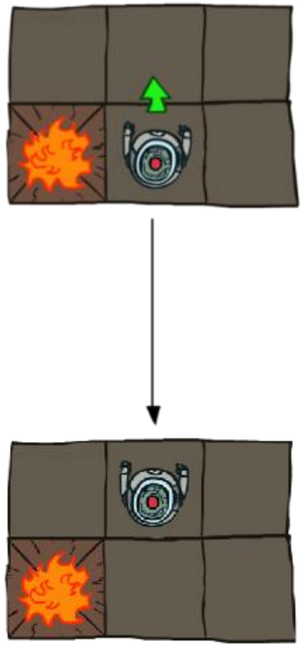
Slide based on : ***UC Berkeley CS188 – Intro to AI course***

Co-financed by the European Union
Connecting Europe Facility

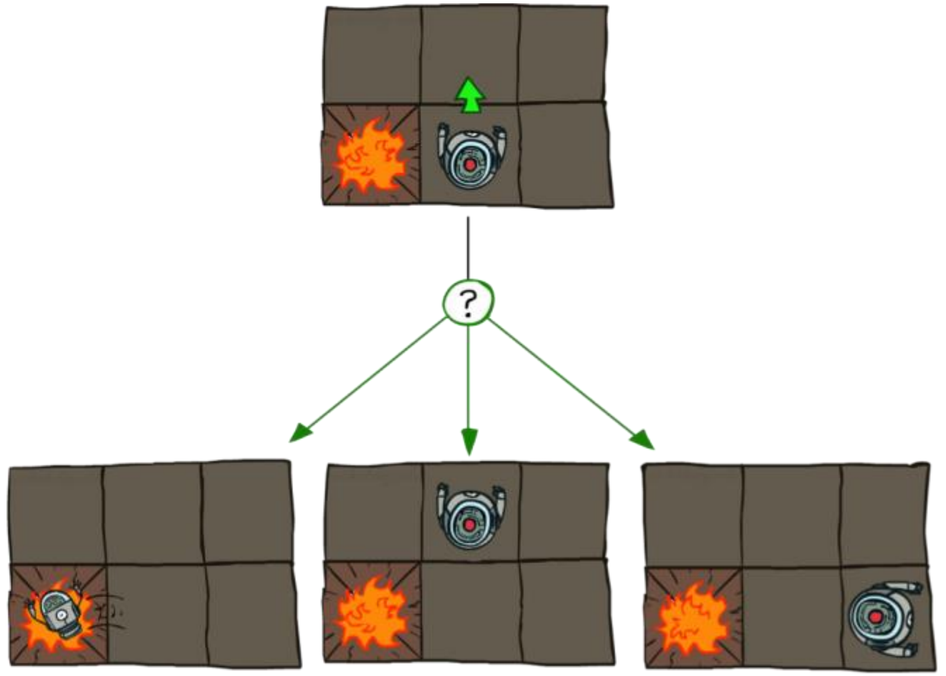
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: Grid World Actions

Deterministic Grid World



Stochastic Grid World



Slide based on : ***UC Berkeley CS188 – Intro to AI course***

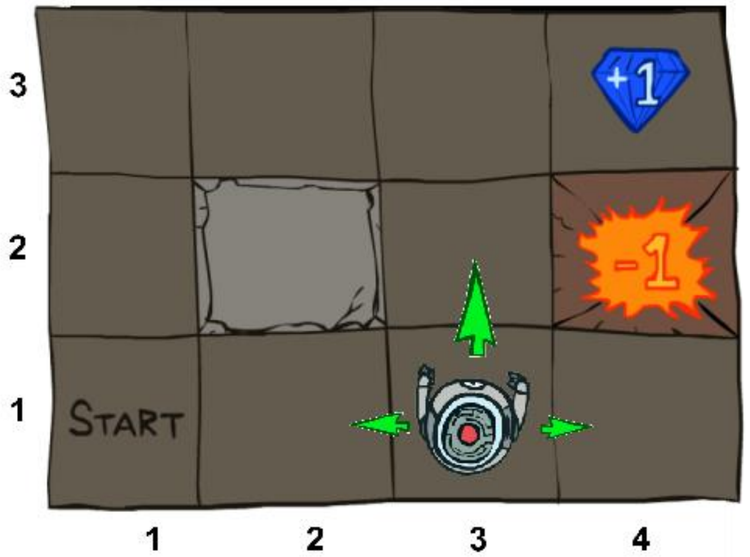
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Markov Decision Process Definition

A **Markov Decision Process** is a tuple (*S, A, T, r*) where:

- *S* is the set of all possible states
- *A* is the set of all possible actions (e.g., motor controls)
- *T(s,a,s')* is a transition function that defines the probability of transitioning to state *s'*, given a state *s* and action *a, i.e., p(s' | s,a)*
  - defines the **dynamics** of the problem
- *r(s,a)* is the reward function (sometimes r(s), r(s') or r(s,a,s'))



Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Markov Property

- "Markov" means that given the present state, the future is independent of the past
- For MDPs, "Markov" means action outcomes depend only on the current state

$$P(S_{t+1} = s' | S_t = s_t, A_t = a_t, S_{t-1} = s_{t-1}, A_{t-1}, \dots S_0 = s_0)$$

$$=$$

$$P(S_{t+1} = s' | S_t = s_t, A_t = a_t)$$

Andrey Markov
(1856-1922)

In an MDP **all states** are assumed to have the Markov property

- The state captures all relevant information from the history.
- Once the state is known, the history may be thrown away.
- The state is a sufficient statistic of the past.

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: cleaning robot

- Consider a robot that cleans soda cans
- Two states: **high** battery charge or **low** battery charge
- Actions: {*wait, search*} in high, {*wait, search, recharge*} in low
- Dynamics may be stochastic
  - p(*St+1* = high | *St* = high, *At* = search) = α
  - p(*St+1* = low | *St* = high, *At* = search) = 1 − α
- Reward could be expected number of collected cans (deterministic), or actual number of collected cans (stochastic)

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: cleaning robot MDP

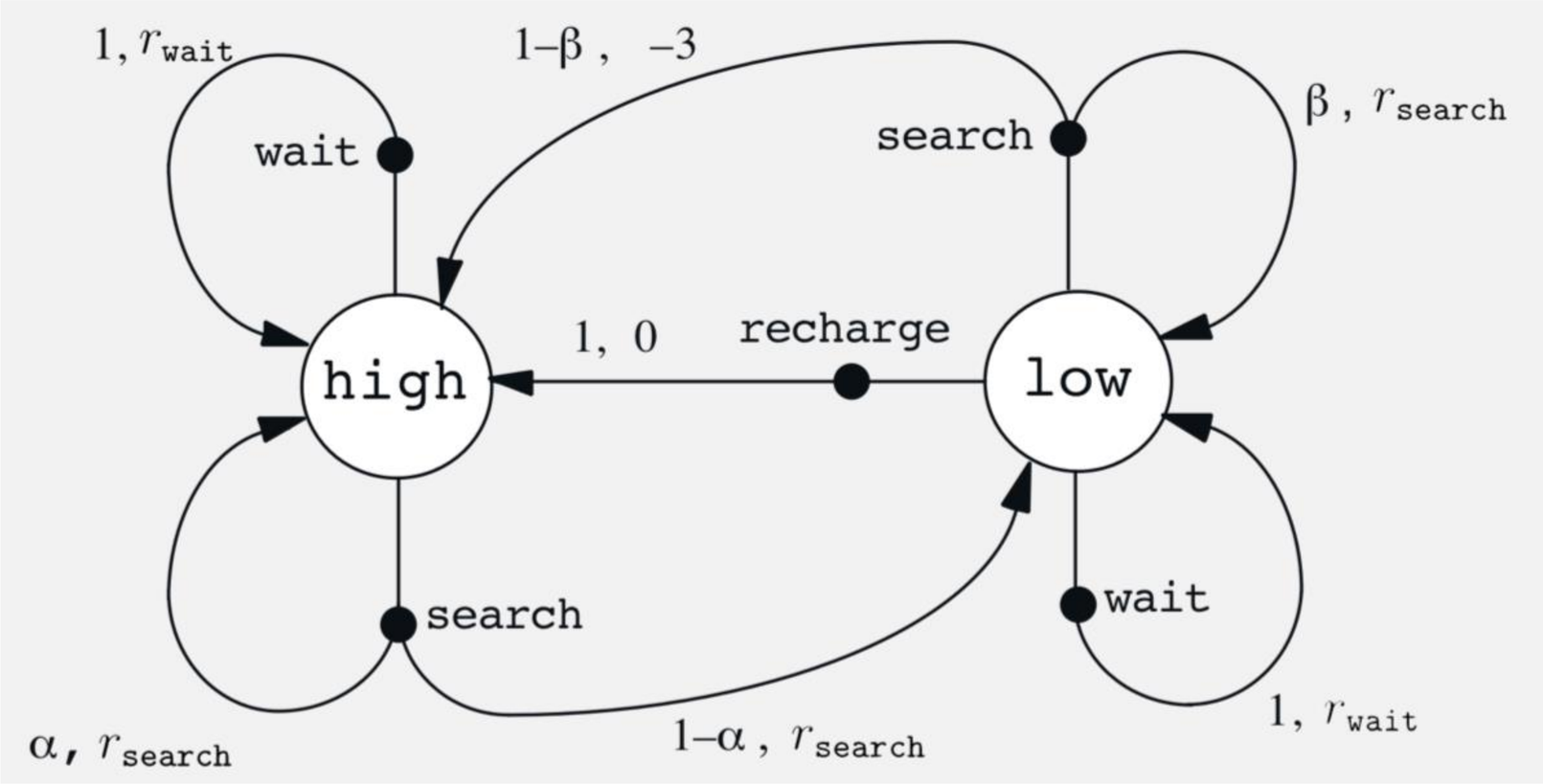
s	a	s'	$p(s' \mid s, a)$	$r(s, a, s')$
high	search	high	α	$r_{\texttt{search}}$
high	search	low	$1-\alpha$	$r_{\texttt{search}}$
low	search	high	$1-\beta$	-3
low	search	low	β	$r_{\texttt{search}}$
high	wait	high	1	$r_{\texttt{wait}}$
high	wait	low	0	$r_{\texttt{wait}}$
low	wait	high	0	$r_{\texttt{wait}}$
low	wait	low	1	$r_{\texttt{wait}}$
low	recharge	high	1	0
low	recharge	low	0	0

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: cleaning robot MDP



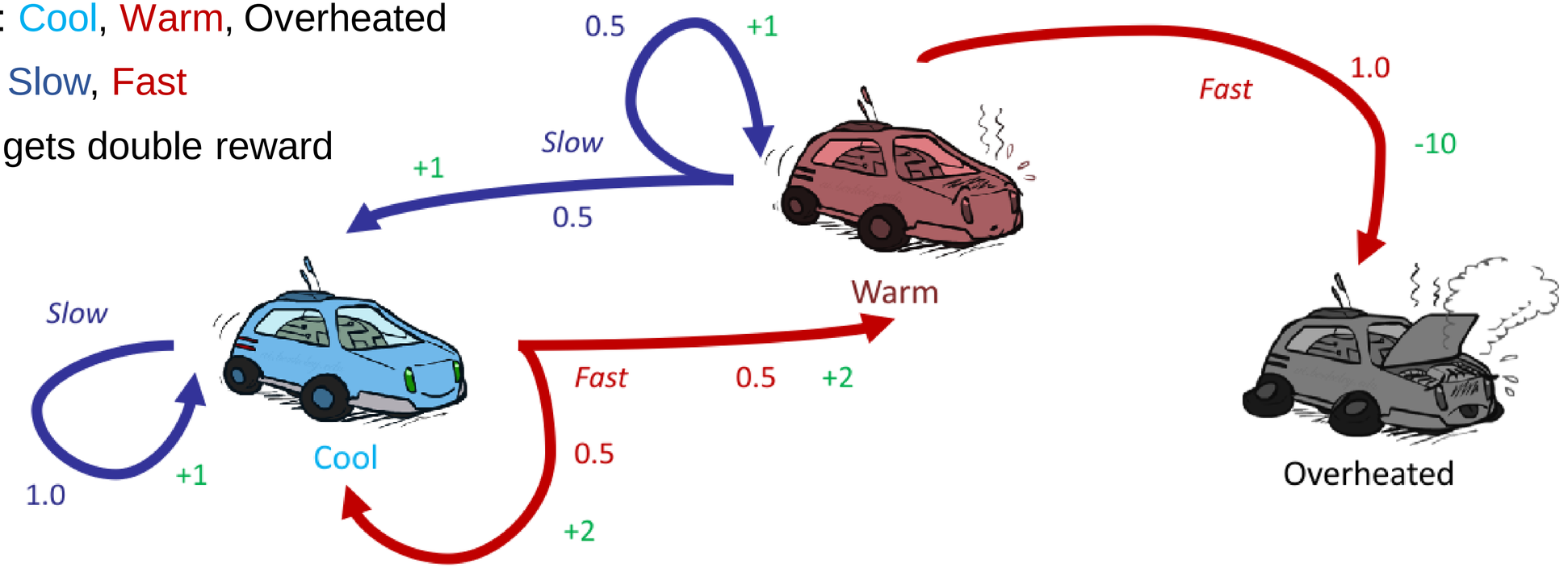
Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: racing

- A robot car wants to travel far, quickly
- Three states: Cool, Warm, Overheated
- Two actions: Slow, Fast
- Going faster gets double reward

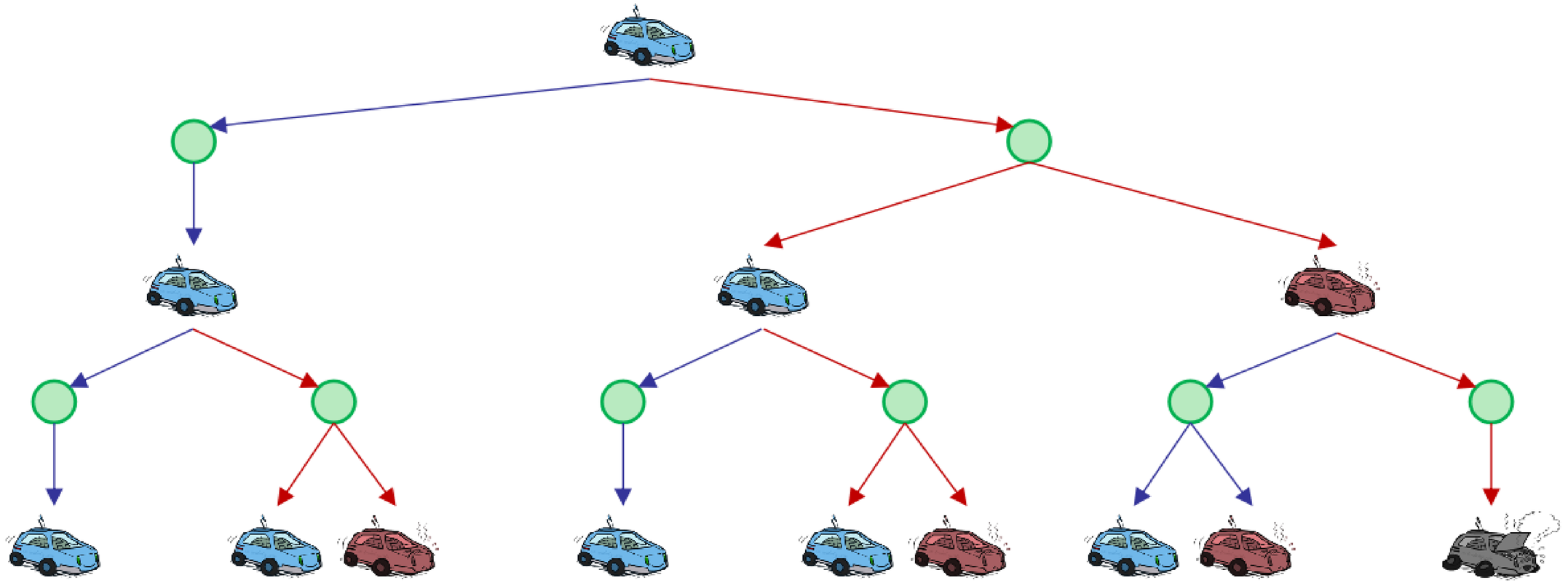


Slide based on : *UC Berkeley CS188 – Intro to AI course*

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: racing search tree



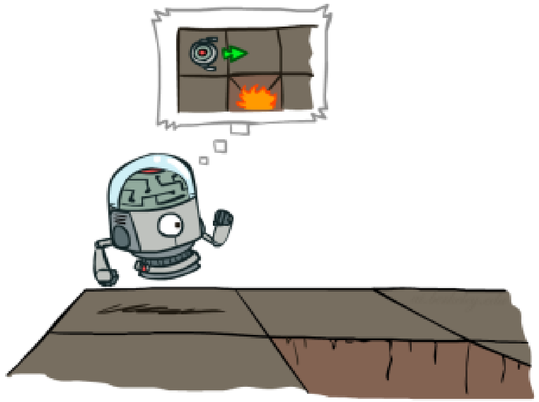
Slide based on : *UC Berkeley CS188 – Intro to AI course*

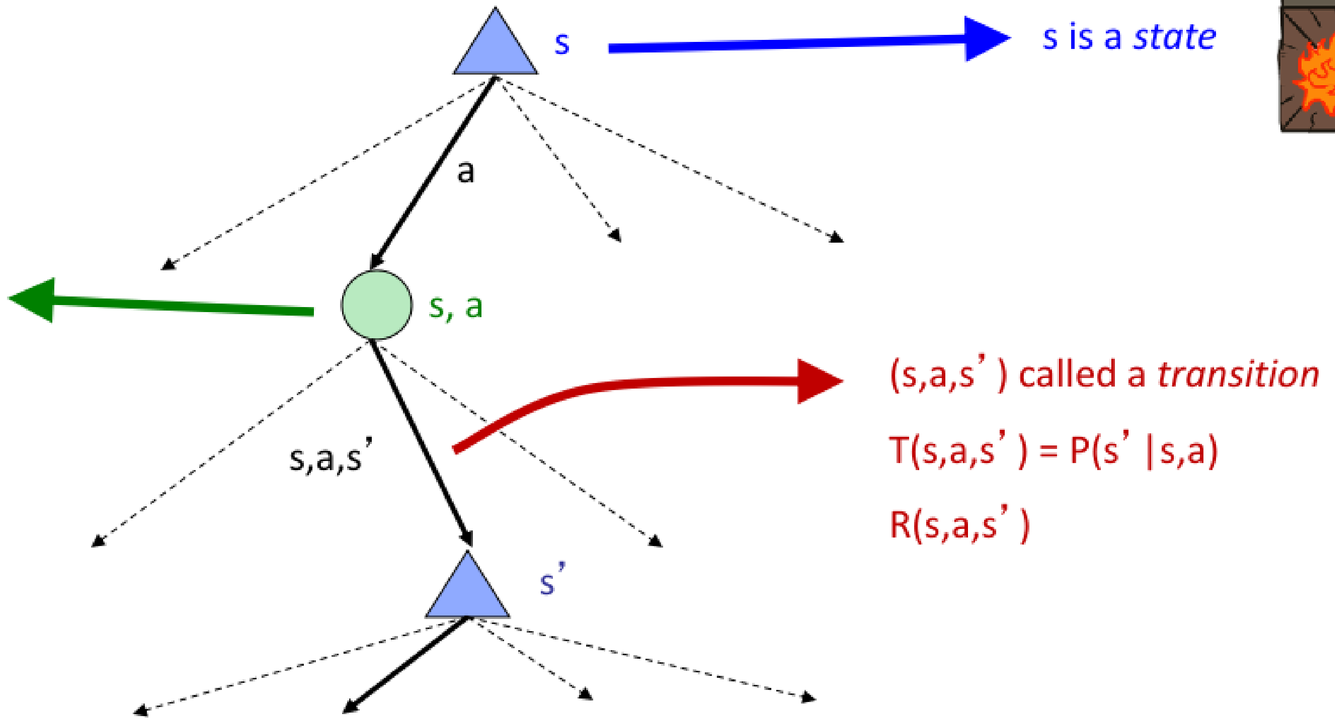
Co-financed by the European Union
Connecting Europe Facility

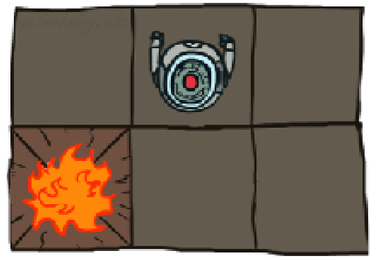
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MDP Search Trees

- Each MDP state projects a search tree







Slide based on : ***UC Berkeley CS188 – Intro to AI course***

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Formalizing the objective

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Returns

- Acting in a MDP results in **immediate rewards** *Rt* , which leads to **returns** *Gt* :
  - Undiscounted return (episodic/finite horizon problem)

$$G_t = R_{t+1} + R_{t+2} + ... + R_T = \sum_{k=0}^{T-t-1} R_{t+k+1}$$

  - Discounted return (finite or infinite horizon problem)

$$G_t = R_{t+1} + \gamma R_{t+2} + ... + \gamma^{T-t} R_T = \sum_{k=0}^{T-t-1} \gamma^k R_{t+k+1}$$

  - Average return (continuing, infinite horizon problem)

$$G_t = \frac{1}{T-t-1}(R_{t+1} + R_{t+2} + ... + R_T) = \frac{1}{T-t-1} \sum_{k=0}^{T-t-1} R_{t+k+1}$$

Note: These are random variables that depend on **MDP** and **policy**

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Discounted Return

- It is reasonable to maximize the sum of rewards
- It is also reasonable to prefer rewards now to rewards later
- One solution: values of rewards decay exponentially
- Discounted **returns** G_t for infinite horizon $T \to \infty$: $$G_t = R_{t+1} + \gamma R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$
- The **discount** $\gamma \in [0, 1]$ is the present value of future rewards
  - The value of receiving reward R after k + 1 time-steps is $\gamma^k R$

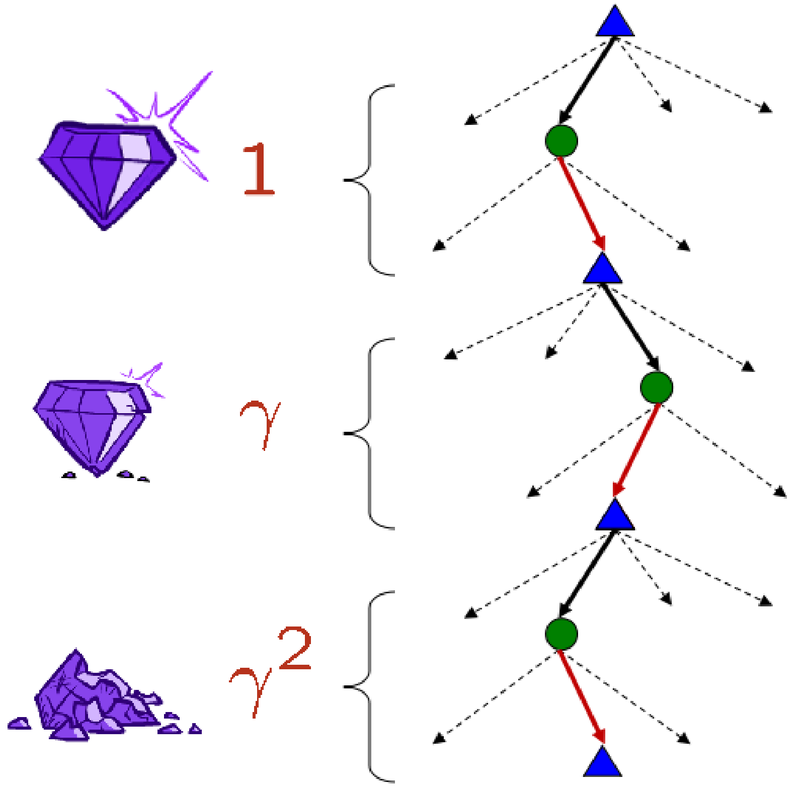
Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Discounting

- How to discount?
  - Each time we descend a level, we multiply in the discount once
- Why discount?
  - Sooner rewards probably have higher value that later rewards
  - γ close to 0 leads to "myopic" evaluation: immediate rewards more important than delayed rewards
  - γ close to 1 leads to "far-sighted" evaluation
  - Mathematically convenient: avoids infinite returns in cyclic MDPs



Slides based on DeepMind's Reinforcement Learning Lectures

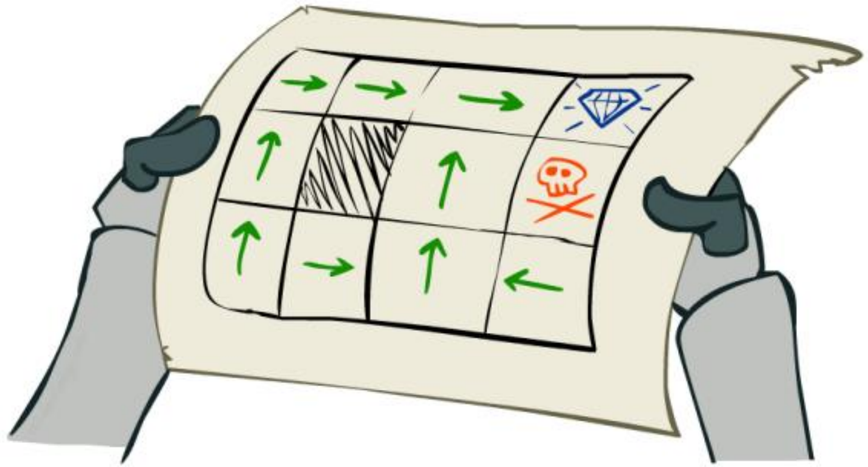
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Policies

Goal of an RL agent:

find a behaviour policy that maximises the expected return G_t



- A **policy** is a mapping $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ that, for every state *s* assigns **for each action a ∈ A the probability of taking that action in state s**. Denoted by $\pi(a|s)$.
- An optimal policy is one that maximises expected return if followed.
- For deterministic policies, we sometimes use the notation $a_t = \pi(s_t)$ to denote the action taken by the policy.

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Value Functions

- The **value function** *v(s)* gives the long-term value (expected return) of state *s*

$$v_\pi(s) = \mathbb{E}\left[G_t \mid S_t = s, \pi\right]$$

- We can define **(state-)action values**:

$$q_\pi(s, a) = \mathbb{E}\left[G_t \mid S_t = s, A_t = a, \pi\right]$$

- Connection between them:

$$v_\pi(s) = \sum_a \pi(a \mid s) q_\pi(s, a) = \mathbb{E}\left[q_\pi(S_t, A_t) \mid S_t = s, \pi\right] \quad , \ \forall s$$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Optimal Value Functions

- The optimal state-value function $v^*(s)$ is the maximum value function over all policies

$$v^*(s) = \max_\pi v_\pi(s)$$

- The optimal action-value function $q^*(s, a)$ is the maximum action-value function over all policies

$$q^*(s, a) = \max_\pi q_\pi(s, a)$$

- The optimal value function specifies the best possible performance in the MDP
- An MDP is “solved” when we know the optimal value function

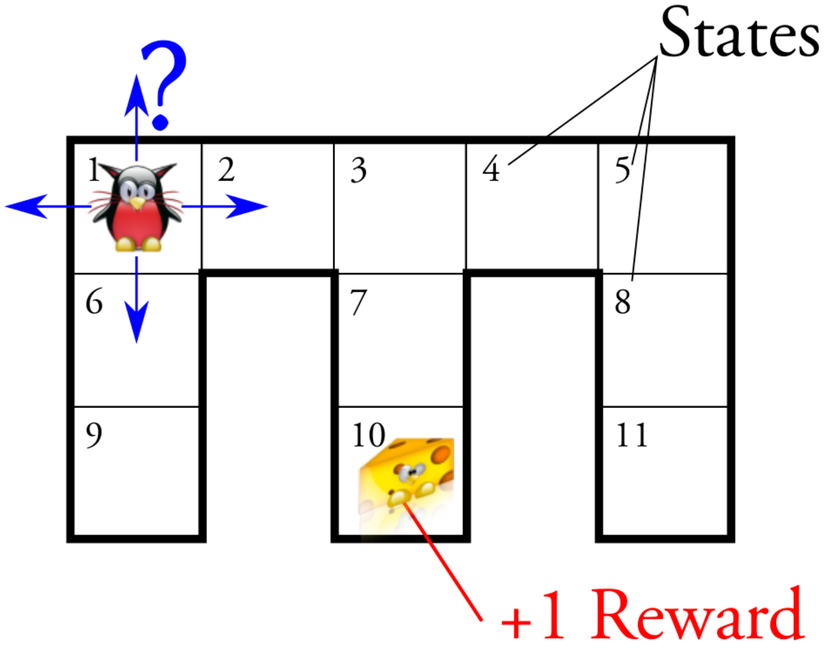
Slides based on DeepMind’s Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Value Functions Intuition Example

- **Goal**: find policy $\pi : S \rightarrow A$ to maximize $\mathbb{E}_{\sim\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$

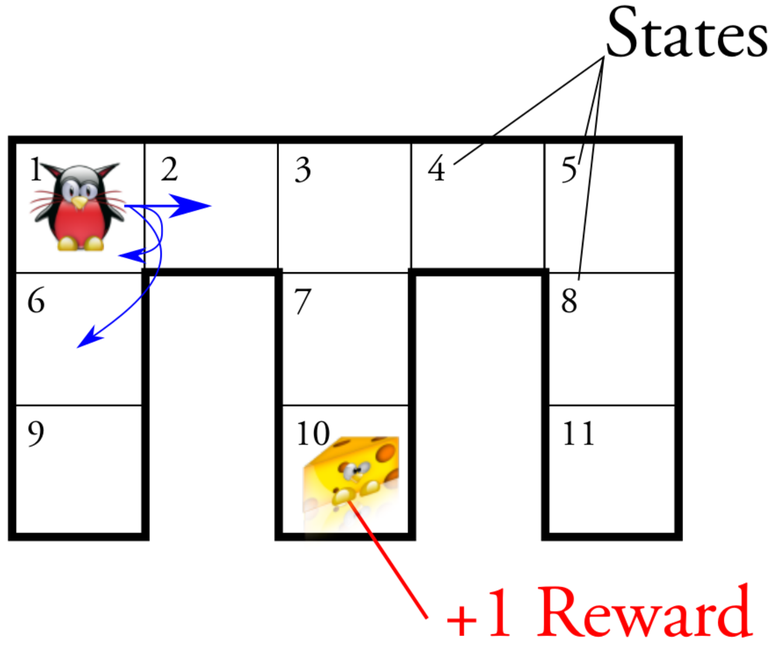


Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Value Functions Intuition Example

- **Goal**: find policy $\pi : S \rightarrow A$ to maximize $\mathbb{E}_{\sim\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$



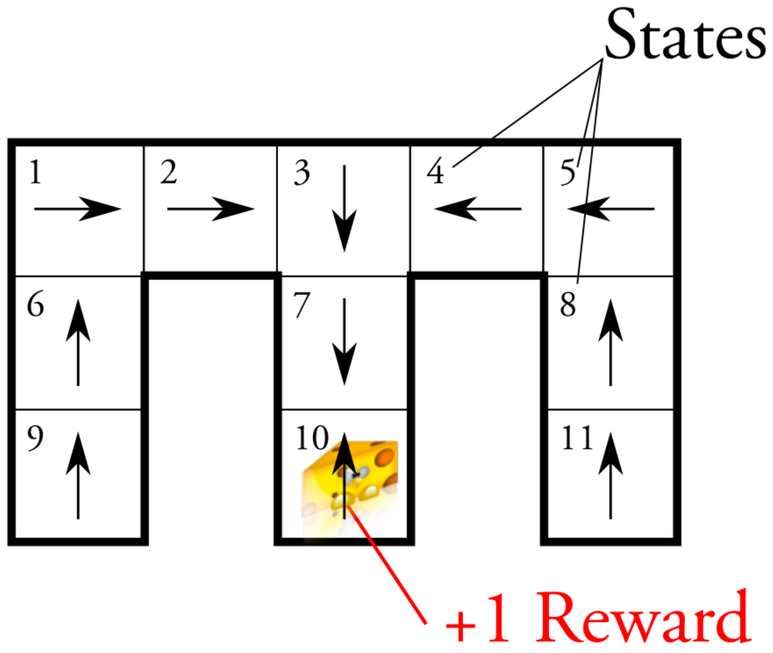
The agent's actions may be stochastic

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Value Functions Intuition Example

- **Goal**: find policy $\pi : S \rightarrow A$ to maximize $\mathbb{E}_{\sim\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$



This is the optimal policy

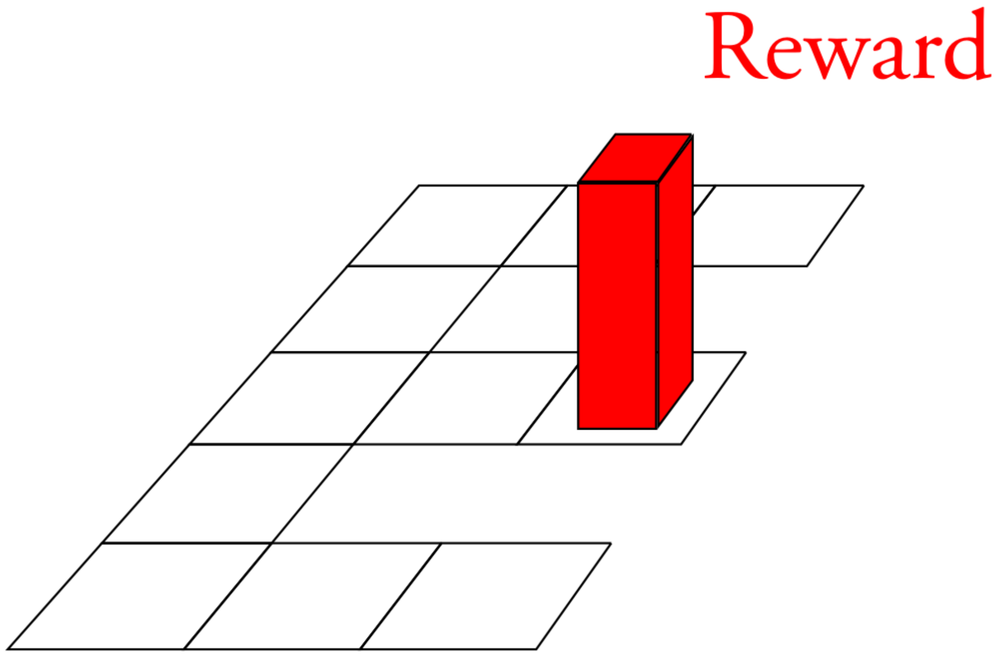
… but how do we find it?

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Value Functions Intuition Example

The optimal value function transforms this "sparse landscape"

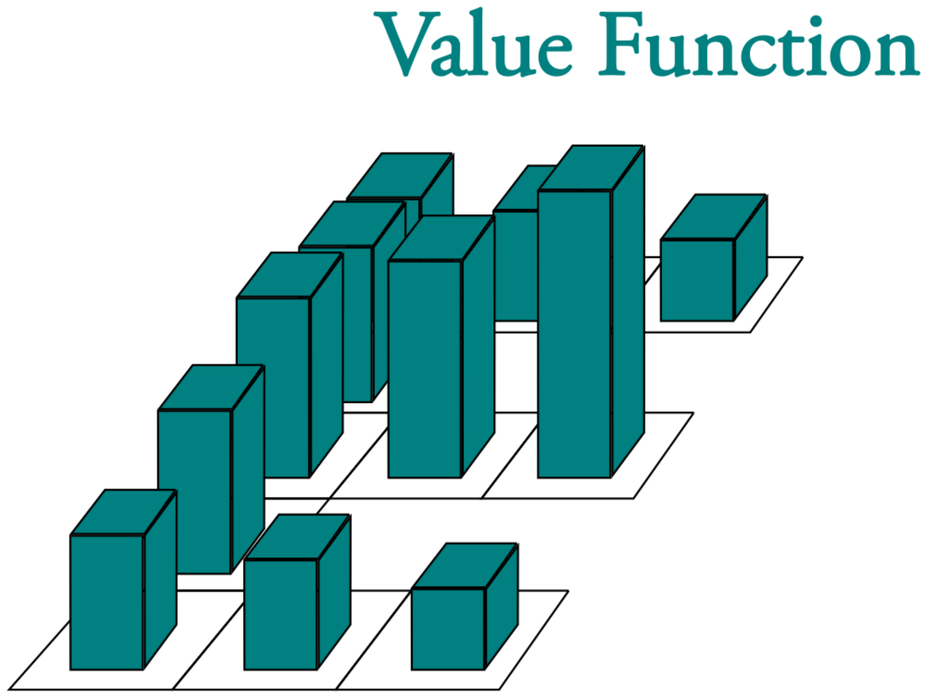


Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Value Functions Intuition Example

… to this "dense" one, which can be used to compute the optimal policy



Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Finding an Optimal Policy

- An optimal policy can be found by maximising over $q_*(s, a)$

$$\pi^*(s, a) = \begin{cases} 1 & \text{if } a = \underset{a \in \mathcal{A}}{\text{argmax}}\ q^*(s, a) \\ 0 & \textit{otherwise} \end{cases}$$

- Observations:
  - There is always a deterministic optimal policy for any MDP
  - If we know $q_*(s, a)$, we immediately have the optimal policy
  - There can be multiple optimal policies
  - If multiple actions maximize $q_*(s, \cdot)$, we can also just pick any of these (including stochastically)

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Bellman Equations

Richard Bellman

(1920 – 1984)

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Bellman Equations

Value Function

- The value function v (s) gives the long-term value of state s: $v_\pi(s) = \mathbb{E}\left[G_t \mid S_t = s, \pi\right]$
- It can be defined recursively:

$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s, \pi\right] \\
&= \mathbb{E}\left[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t \sim \pi(S_t)\right] \\
&= \sum_a \pi(a \mid s) \sum_r \sum_{s'} p(r, s' \mid s, a) \left(r + \gamma v_\pi(s')\right)
\end{aligned}
$$

- The final step writes out the expectation explicitly

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Bellman Equations

Action values

- We can define state-action values: $q_\pi(s,a) = \mathbb{E}\left[G_t \mid S_t = s, A_t = a, \pi\right]$
- This implies

$$\begin{aligned} q_\pi(s,a) &= \mathbb{E}\left[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = a\right] \\ &= \mathbb{E}\left[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a\right] \\ &= \sum_r \sum_{s'} p(r, s' \mid s, a) \left(r + \gamma \sum_{a'} \pi(a' \mid s') q_\pi(s', a') \right) \end{aligned}$$

- Note that:

$$v_\pi(s) = \sum_a \pi(a \mid s) q_\pi(s,a) = \mathbb{E}\left[q_\pi(S_t, A_t) \mid S_t = s, \pi\right] \quad , \forall s$$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Bellman Equations

Bellman Expectation Equations

Given an MDP, for any policy π, the value functions obey the following expectation equations:

$$
\begin{aligned}
v_\pi(s) &= \sum_a \pi(s,a)\left[r(s,a) + \gamma \sum_{s'} p(s'|a,s) v_\pi(s')\right] \\
q_\pi(s,a) &= r(s,a) + \gamma \sum_{s'} p(s'|a,s) \sum_{a' \in \mathcal{A}} \pi(a'|s') q_\pi(s',a')
\end{aligned}
$$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Bellman Equations

Bellman Optimality Equations

Given an MDP, the optimal value functions obey the following expectation equations:

$$v^*(s) = \max_a \left[r(s,a) + \gamma \sum_{s'} p(s'|a,s) v^*(s') \right]$$

$$q^*(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|a,s) \max_{a' \in \mathcal{A}} q^*(s',a')$$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Solving RL problems using the Bellman Equations

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Problems in RL

Two problems:

1. Estimating v_π or q_π is called policy evaluation or, simply, prediction
   - Given a policy, what is my expected return under that behaviour?
   - Given this treatment protocol/trading strategy, what is my expected return?

2. Estimating v_* or q_* is sometimes called control, because these can be used for policy optimisation
   - What is the optimal way of behaving? What is the optimal value function?
   - What is the optimal treatment? What is the optimal control policy to minimise time, fuel consumption, etc?

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

A solution

Bellman Equation in Matrix Form

- The Bellman equation, for given policy π, can be expressed using matrices,

$$\mathbf{v} = \mathbf{r}^{\pi} + \gamma \mathbf{P}^{\pi} \mathbf{v}$$

where

$$v_i = v(s_i)$$
$$r_i^{\pi} = \mathbb{E}\left[R_{t+1} \mid S_t = s_i, A_t \sim \pi(S_t)\right]$$
$$P_{ij}^{\pi} = p(s_j \mid s_i) = \sum_a \pi(a \mid s_i) p(s_j \mid s_i, a)$$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

A solution

Bellman Equation in Matrix Form

- The Bellman equation, for given policy π, can be expressed using matrices,

$$\mathbf{v} = \mathbf{r}^{\pi} + \gamma \mathbf{P}^{\pi} \mathbf{v}$$

- This is a linear equation that can be solved directly:

$$\begin{aligned} \mathbf{v} &= \mathbf{r}^{\pi} + \gamma \mathbf{P}^{\pi} \mathbf{v} \\ \left(\mathbf{I} - \gamma \mathbf{P}^{\pi}\right) \mathbf{v} &= \mathbf{r}^{\pi} \\ \mathbf{v} &= \left(\mathbf{I} - \gamma \mathbf{P}^{\pi}\right)^{-1} \mathbf{r}^{\pi} \end{aligned}$$

- Computational complexity is $O(|S|^3)$ - only possible for small problems
- There are **iterative methods** for larger problems

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

A solution

Solving the Bellman Optimality Equation

- The Bellman optimality equation is non-linear (because of the max operator)
- Cannot use the same direct matrix solution as for policy evaluation

- Many iterative solution methods:
  - Using models / **dynamic programming**
    - Value iteration
    - Policy iteration
  - Using samples
    - Monte Carlo
    - Q-learning
    - SARSA

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Dynamic Programming

Dynamic programming refers to a collection of algorithms that can be used to compute optimal policies given a perfect model of the environment as a Markov decision process (MDP).

Sutton & Barto 2018

- Dynamic programming:
  - decomposes a problem into smaller subproblems in a recursive manner
  - consist of two important parts:

policy evaluation and **policy improvement**

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Policy Evaluation

- We start by discussing how to estimate $v_\pi(s) = \mathbb{E}\left[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid s, \pi\right]$

- Idea: turn this equality into an update

Algorithm

- Initialize v_0, e.g., to zero
- Iterate:

$$\forall s: \quad v_{k+1}(s) \leftarrow \mathbb{E}\left[R_{t+1} + \gamma v_k(S_{t+1}) \mid s, \pi\right]$$

- Stopping: whenever $v_{k+1}(s) = v_k(s)$, for all s, we must have found v_π

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: Policy Evaluation

actions

Goal	1	2	3
4	5	6	7
8	9	10	11
12	13	14	Goal

$$R_t = -1$$
on all transitions

π = random policy

γ = 1.0

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: Policy Evaluation

$v_0 = 0$

$v_{k+1}(s_t) = E\,[\, R_{t+1} + \gamma\, v_k(s_{t+1}) \,]$

$= E\,[\, -1 + v_k(s_{t+1}) \,]$

$= -1 + E\,[\, v_k(s_{t+1}) \,]$

$k = 0$

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

$v_1(s_t) = -1 + E\,[\, v_0(s_{t+1}) \,] = -1$

$k = 1$

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

$v_2(s_t) = -1 + E\,[\, v_1(s_{t+1}) \,]$

$= -1 + (-1-1-1+0)/4 = -1.75$

$k = 2$

0.0	-1.7	-2.0	-2.0
-1.7	-2.0	-2.0	-2.0
-2.0	-2.0	-2.0	-1.7
-2.0	-2.0	-1.7	0.0

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: Policy Evaluation

$k = 3$

0.0	-2.4	-2.9	-3.0
-2.4	-2.9	-3.0	-2.9
-2.9	-3.0	-2.9	-2.4
-3.0	-2.9	-2.4	0.0

$k = 10$

0.0	-6.1	-8.4	-9.0
-6.1	-7.7	-8.4	-8.4
-8.4	-8.4	-7.7	-6.1
-9.0	-8.4	-6.1	0.0

Value function of the random policy

$k = \infty$

0.0	-14.	-20.	-22.
-14.	-18.	-20.	-20.
-20.	-20.	-18.	-14.
-22.	-20.	-14.	0.0

Policy evaluation may require many steps

$v_{k+1}(s_t) = -1 + E\,[\, v_k(s_{t+1}) \,]$

$v_{k+1}(s_t) = v_k(s_{t+1})$

$v_k = v_\pi$

When the value function does not change, we have finished policy evaluation

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Policy Improvement

- We can use evaluation to then improve our policy
- Just being greedy with respect to the values of the random policy may suffice (not true in general)

Algorithm

- Iterate using:

$$\forall s: \quad \pi_{\text{new}}(s) = \underset{a}{\text{argmax}}\, q_\pi(s,a)$$
$$= \underset{a}{\text{argmax}}\, \mathbb{E}\left[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = a\right]$$

- Evaluate π_{new} and repeat

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

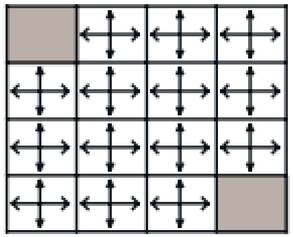
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: Policy evaluation + Greedy Improvement

We can take some value function and extract some policy using greedy improvement

$k = 0$

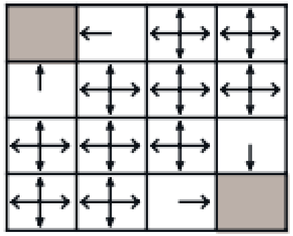
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0



← random policy

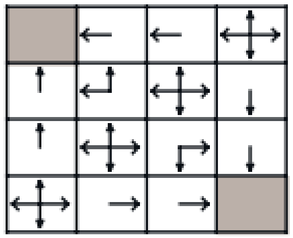
$k = 1$

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0



$k = 2$

0.0	-1.7	-2.0	-2.0
-1.7	-2.0	-2.0	-2.0
-2.0	-2.0	-2.0	-1.7
-2.0	-2.0	-1.7	0.0



Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example: Policy evaluation + Greedy Improvement

We can take some value function and extract some policy using greedy improvement

$k = 3$

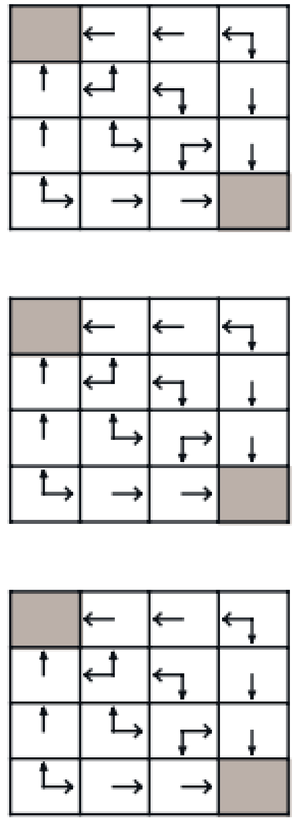
0.0	-2.4	-2.9	-3.0
-2.4	-2.9	-3.0	-2.9
-2.9	-3.0	-2.9	-2.4
-3.0	-2.9	-2.4	0.0

$k = 10$

0.0	-6.1	-8.4	-9.0
-6.1	-7.7	-8.4	-8.4
-8.4	-8.4	-7.7	-6.1
-9.0	-8.4	-6.1	0.0

$k = \infty$

0.0	-14.	-20.	-22.
-14.	-18.	-20.	-20.
-20.	-20.	-18.	-14.
-22.	-20.	-14.	0.0



optimal policy

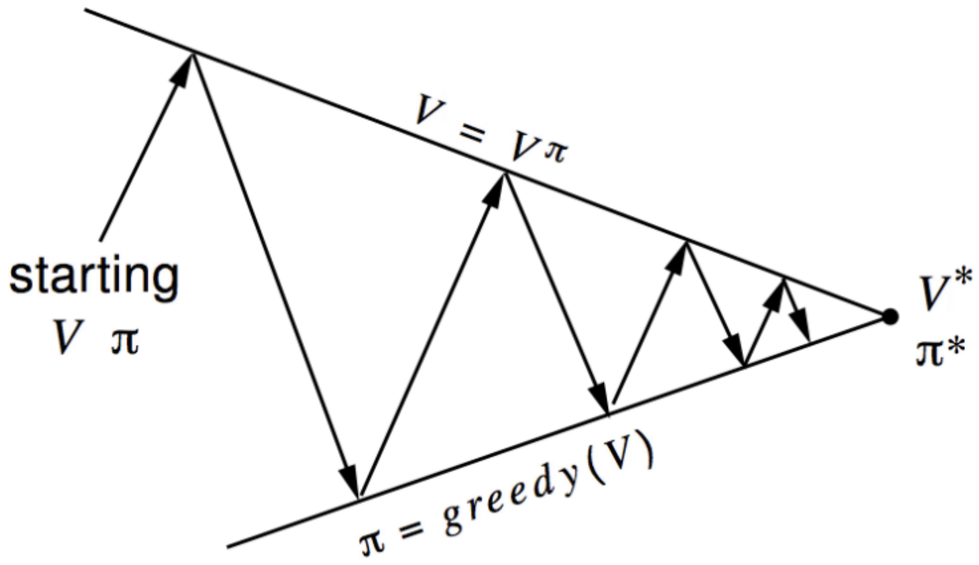
Notice that value function still changing but the optimal policy has been learned from k=3

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

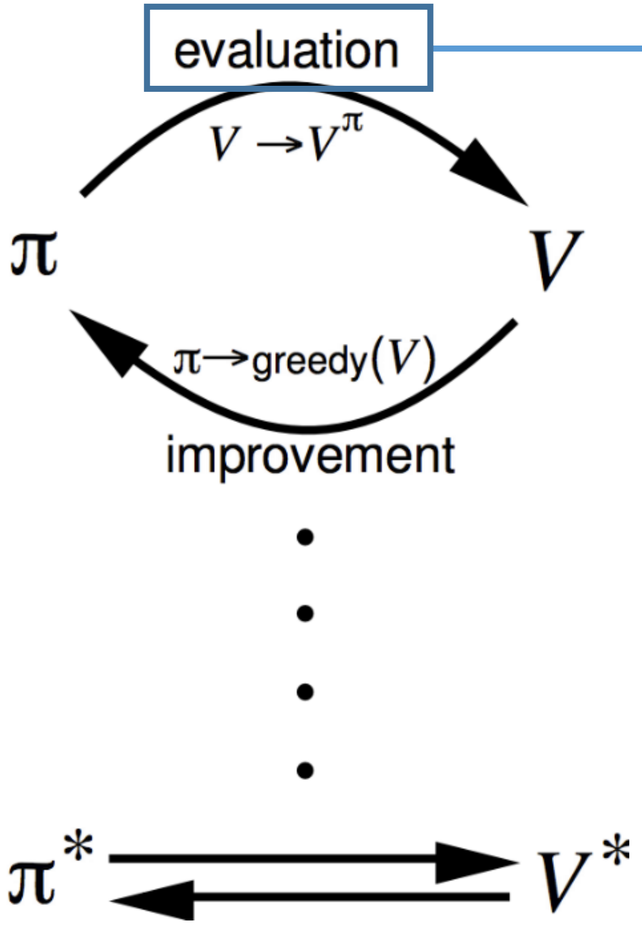
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Policy Iteration



Policy evaluation Estimate v^π

Policy improvement Generate $\pi' \geq \pi$



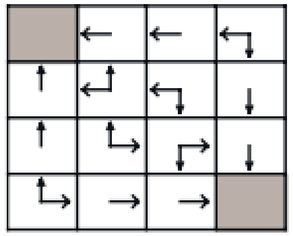
Iterative, i.e., may require a lot of computation

Policy Evaluation

$k = \infty$

0.0	-14.	-20.	-22.
-14.	-18.	-20.	-20.
-20.	-20.	-18.	-14.
-22.	-20.	-14.	0.0

Policy Improvement



Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Policy Iteration

- Does policy evaluation need to converge to v_{π} ?
- Or should we stop when we are ‘close’ ? (E.g., with a threshold on the change to the values)
  - Or simply stop after k iterations of iterative policy evaluation?
  - In the small gridworld k = 3 was sufficient to achieve optimal policy

- Extreme: Why not update policy every iteration — i.e. stop after k = 1?
  - This is equivalent to **value iteration**

Slides based on DeepMind’s Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Value Iteration

- We could take the Bellman optimality equation, and turn that into an update

$$\forall s: \quad v_{k+1}(s) \leftarrow \max_a \mathbb{E}\left[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s, A_t = s\right]$$

- This is equivalent to policy iteration, with k = 1 step of policy evaluation between each two (greedy) policy improvement steps

Algorithm: Value Iteration

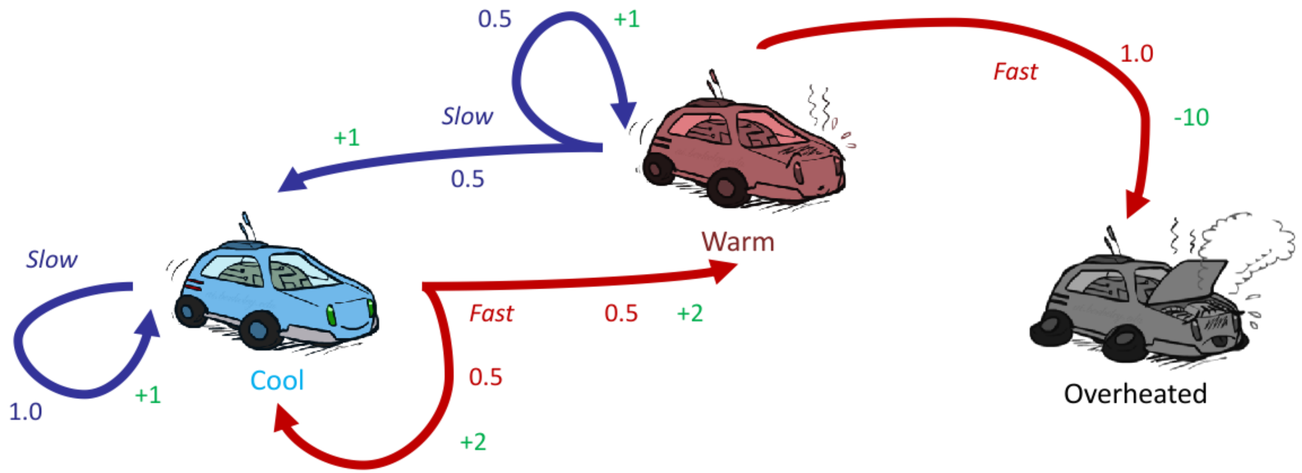
- Initialise v_0
- Update: $v_{k+1}(s) \leftarrow \max_a \mathbb{E}\left[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s, A_t = s\right]$
- Stopping: whenever $v_{k+1}(s) = v_k(s)$, for all s, we must have found v^*

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Value Iteration racing example



Algorithm

1. $\forall s \in S$, initialize $V_0(s) = 0$

2. Repeat until convergence:

$$\forall s \in S,\ V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_k(s')]$$

Convergence:

$$\forall s, V_{k+1}(s) = V_k(s)$$

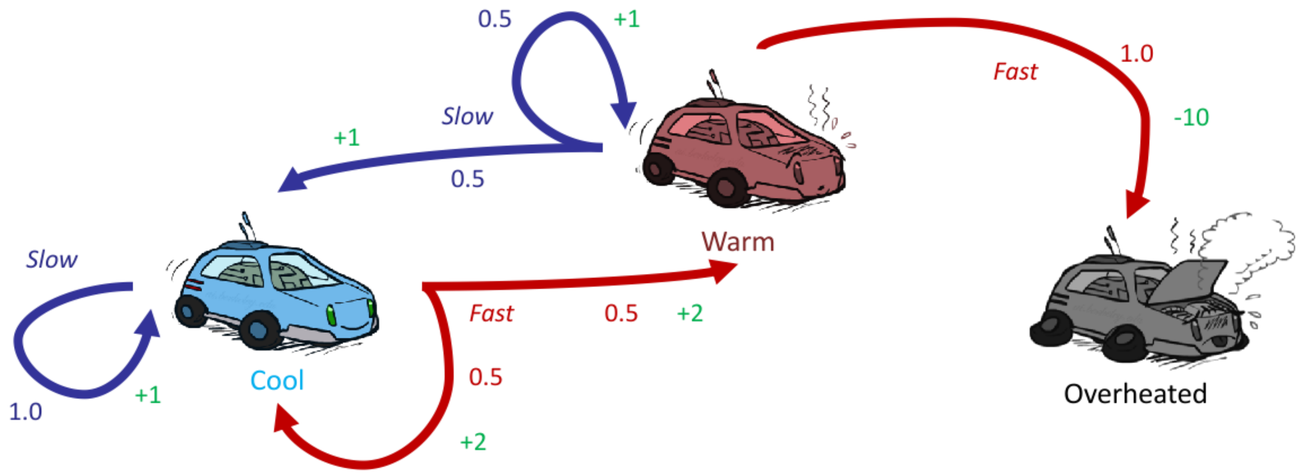
Slide based on : ***UC Berkeley CS188 – Intro to AI course***

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Value Iteration racing example

$$\forall s \in S,\ V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_k(s')]$$



	cool	warm	overheated
V_0	0	0	0

$$\begin{aligned} V_1(cool) &= \max\{1\cdot[1+0.5\cdot 0],\ 0.5\cdot[2+0.5\cdot 0]+0.5\cdot[2+0.5\cdot 0]\} \\ &= \max\{1,2\} \\ &= \boxed{2} \end{aligned}$$

$$\begin{aligned} V_1(warm) &= \max\{0.5\cdot[1+0.5\cdot 0]+0.5\cdot[1+0.5\cdot 0],\ 1\cdot[-10+0.5\cdot 0]\} \\ &= \max\{1,-10\} \\ &= \boxed{1} \end{aligned}$$

$$\begin{aligned} V_1(overheated) &= \max\{\} \\ &= \boxed{0} \end{aligned}$$

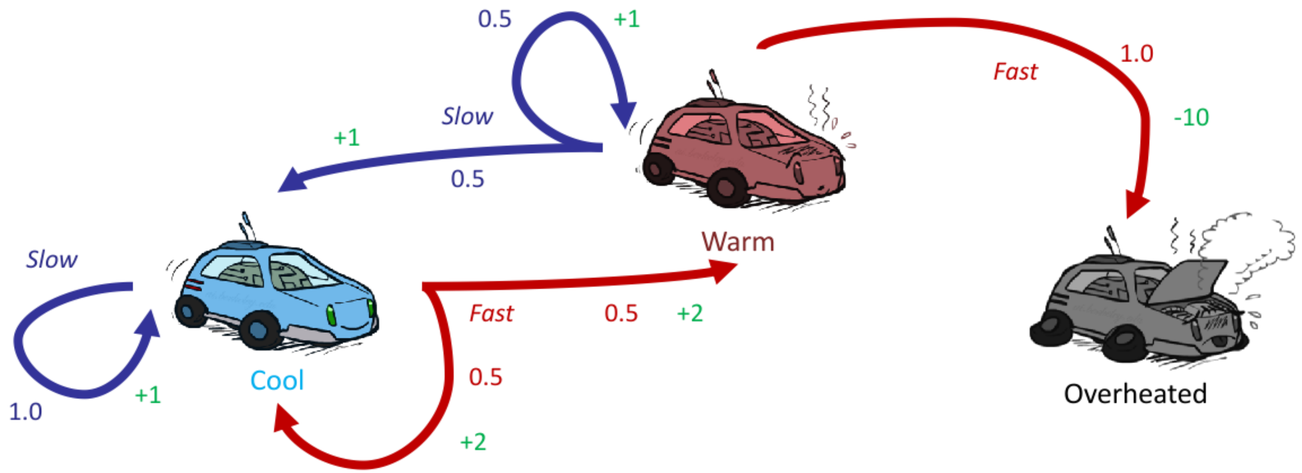
Slide based on : ***UC Berkeley CS188 – Intro to AI course***

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Value Iteration racing example

$$\forall s \in S,\ V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_k(s')]$$



	cool	warm	overheated
V_0	0	0	0
V_1	2	1	0

$$\begin{aligned} V_2(cool) &= \max\{1\cdot[1+0.5\cdot 2],\ 0.5\cdot[2+0.5\cdot 2]+0.5\cdot[2+0.5\cdot 1]\} \\ &= \max\{2, 2.75\} \\ &= \boxed{2.75} \end{aligned}$$

$$\begin{aligned} V_2(warm) &= \max\{0.5\cdot[1+0.5\cdot 2]+0.5\cdot[1+0.5\cdot 1],\ 1\cdot[-10+0.5\cdot 0]\} \\ &= \max\{1.75, -10\} \\ &= \boxed{1.75} \end{aligned}$$

$$\begin{aligned} V_2(overheated) &= \max\{\} \\ &= \boxed{0} \end{aligned}$$

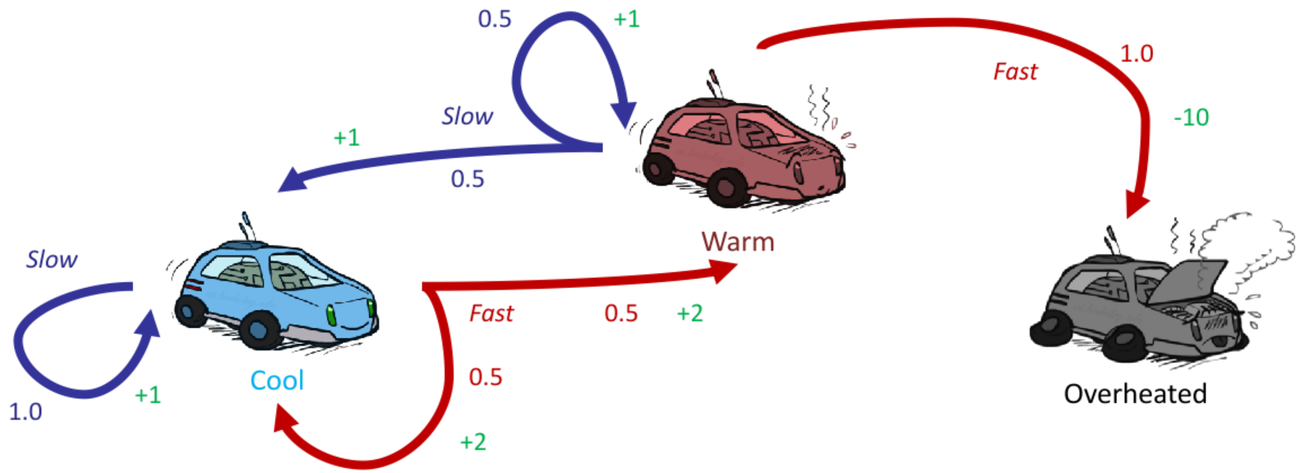
Slide based on : ***UC Berkeley CS188 – Intro to AI course***

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Value Iteration racing example

$$\forall s \in S,\ V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_k(s')]$$



	cool	warm	overheated
V_0	0	0	0
V_1	2	1	0
V_2	2.75	1.75	0

Need more iterations to converge to the optimal value function

Slide based on : *UC Berkeley CS188 – Intro to AI course*

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Policy iteration vs Value iteration

- Policy iteration:
  - Starts with random policy
  - Runs policy evaluation until convergence, and then a policy improvement step
  - Requires few iterations to converge, but at the expense of more computation time due to policy evaluation

- Value iteration:
  - Starts with a random (or zero) value function
  - Runs one step of policy evaluation followed by policy improvement
  - Requires more iterations to converge to the optimal value function

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

What have we covered

- Markov Decision Processes
- Objectives in an MDP : different notion of return
- Value functions – expected returns, condition on state (and action)
- Optimality principles in MDPs: optimal value functions and optimal policies
- Bellman Equations
- Two class of problems in RL: evaluation and control
- How to compute v_{π} (aka solve an evaluation/prediction problem)
- How to compute the optimal value function via dynamic programming
  - Policy Iteration
  - Value Iteration

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Next Lectures

- Model-free Reinforcement Learning
- Revision

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 18: Model-free Reinforcement Learning

Vassilis Vassiliades, PhD

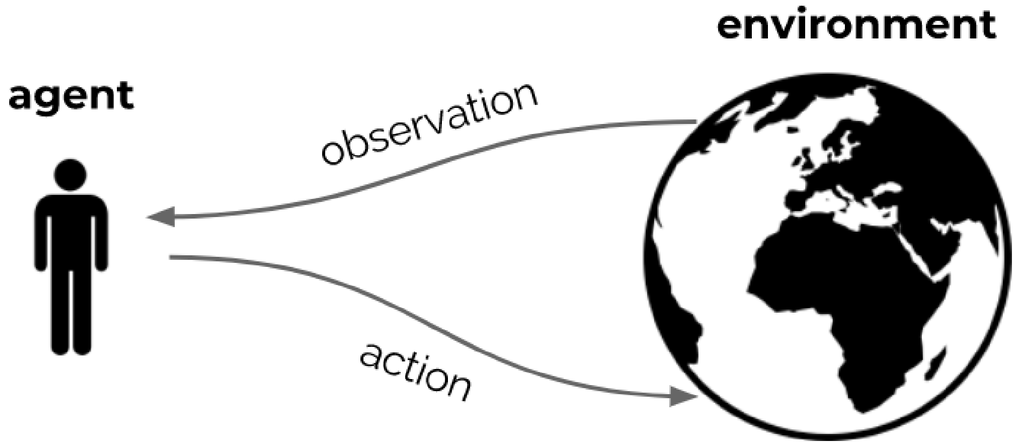
Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Recap

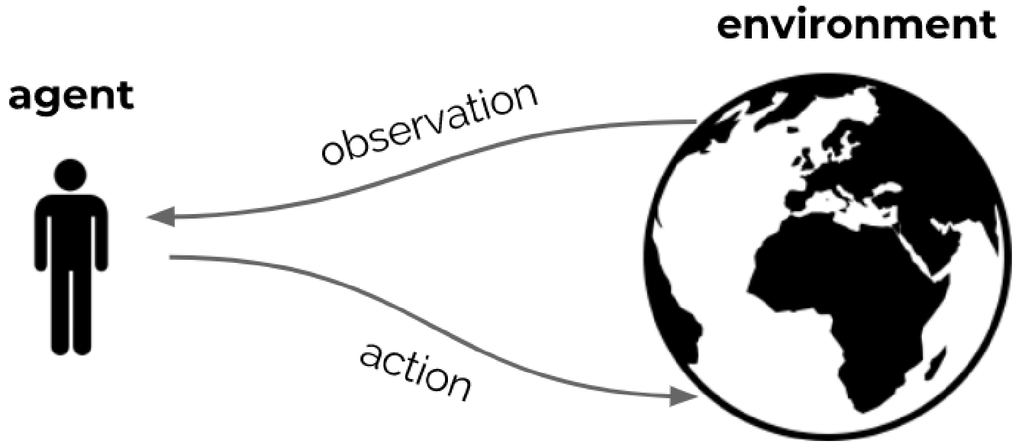


- Reinforcement learning is the science of learning to make decisions
- Agents can learn a **policy**, **value function** and/or a **model**
- The general problem involves taking into account **time** and **consequences**
- Decisions affect the **reward**, the **agent state**, and **environment state**
- Learning is **active**: decisions impact data

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423



- Last lecture:
  - If we know a model of the MDP we can use planning by dynamic programming to solve it.
- This lecture:
  - If we don't know the model of the MDP we can use sampling methods

Slides based on DeepMind's Reinforcement Learning Lectures

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Lecture 18: Model-free Reinforcement Learning

Learning Outcomes

You will learn about:

1. The simpler framework of multi-armed bandits and the exploration-exploitation tradeoff
2. Model-free prediction to estimate values in an unknown MDP: Monte Carlo, Temporal-Difference (TD) Learning, and Multi-step TD learning
3. Model-free control to optimise values in an unknown MDP: SARSA and Q-learning algorithms
4. Optimistic initialization of the value function to help exploration

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Exploration vs Exploitation

- Learning agents need to trade off two things
  - **Exploitation**: Maximize performance based on current knowledge
  - **Exploration**: Increase knowledge
- We need to gather information to make the best overall decisions
- The best long-term strategy may involve short-term sacrifices

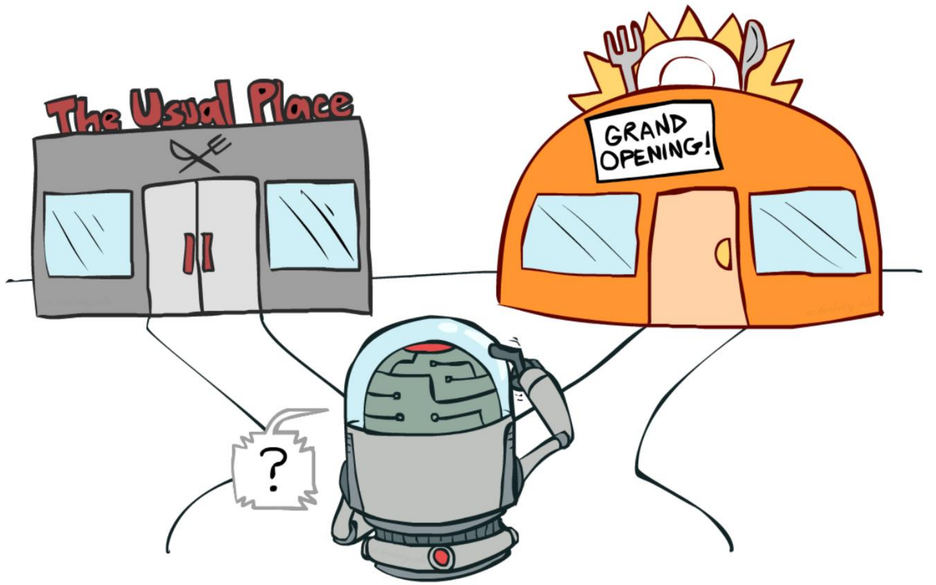


Image source: UC Berkeley AI course slide, lecture 11.

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Multi-armed bandits

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

A simpler setting: the multi-armed bandit

- The environment is assumed to have only a **single state**
- ⇒ actions no longer have long-term consequences in the environment
- ⇒ actions still do impact **immediate reward**
- ⇒ other observations can be ignored
- We discuss how to learn a policy in this setting

Slides based on DeepMind's Reinforcement Learning Lectures

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

The multi-armed bandit

- A multi-armed bandit is a set of distributions $\{\mathcal{R}_a | a \in \mathcal{A}\}$
- $\mathcal{A}$ is a (known) set of actions (or “arms")
- $\mathcal{R}_a$ is a distribution on rewards, given action *a*
- At each step *t* the agent selects an action $A_t \in \mathcal{A}$
- The environment generates a reward $R_t \sim \mathcal{R}_{A_t}$
- The goal is to maximise cumulative reward $\sum_{i=1}^{t} R_i$
- We do this by learning a **policy**: a distribution on $\mathcal{A}$

Slides based on DeepMind’s Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

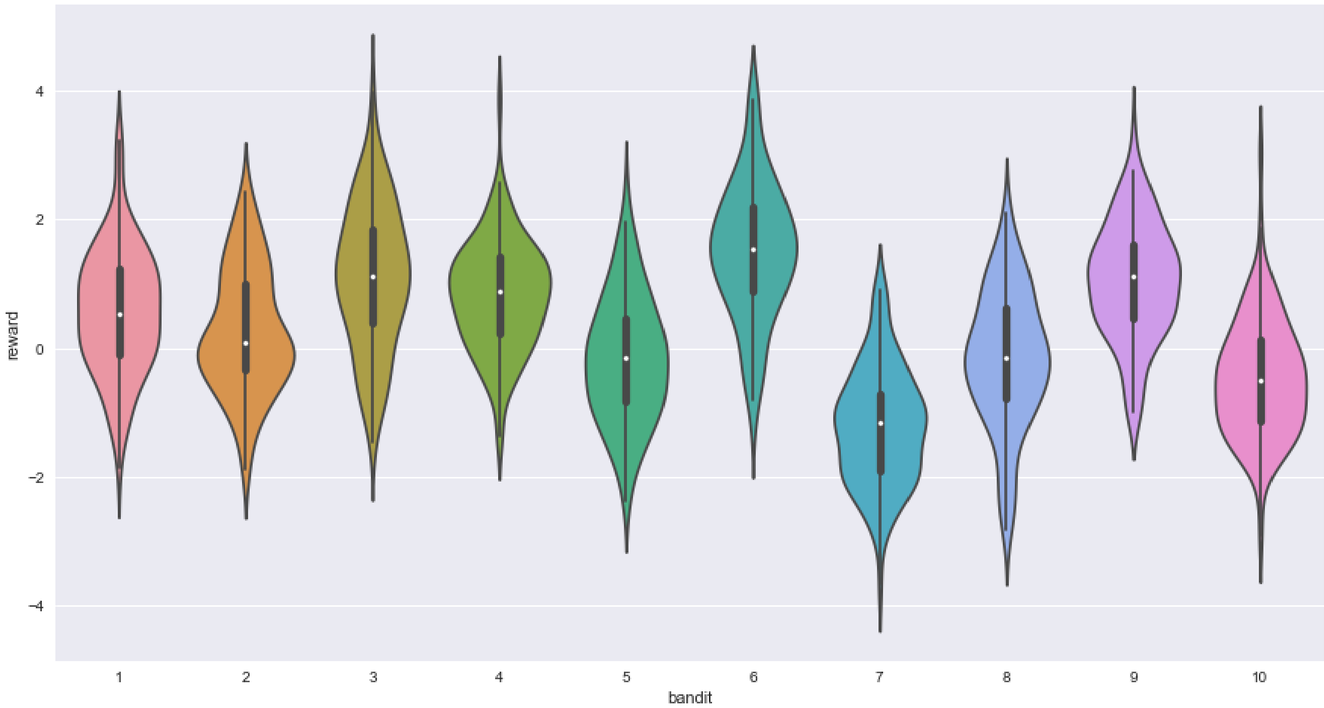
The multi-armed bandit

One-armed bandit
- One slot machine
- Actions are independent

- K-armed bandit
  - K slot machines
  - Different reward distributions

- Goal is to find the slot machine that is expected to give us more reward

Image source



Which arm has the highest median reward?

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

The multi-armed bandit: industrial applications

- **Clinical trials:** one of the main practical problems that uses multi-armed bandits.
  - To evaluate K possible treatments for a disease, pool of N patients is partitioned randomly into K groups.
  - Reward: 1 if the treatment is successful else 0.
  - After a while the majority of the patients can be put to the best found treatment.
- **Online ad-placement:** Deciding which advertisement to display on the webpage.
- **Website optimization:** sequentially choosing design elements (font, images, layout) for the web page to maximize click through rate
- **Network packet routing:** reward is the time it takes to deliver a packet from source to destination and there are K different paths available.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Values

- The **action value** for action *a* is the expected reward

$$q(a) = \mathbb{E}\left[R_t | A_t = a\right]$$

- The **optimal value** is

$$v_* = \max_{a \in \mathcal{A}} q(a) = \max_a \mathbb{E}\left[R_t \mid A_t = a\right]$$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Algorithms

- There are various algorithms for finding the policy:
  - Greedy
  - ε-greedy
  - UCB
  - Thomson sampling
  - Policy gradients
- We will discuss greedy and ε-greedy:
  - They use **action value estimates** $Q_t(a) \approx q(a)$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Action value estimates

- The **action value** for action *a* is the expected reward $q(a) = \mathbb{E}[R_t | A_t = a]$
- A simple estimate is the average of the sampled rewards: $Q_t(a) = \frac{\sum_{n=1}^{t} \mathcal{I}(A_n = a) R_n}{\sum_{n=1}^{t} \mathcal{I}(A_n = a)}$

$\mathcal{I}(\cdot)$ is the **indicator** function $\mathcal{I}(\text{True}) = 1$ and $\mathcal{I}(\text{False}) = 0$

- The **count** for action *a* is $N_t(a) = \sum_{n=1}^{t} \mathcal{I}(A_n = a)$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Action value estimates

- This can also be updated incrementally:

$$Q_t(A_t) = Q_{t-1}(A_t) + \alpha_t \underbrace{(R_t - Q_{t-1}(A_t))}_{\text{error}},$$

$$\forall a \neq A_t \; : Q_t(a) = Q_{t-1}(a)$$

with $\alpha_t = \frac{1}{N_t(A_t)}$ $N_t(A_t) = N_{t-1}(A_t) + 1$ $N_0(a) = 0.$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

The greedy policy

- One of the simplest policies is **greedy**:
  - Select action with highest value: $A_t = \underset{a}{\text{argmax}}\ Q_t(a)$
  - Equivalently: $\pi_t(a) = \mathcal{I}(A_t = \underset{a}{\text{argmax}}\ Q_t(a))$
  - **Pure exploitation**: can get stuck on a suboptimal action forever

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

ε-greedy algorithm

- The **ε-greedy** algorithm:
  - With probability 1-ε select greedy action: $a = \underset{a \in \mathcal{A}}{\text{argmax}}\ Q_t(a)$
  - With probability ε select a random action
  - Equivalently: $$\pi_t(a) = \begin{cases} (1-\epsilon) + \epsilon/|\mathcal{A}| & \text{if } Q_t(a) = \max_b Q_t(b) \\ \epsilon/|\mathcal{A}| & \text{otherwise} \end{cases}$$
  - ε-greedy **continues to explore**

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Quiz

In the **ε-greedy** algorithm, setting ε=0 results in the greedy algorithm (pure exploitation), and setting ε=1 results in pure exploration. True or False?

True

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Model-free Prediction

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Monte Carlo Algorithms

- We can use experience samples to learn without a model
- Monte Carlo learning: direct sampling of episodes
- MC is model-free: no knowledge of MDP is required, only samples

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Monte Carlo Policy Evaluation

- Goal: learn v_π from episodes of experience under policy π

$$S_1, A_1, R_2, \ldots, S_k \sim \pi$$

- The return is the total discounted reward (for an episode ending at time T>t):

$$G_t = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{T-t-1} R_T$$

- The value function is the expected return:

$$v_\pi(s) = \mathbb{E}\left[G_t \mid S_t = s, \pi\right]$$

- We can just use sample average return instead of expected return
- We call this Monte Carlo policy evaluation

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Disadvantages of Monte Carlo Learning

- MC algorithms can be used to learn value predictions
- But when episodes are long, learning can be slow
  - … we have to wait until an episode ends before we can learn
  - … return can have high variance

- Are there alternatives?

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Temporal-Difference Learning

- Previous lecture: Bellman equations

$$v_\pi(s) = \mathbb{E}\left[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t \sim \pi(S_t)\right]$$

- Previous lecture: Approximate by iterating

$$v_{k+1}(s) = \mathbb{E}\left[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s, A_t \sim \pi(S_t)\right]$$

- We can sample this!

$$v_{t+1}(S_t) = R_{t+1} + \gamma v_t(S_{t+1})$$

- This is likely quite noisy – better to take a small step (with parameter α):

$$v_{t+1}(S_t) = v_t(S_t) + \alpha_t\Big(\underbrace{R_{t+1} + \gamma v_t(S_{t+1})}_{\text{target}} - v_t(S_t)\Big)$$

Slides based on DeepMind's Reinforcement Learning Lectures

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Temporal-Difference Learning

- Prediction setting: learn v_π online from experience under policy π
- Monte-Carlo
  - Update value $v_n(S_t)$ towards sampled return G_t

$$v_{n+1}(S_t) = v_n(S_t) + \alpha\,(G_t - v_n(S_t))$$

- Temporal-difference learning:
  - Update value $v_n(S_t)$ towards estimated return $R_{t+1} + \gamma\, v(S_{t+1})$

$$v_{t+1}(S_t) \leftarrow v_t(S_t) + \alpha \left(\overbrace{\underbrace{R_{t+1} + \gamma v_t(S_{t+1})}_{\text{target}} - v_t(S_t)}^{\text{TD error}} \right)$$

  - $\delta_t = R_{t+1} + \gamma\, v_t(S_{t+1}) - v_t(S_t)$ is called the TD error

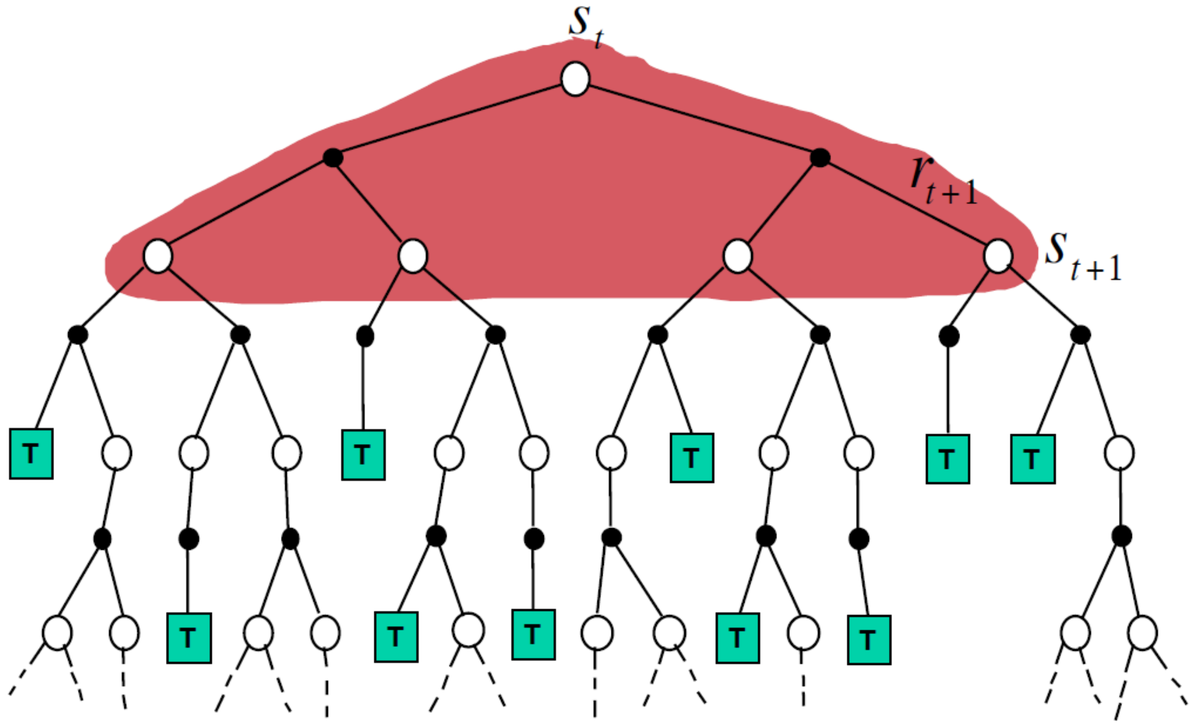
Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Dynamic Programming Backup

$$v(S_t) \leftarrow \mathbb{E}\left[R_{t+1} + \gamma v(S_{t+1}) \mid A_t \sim \pi(S_t)\right]$$

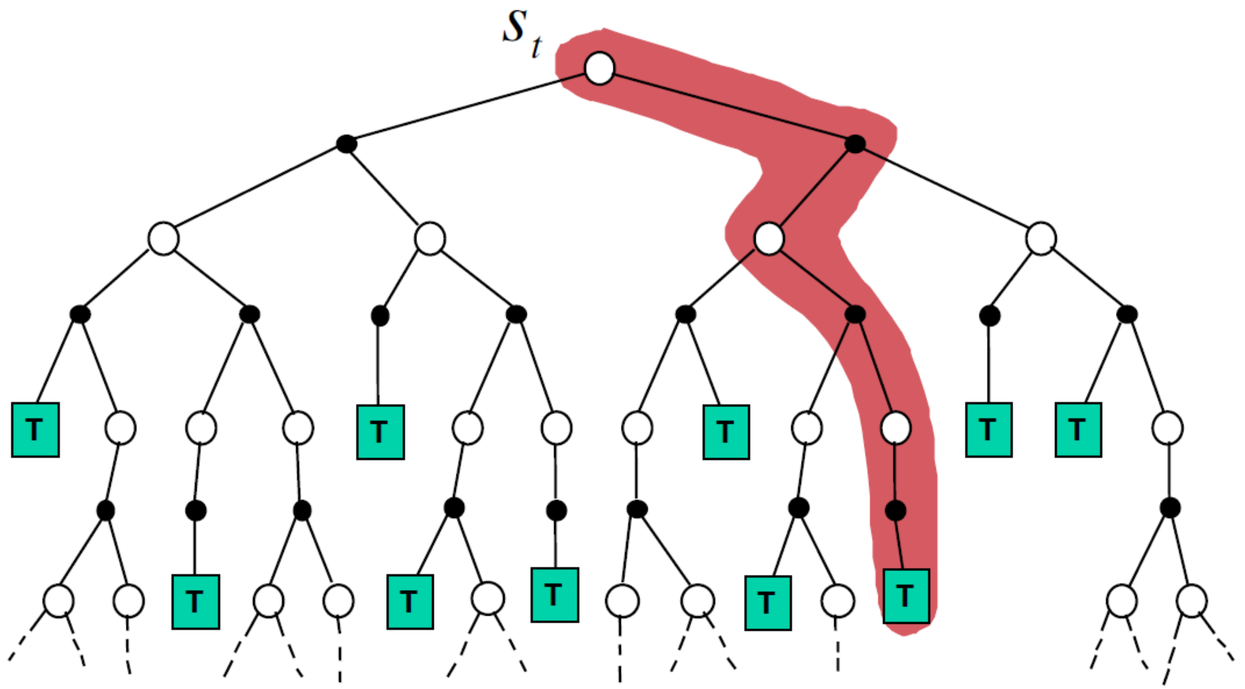


Slides based on DeepMind's Reinforcement Learning Lectures

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Monte Carlo Backup

$$v(S_t) \leftarrow v(S_t) + \alpha\,(G_t - v(S_t))$$



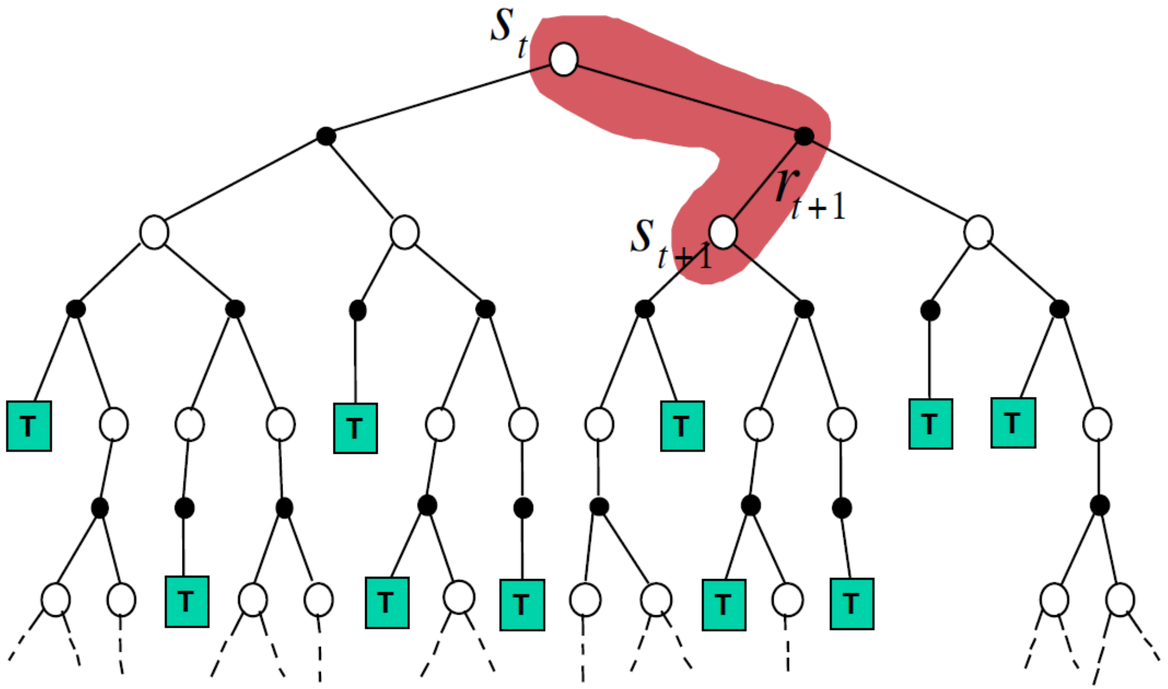
Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Temporal-Difference Backup

$$v(S_t) \leftarrow v(S_t) + \alpha \left(R_{t+1} + \gamma v(S_{t+1}) - v(S_t)\right)$$



Slides based on DeepMind's Reinforcement Learning Lectures

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Bootstrapping and Sampling

- **Bootstrapping**: update involves an estimate (i.e., value of the next state)
  - MC does not bootstrap
  - DP bootstraps
  - TD bootstraps

- **Sampling**: update samples an expectation
  - MC samples
  - DP does not sample (i.e., computes the update as it has knowledge of the model)
  - TD samples

Slides based on DeepMind's Reinforcement Learning Lectures

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Temporal-Difference Learning

- We can apply the same idea to action values
- TD learning for action values:
  - Update value $q_t(S_t, A_t)$ towards estimated return $R_{t+1} + \gamma\, q(S_{t+1}, A_{t+1})$

$$q_{t+1}(S_t, A_t) \leftarrow q_t(S_t, A_t) + \alpha \left(\overbrace{\underbrace{R_{t+1} + \gamma q_t(S_{t+1}, A_{t+1})}_{\text{target}} - q_t(S_t, A_t)}^{\text{TD error}} \right)$$

  - This algorithm is known as **SARSA**, because it uses $(S_t, A_t, R_{t+1}, S_{t+1}, A_{t+1})$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Temporal-Difference Learning

- TD is model-free (no knowledge of MDP) and learns directly from experience
- TD can learn from incomplete episodes by bootstrapping
- TD can learn during each episode

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Advantages and Disadvantages of MC vs TD

- TD can learn before knowing the final outcome
  - TD can learn online after every step
  - MC must wait until the end of episode before return is known
- TD can learn without the final outcome
  - TD can learn from incomplete sequences
  - MC can only learn from complete sequences
  - TD works in continuing (non-terminating) environments
  - MC only works for episodic (terminating) environments
- TD is independent of the temporal span of the prediction
  - TD can learn from single transitions
  - MC must store all predictions (or states) to update at the end of an episode

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

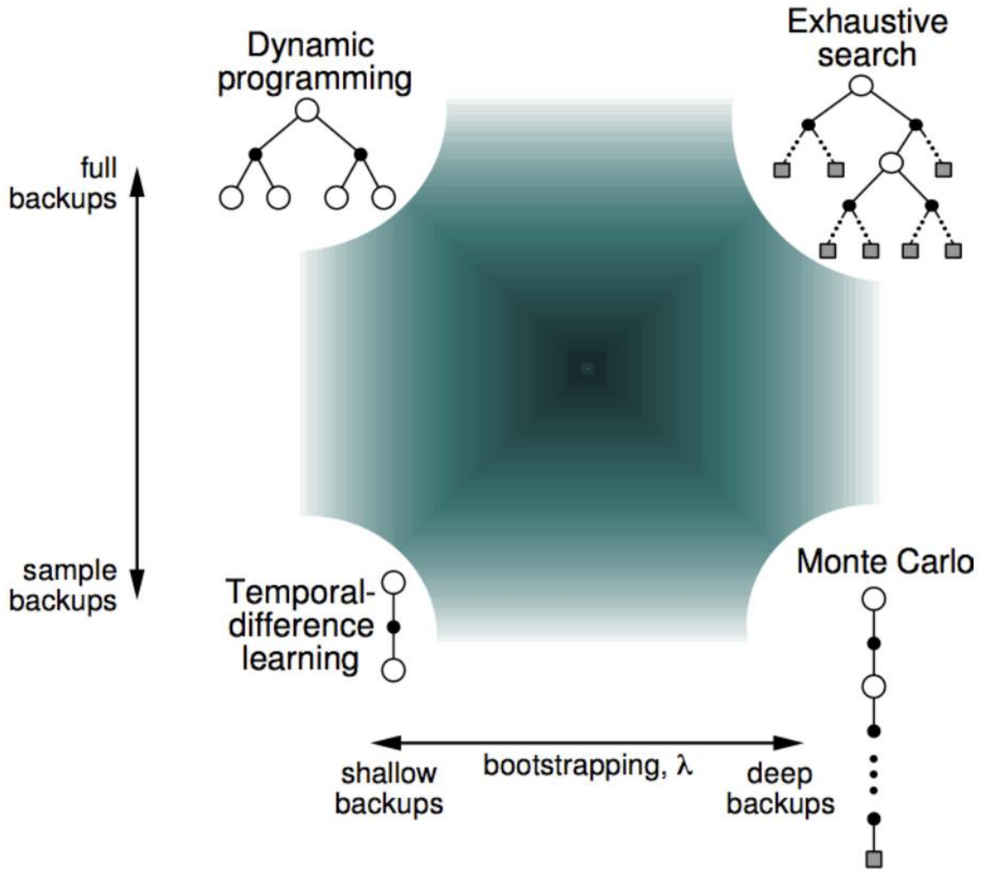
Master programmes in Artificial Intelligence 4 Careers in Europe

Between MC and TD: Multi-Step TD

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Unified View of RL



Slides based on DeepMind's Reinforcement Learning Lectures

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Multi-Step Updates

- TD uses value estimates which might be inaccurate
- In addition, information can propagate back quite slowly
- In MC information propagates faster, but the updates are noisier
- We can go in between TD and MC

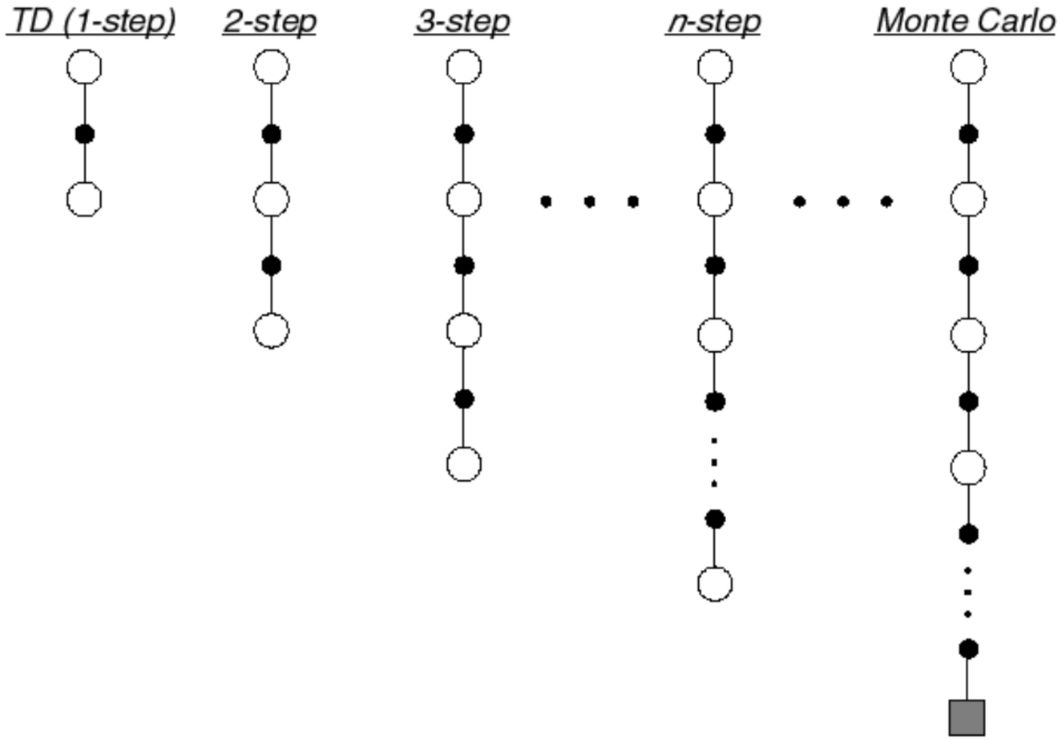
Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Multi-Step Prediction

- Let TD target look n steps into the future



Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Multi-Step Returns

- Consider the following n-step returns for $n = 1,2,\infty$:

$$
\begin{aligned}
n = 1 \quad & (TD) \quad G_t^{(1)} = R_{t+1} + \gamma v(S_{t+1}) \\
n = 2 \quad & \quad\;\; G_t^{(2)} = R_{t+1} + \gamma R_{t+2} + \gamma^2 v(S_{t+2}) \\
\vdots \quad & \quad\;\; \vdots \\
n = \infty \quad & (MC) \quad G_t^{(\infty)} = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{T-t-1} R_T
\end{aligned}
$$

- In general, the n-step return is defined by

$$G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{n-1} R_{t+n} + \gamma^n v(S_{t+n})$$

- Multi-step TD learning

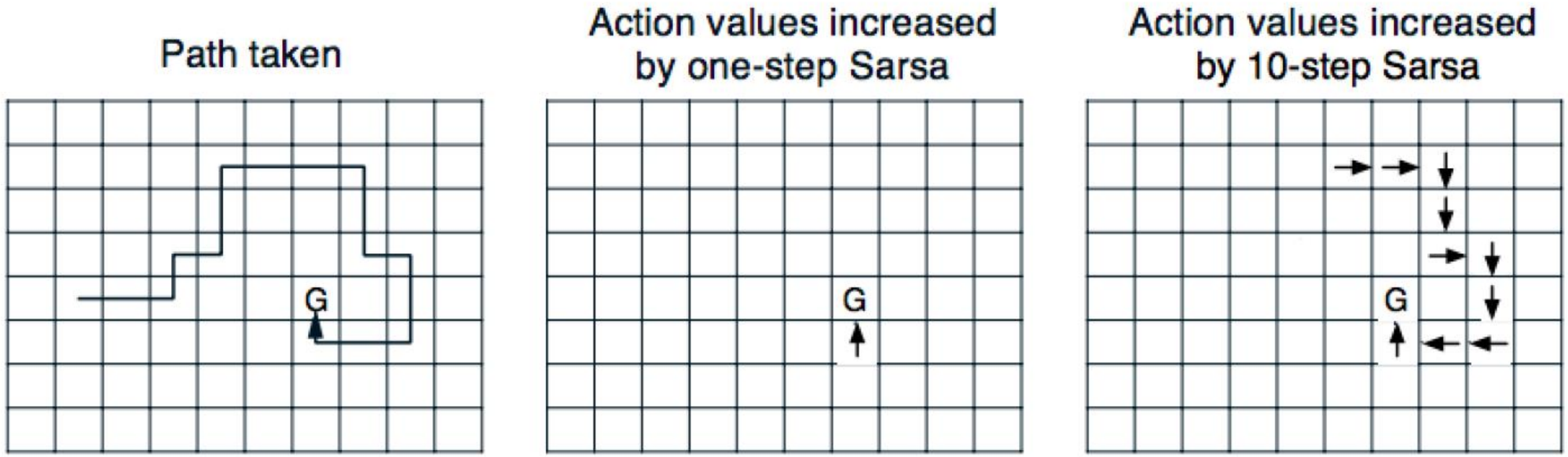
$$v(S_t) \leftarrow v(S_t) + \alpha \left(G_t^{(n)} - v(S_t) \right)$$

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Grid Example



(Reminder: SARSA is TD for action values $q(s, a)$)

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Temporal-Difference Learning for Control

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

On and Off-Policy Learning

- On-policy learning
  - Learn about behaviour policy π from experience sampled from π
  - SARSA algorithm
- Off-policy learning
  - Learn about target policy π from experience sampled from μ
  - Learn “counterfactually” about other things you could do: “what if…?”
  - E.g., “What if I would turn left?” => new observations, rewards?
  - E.g., “What if I would play more defensively?” => different win probability?
  - E.g., “What if I would continue to go forward?” => how long until I bump into a wall?

Slides based on DeepMind’s Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

SARSA algorithm

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize S
  Choose A from S using policy derived from Q (e.g., ε-greedy)
  Loop for each step of episode:
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ε-greedy)
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma Q(S', A') - Q(S, A)\big]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  until S is terminal

Source: Sutton & Barto (2018). Reinforcement Learning – An Introduction (2nd edition)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

SARSA algorithm

- Tabular SARSA converges to the optimal action-value function, if the policy is Greedy in the Limit of Infinite Exploration (GLIE)
  - All state-action pairs are explored infinitely many times
  - The policy converges to a greedy policy
  - For example, ε-greedy with ε decaying over time

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Off-Policy Learning

- Evaluate target policy $\pi(a|s)$ to compute $v_\pi(s)$ or $q_\pi(s,a)$
- While using behaviour policy $\mu(a|s)$ to generate actions
- Why is this important?
  - Learn from observing humans or other agents (e.g., from logged data)
  - Re-use experience from old policies (e.g., from your own past experience)
  - Learn about multiple policies while following one policy
  - Learn about greedy policy while following exploratory policy
- Q-learning estimates the value of the greedy policy

$$q_{t+1}(s,a) = q_t(S_t, A_t) + \alpha_t \left(R_{t+1} + \gamma \max_{a'} q_t(S_{t+1}, a') - q_t(S_t, A_t) \right)$$

  - Acting greedy all the time would not explore sufficiently

Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Q-learning algorithm

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    Choose A from S using policy derived from Q (e.g., ε-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
    $S \leftarrow S'$
  until S is terminal

Source: Sutton & Barto (2018). Reinforcement Learning – An Introduction (2nd edition)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Q-learning algorithm

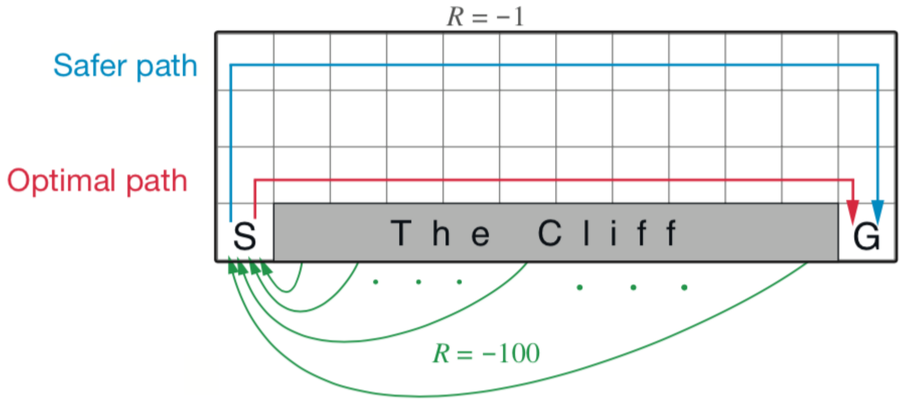
- Q-learning control converges to the optimal action-value function, as long as we take each action in each state infinitely often
- No need for greedy behaviour!
- Works for any policy that eventually selects all actions sufficiently often
- Requires appropriately decaying step sizes

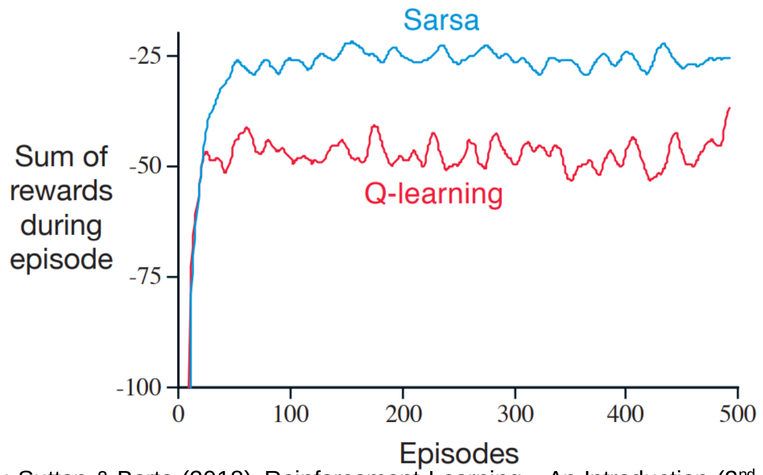
Slides based on DeepMind's Reinforcement Learning Lectures

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Cliff Walking Example





Source: Sutton & Barto (2018). Reinforcement Learning – An Introduction (2nd edition)

- Q-learning learns values for the optimal policy, that which travels right along the edge of the cliff.
- This results in occasionally falling off the cliff because of the ε-greedy action selection
- SARSA takes action selection into account and learns the longer but safer path through the upper part of the grid.
- Although Q-learning learns the values of the optimal policy, its online performance is worse than that of SARSA, which learns the roundabout policy
- If ε were gradually reduced, then both methods would asymptotically converge to the optimal policy.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Optimistic Initialization

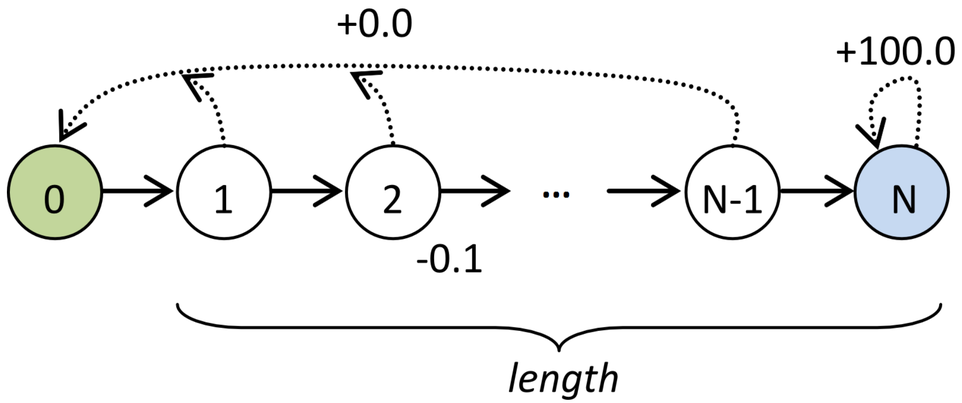
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Optimistic Initialization

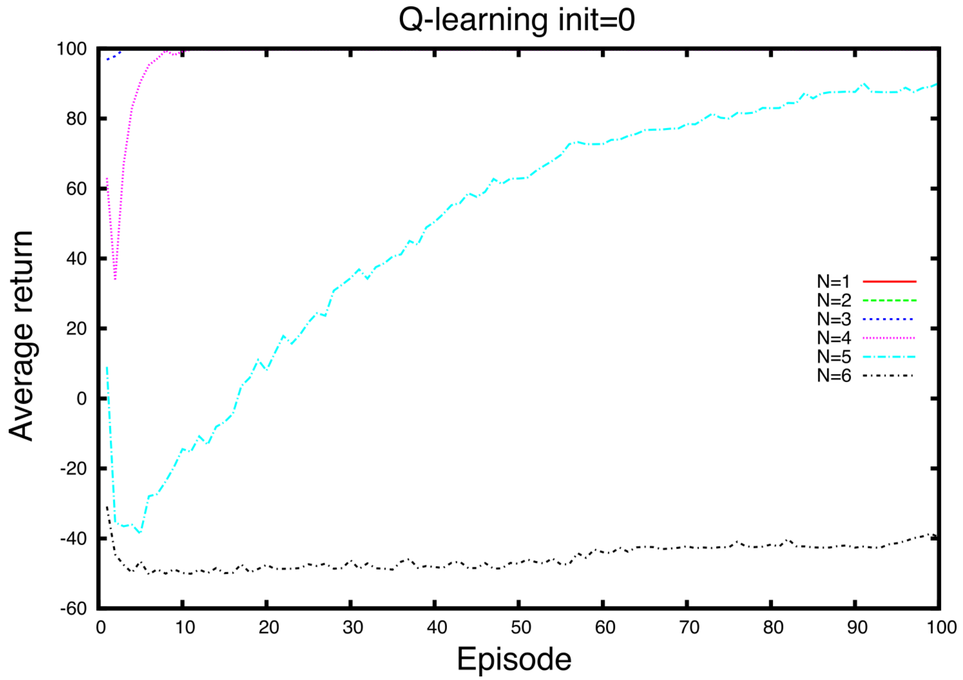
- Simple approach to ensure exploration of state(-action) space
- Instead of initializing the value function to 0, initialize to r_{max} / (1-γ)
- Acting greedily (instead of ε-greedy) allows the algorithm to try each action in every state at least once
- Doing so the agent is able to discover a delayed reward

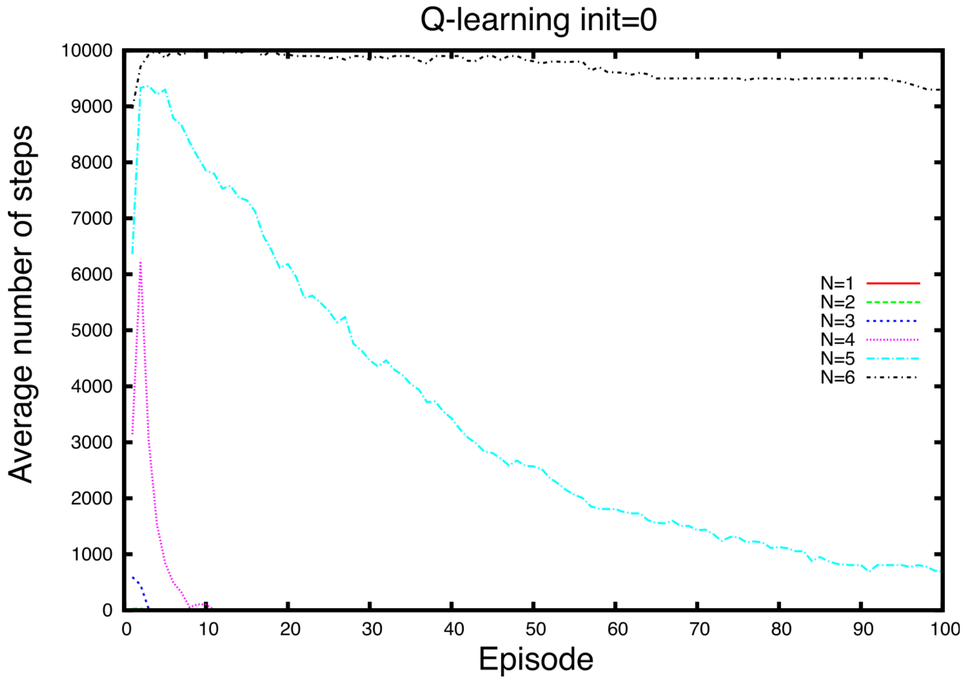
- Example: Combination-lock MDP



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Combination-lock MDP results: $Q_{init} = 0$





Parameters

ε=0.1

α=0.95

γ=0.95

episodes=100

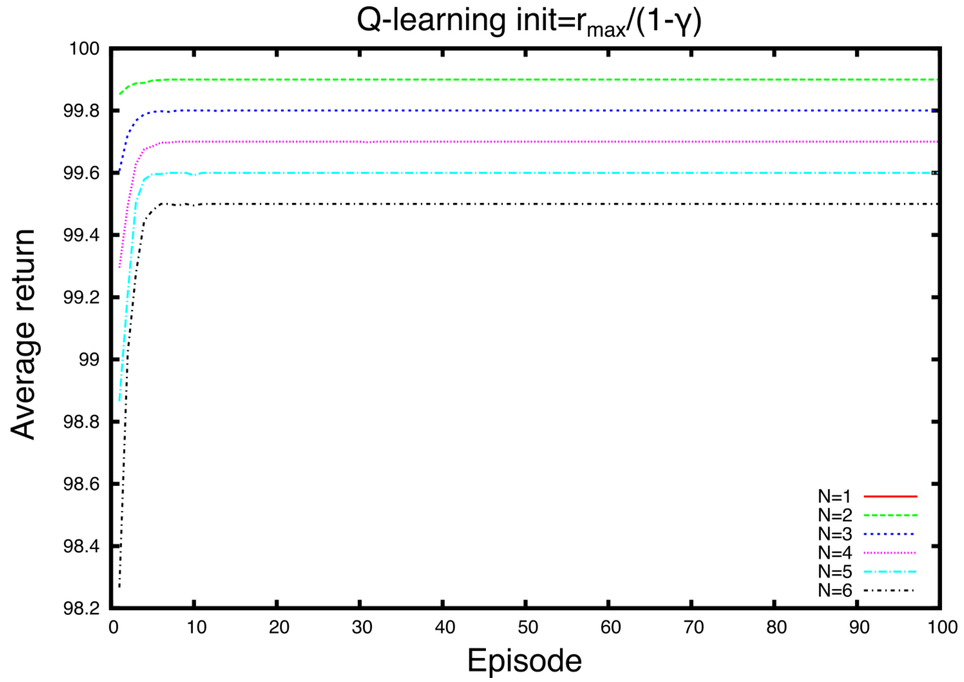
max_steps=10K

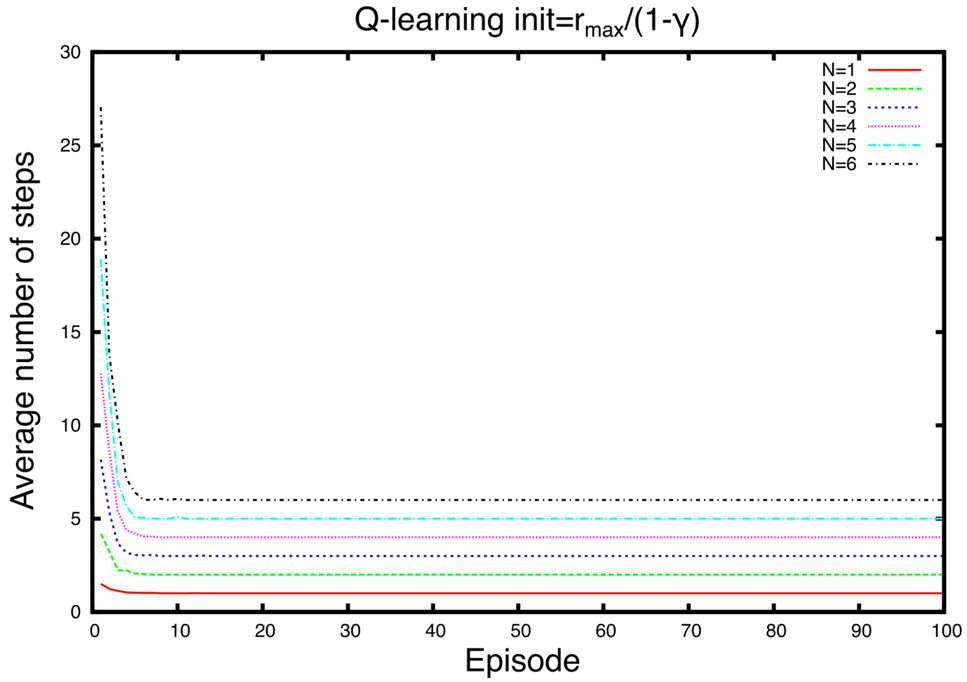
runs=100

Qinit=0

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Combination-lock MDP results: $Q_{init} = r_{max}/1-\gamma$





Parameters

ε=0

α=0.95

γ=0.95

episodes=100

max_steps=10K

runs=100

Qinit=2000

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus - MSc Artificial Intelligence

MAI612 - MACHINE LEARNING

Lecture 19: Revision and Overview of Advanced ML

Vassilis Vassiliades, PhD

Winter Semester 2022/23

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Revision

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to ML

- ML is a subfield of AI that uses data to teach computers how to predict and act
  - Can be seen as program discovery from data
  - Its goal is to generalize to unseen data, rather than memorizing
- ML has countless applications:
  - Spam filter
  - Grouping customers to drive marketing actions
  - Predicting whether a tumor is benign or malignant from (from features such as its size)
  - Predicting house prices (from features such as size and number of bedrooms)
  - Animal sound recognition
  - Image recognition (e.g., faces, products, characters, medical images)
  - Natural language translation
  - Recommender systems (e.g., movies, music, news)
  - Identifying whether an industrial machine is faulting from its noise levels
  - Teaching a robot dog to walk

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to ML

- There are 3 main types of ML approaches:
  - Supervised learning: when we have labels
    - Classification
    - Regression
  - Unsupervised learning: when we do not have labels
    - Clustering
    - Dimensionality Reduction
    - Anomaly Detection
    - Matrix completion (e.g. recommender systems)
  - Reinforcement learning: when the problem involves sequential decision making
- The ML project lifecycle:
  - Strategy -> Data Preparation <-> Model Development <-> Model Deployment

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Preparation

- Types of data:
  - Tabular, Text, Images, Signals (audio), Video, Point clouds, Graphs…
- Data Collection: acquiring, integrating, labelling
  - Need to have diversity
- Data Preprocessing:
  - Data cleaning: fix inconsistencies, missing data, remove duplicates
  - Data encoding: one-hot encoding, ordinal encoding
- Data Visualization: see patterns, problems
  - Can use dimensionality reduction

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Data Preparation

- Data Transformation:
  - Feature scaling:
    - min-max normalization: when we know the feature ranges
    - Mean-normalization: when we don’t know the feature ranges
  - Feature selection: choose a subset of the features
    - L1-regularization can sometimes perform feature selection
  - Feature Extraction: features with lower dimensionality than the original
    - PCA, Autoencoders
  - Feature Engineering: additional features
    - Polynomial features, domain knowledge
  - Data Augmentation: additional data points (when we have small or imbalanced dataset)
  - Data Sampling: remove points (when we have a huge or imbalanced dataset)
- Dataset splitting: training, validation (or cross-validation), test sets

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression

- Regression is the supervised learning problem of predicting a continuous value
- K-nearest neighbor regression
  - Nonparametric model
  - Prediction is the average of the K nearest neighbors
  - K=1: noisy, 1<K<m better captures the trend, K=m always produces the average of all
  - Strengths: no training time, handles nonlinearities, …
  - Weaknesses: prediction becomes slower as the dataset becomes larger, …
- Linear regression
  - Parametric model: 1 parameter per feature + intercept term
  - Prediction is the dot product of parameter vector and input vector (weighted sum)
  - Learning can be done either using gradient-based methods (iterative) or computing the analytic solution (non-iterative)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Regression

- Linear regression optimizes the MSE, which has a convex shape (single optimum)
- Gradient descent starts from a random point (parameter vector), and alternates by computing the partial derivatives of a function at that point and modifying the point by adding a value proportional to the negative gradient
  - A small learning rate results in slower convergence, a large learning rate may result in divergence
  - Need to use feature scaling with gradient descent
- The analytic solution is not iterative (does not start from a random initial parameter vector)
  - Fast for small number of features and points
  - Cannot use it for very large number of features (e.g., thousands) and points (e.g., thousands or millions)
  - Does not use regularization (could overfit when using a large number of features)
- Linear regression
  - Strengths: constant prediction time, analytic solution easy to implement,…
  - Weaknesses: cannot model nonlinear relationships,…
- Weakness can be addressed using polynomial features

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Classification

- Classification is the supervised learning problem of predicting a discrete value
- K-nearest neighbor classification
  - Prediction is the majority vote of the K nearest neighbors
  - K=1: fit the noise, K=m always predict the majority class
- Logistic regression
  - Simple method for binary classification
  - Feeds the output of linear regression through the sigmoid function which makes it in [0,1]
  - The output is seen as the estimated probability of predicting the positive class
  - Uses a default (probability) threshold of 0.5 for placing the decision boundary
  - Decision boundary of logistic regression is linear, can be nonlinear if we use polynomial features
- Logistic regression optimizes the Cross-Entropy error which has a convex shape
  - We use gradient-based methods to find the minimum
  - The CE error penalizes the model a lot if its predicted probability is very far from the actual

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Classification

- Error analysis for binary classifiers
  - True positives, True Negatives, False Positives and False Negatives
  - Metrics: Accuracy, Precision, TP rate, FP rate, F1-score
  - ROC curve: plots the FP rate and the TP rate for all classification thresholds
  - AUC score: a single metric based on ROC which can be used to compare classifiers
- Multiclass classification
  - One-vs-rest: train K binary classifiers; prediction is the class of the most confident classifier
  - Softmax: uses one-hot encoding of classes; prediction is a probability distribution over K classes
- Confusion matrix for multiclass classification
  - TPs: diagonal
  - TP rate (class A): TP(class A) / number of samples for class A
  - Precision: TP(class A) / number of predictions for class A
  - Accuracy: number of TPs / total number of samples

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model Evaluation and Improvement

- We want our models to exhibit generalization capabilities instead of memorizing the training set.
- Generalization: good performance on unseen data, drawn from the same distribution
- Underfitting: too simple model / high bias
- Overfitting: too complex model / fit the noise rather than the trend / high variance
- The bias-variance tradeoff is the conflict of trying to minimize both bias and variance
- Practically, we achieve that by splitting the dataset into a training, validation and test sets, and selecting the model that has the lowest validation error.
- k-fold cross validation:
  - splits the training+validation dataset into k subsets, and trains k independent models where subset i is used for validation and the remaining as training set
  - the performance of the model is the average validation error over all k subsets
  - used when the dataset is small, because of its complexity in training k models rather than just one.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model Evaluation and Improvement

- Learning curves can be used to inspect whether we need to acquire more data
  - More data typically benefits high variance models but not high bias ones
- Regularization is a penalty given to the loss function of high variance models to reduce the magnitude of their parameters, and thus, their complexity
  - L1 regularization uses the absolute value norm and can sometimes be used for feature selection
  - L2 regularization users the Euclidean norm and typically produces a better fit than L1
- Hyperparameter tuning is the process of varying the hyperparameters of models and learning algorithms, and select the combination that results in the lowest validation error.
- Model improvement can be achieved using ensembles: training multiple models instead of one and aggregating their outputs
- Improve high bias models through: complexification, ensembles
- Improve high variance models through: hyperparameter tuning, regularization, ensembles, more data

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Trees and Forests

- Decision trees are models that have a natural if-then-else structure, thus being interpretable and fast
- For a given dataset, there can be multiple decision trees that classify the data
- An algorithm for learning decision trees needs to choose one that generalizes well by deciding the feature to use for splitting at each node, and when to stop splitting
- Using irrelevant features creates larger decision trees, thus simplicity is preferred
- The best feature to use for splitting is the one that is most informative, i.e., the one that minimizes the disorder aka entropy
- The entropy H takes as input the proportion of positive examples p_+, and as p_+ goes from 0 to 0.5, H increases from 0 to 1, and as p_+ goes from 0.5 to 1, H decreases from 1 to 0.
- Information gain measures the expected reduction in entropy due to splitting on some feature A
  - It is measured as the difference between the entropy of the initial set, and the weighted sum of the entropies in the branches
  - We want to maximize information gain, or equivalently minimize the weighted sum
- To split on a continuous variable, we first calculate all possible unique thresholds (using the midpoints of pairs of sorted points), and select the one with the highest information gain.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Trees and Forests

- Regression trees:
  - predict the average of the values in their leaf nodes
  - use the variance instead of the entropy, and the variance reduction instead of the information gain
- Ensemble methods:
  - typically have lower generalization error than single learners
  - can reduce both bias and variance
  - they rely on diverse learners that produce different errors
  - aggregation function: average for regression, majority vote for classification
- Bagging trains models in parallel by varying their training data using sampling with replacement
- Random forests use bagging with the additional step of randomizing the feature choice
- Boosting trains models incrementally by focusing on previously misclassified examples
- Stacking is a method that trains models typically at 2 levels, where the predictions of the models at level 1 become training data for a model at level 2 which learns how to combine them.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel methods

- Classification problems often require nonlinear decision boundaries
  - These can be constructed using feature engineering, e.g., adding polynomial features
  - As we add features, we increase the dimensionality of the input which often makes the problem linearly separable, i.e., easier to solve in higher dimensions
  - However, this approach can exponentially increase the number of parameters to be learned, and has the disadvantage of needing to compute features explicitly
- Kernel trick: the technique of using kernel functions (i.e., similarity functions over pairs of raw data points) that allow the model to operate in a high-dimensional space without explicitly transforming the raw data in the higher dimensional space.
- Kernel examples: polynomial, Gaussian
- We can make valid kernels by combining valid kernels through addition, multiplication and scaling with a positive constant
- Kernel methods are instance-based learners which compute the similarity of an input with stored training points and multiply this by some learned weight which is specific for that particular training point

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel methods

- Support Vector Machines are binary classification models that learn the separating hyperplane with the maximum margin, and use kernels to deal with nonlinearly separable problems
- Maximum margin decision boundary:
  - minimizes the generalization error, as it ensures that points are classified far from the separating hyperplane
  - computed by finding the support vectors, i.e., the training points closest to the separating hyperplane
- Trained SVMs only need to keep the support vectors to make a prediction, which makes the prediction fast
- We can do multi-class classification with SVMs using the one-vs-rest approach
- Support Vector Regression extends SVMs to regression problems
  - fits a line to the data with a margin (tube) around it
  - ignores points inside the tube because their combined error is small

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel methods

- Radial-basis function (RBF) networks are kernel methods that compute nonlinear features of the input based on proximity to fixed centres.
  - The output is a linear combination of coefficients and features
  - Mainly used for regression, in particular for exact interpolation and approximation
- The centre of the kernel can be at a training point (exact interpolation), however, it can be elsewhere
- Commonly used kernel: Gaussian
  - Value of 0 if distance between queried point and kernel centre is large
  - Value of 1 if distance between queried point and kernel centre is 0 (same point)
- If σ (Gaussian width) is large: smoother fit, thus, higher bias, lower variance
- We can use the normal equations to compute the RBF network weights for interpolation and approximation (fixed centres)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel methods

- We can adapt the weights, centre coordinates and Gaussian widths of an RBF network using gradient descent: results in better fit
- Unnormalized RBFs: more localized, need more (or wider) to cover the input space
- Normalized RBF networks may exhibit better generalization with fewer nodes
- RBF networks can be used for classification by feeding the output of regression RBF network into a sigmoid function (similarly to logistic regression)
- Gaussian processes: nonparametric probabilistic regression models that output not only the prediction but also the confidence in the prediction
  - Probability distribution over functions: can sample functions from it
  - Good for low-data problems
  - Can be provided with prior knowledge about the function we want to model

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Kernel methods

- Applications of SVMs:
  - Cancer prediction from features
  - Whether a mushroom is poisonous or not
  - Spam filter (with low-dimensional features)
  - Predict whether the income of a person exceeds 50K
- Applications of RBFs:
  - Predict price of houses based on size + number of bedrooms
  - Predict price of stocks based on time series data
  - Predict the value of a continuous function based on sampled points
  - Predict medical insurance costs based on age, gender, BMI, smoker, …
  - Predict wine quality (1-10) based on chemical characteristics

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neural Networks

- Artificial neurons compute the weighted sum of their inputs and feed it through an activation function which then becomes their output
- Activation functions:
  - Heaviside step (non-differentiable) → Perceptron model
  - Linear → Linear regression
  - Sigmoid → Logistic regression
- Perceptrons are linear classifiers
  - By combining multiple perceptrons in layers we can classify nonlinearly separable problems
  - E.g., XOR can be solved using 2 hidden neurons and 1 output neuron
  - However, we cannot train them using gradient descent when they use the Heaviside step function as it is non-differentiable

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neural Networks

- Multilayer Perceptrons is a synonym for feedforward ANNs (typically with differentiable activation functions)
- As a classifier, an MLP with:
  - 1 hidden layer forms open or convex decision regions
  - 2 hidden layers create arbitrary decision regions
- Forward propagation is the process that feeds a data instance to the input layer of a NN and this is gradually transformed into the output prediction (regression or classification) through some nonlinear transformation.
  - This nonlinear transformation computes features of the input which are learned
  - A second hidden layer computes features as functions of existing features
- Learning in NNs can be done using backpropagation and gradient descent

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neural Networks

- Backpropagation: an efficient way to compute partial derivatives of the error function with respect to each parameter using the chain rule (since a NN is a composition of functions)
- Error function: MSE for regression, Cross-Entropy for classification
- Forward propagation computes the activation (output) of each node, while Backpropagation computes the error (delta) of each node
- Delta (error) of each node: A x B
  - A = derivative of the node's activation function
  - B = derivative of error with respect to the node's output
    - For an output node: this is the derivative of the error function wrt to the activation of the output node
    - For a hidden node i at layer k: this is the sum of all deltas at nodes in layer k+1 (which are connected to node i) multiplied by their connecting weight
- Gradient of the Error wrt to a weight = (delta of postsynaptic node) x (output of presynaptic node)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Neural Networks

- Stochastic GD: weight update after the presentation of every pattern
- Batch GD: weight update after the presentation of all patterns in the training set
- Mini-batch GD: weight update after the presentation of subsets of patterns in the training set
- Stochastic or mini-batch GD are used when we have massive training sets
- Momentum term: memory of previous direction, speeds up learning
- Early stopping: way to prevent overfitting by stopping training when the validation error starts increasing
- We can improve the performance of NN models using regularization, hyperparameter tuning and ensembles
- Learning the NN topology can be done using gradient-free methods, such as evolutionary algorithms.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to Deep Learning

- Deep learning is about learning successive layers of representations
  - Using NNs with more than 2 hidden layers
- Deep Leaning is possible mostly because of hardware advancements (GPUs) and abundance of data
- When we want to detect a particular concept that could be in different places of the input we use weight sharing, i.e., we build a single feature detector for this concept by training the weights of these inputs jointly
  - This helps generalization
  - Example 1: creating a dog image classifier: a dog could appear everywhere in an image
  - Example 2: text completion network: we want the part of the NN that learns what a dog is to be reused every time the NN sees the word dog

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to Deep Learning

- Convolutional networks are NNs more suitable for image data
- They use local receptive fields (filters) and shift them (convolve) over the activation map of the previous layer to create the activation map of the current layer
  - This reduces the number of parameters compared to fully connected feedforward networks
  - Each filter becomes a feature detector over different parts of the input (translation invariance):
    - Layer 1: edge detectors
    - Layer 2: corner detectors
    - Layer 3: parts-of-objects detectors
    - Layer 4: complete-objects detectors

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to Deep Learning

- Sequential data: data ordered into sequences
  - Typically: time series data
- A way to handle sequential data using NNs is by using feedback (delayed) connections: recurrent NNs (RNNs)
  - A recurrent NN creates its own internal representation of time
- We can train RNNs using backpropagation through time
  - We unroll the RNN over time and backpropagate the errors from the last time step to the first
  - We accumulate the gradients and apply gradient descent
  - Unrolled RNNs can become very deep networks
  - When doing so we might have the vanishing or exploding gradients problems

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to Deep Learning

- Echo state networks:
  - use a large sparsely connected hidden layer which is not trained
  - they do not use backprop through time
  - they do not have the vanishing or exploding gradients problems
  - we can compute the analytic solution for a regression problem (similarly to linear regression)
  - good performance in tasks that require fast, adaptive training
  - not good performance in tasks with many variables and long-term dependencies

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to Deep Learning

- Long short-term memory networks:
  - Use gating mechanisms that allow the network to learn what to forget, what to store in memory and what to output
  - Gating: sigmoid multiplied by signal : sigmoid modulates how much of the signal is allowed to pass through
  - Can be applied to tasks requiring long-term dependencies
- Text data:
  - One-hot word representation: sparse, high dimensional, does not generalize well
  - Word embeddings: learned, numerical vector representation
    - Try to capture the meaning of words based on their usage in sentences
    - Words with similar meaning have similar vector representations
    - King – Man + Woman = Queen

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Clustering

- Clustering is the problem of grouping data with similar characteristics
  - We do not have labels that specify the correct outputs
- k-means clustering is a clustering algorithm that alternates between assigning all training points to their closest cluster centroid, and updating the cluster centroids to the average value of all their assigned points
  - Uses a pre-specified number (k) of clusters
  - Randomly initializes the k cluster centroids
- We can avoid local optima by running k-means multiple times and selecting the clustering with the lowest cost
- We choose k using domain knowledge, the Elbow method or the Silhouette score
- Clustering can help supervised learning, e.g., by
  - Finding the initial centres of RBF networks
  - Allowing to use both labelled and unlabelled data to improve generalization

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Dimensionality Reduction

- Dimensionality reduction: transformation of high-dimensional data to low-dimensional space
- Why dimensionality reduction:
  - Data visualization
  - Data compression
    - Can help ML algorithms (supervised learning, clustering)
- Principal Component Analysis
  - Linear transformation into a new coordinate system
  - Maximize variance of low-dimensional data = minimizing reconstruction error
  - Finds n orthogonal vectors each ranked by how much it explains the variance in data
    - These are the principal components or eigenvectors
  - Projection = encoding in low dimensions
  - Reconstruction = decoding from low dimensions to high-dimensions
  - We can choose the number of components based on a desired ratio of explained variance (typically 90-99%)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Dimensionality Reduction

- PCA works well in various problems but it is a linear method
- Nonlinear dimensionality reduction methods address this shortcoming
  - Kernel PCA uses the kernel trick
  - Autoencoders are NNs trained to encode and reconstruct the input
- Manifold learning approaches = nonlinear dimensionality reduction methods that explicitly consider that the data lie on low-dimensional structures embedded in high-dimensional space
  - Isomap: instead of Euclidean distance, uses the geodesic distance
    - Geodesic distance: distance on the manifold
    - Swiss roll example: isomap can unfold it
    - Allows better interpolation as the interpolated points expected to lie on the manifold
  - t-SNE
    - used for visualization
    - stochastic method that preserves local similarities
    - can give different results for different initializations

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Anomaly Detection

- Anomaly detection is the problem of modeling a dataset of normal events and raising an alarm when an unusual event occurs in the future.
  - Normal events are assumed to be concentrated
  - Outlier detection: outliers exist in the training set
  - Novelty detection: no outliers in the training set
- Approaches:
  - Density estimation
  - One-class classification using discriminative models
  - Autoencoders
- Density estimation: Parametric and Non-parametric
  - Build a model of the probability of points
  - Use a threshold (ε) on the probability to classify an anomaly (unlikely point) from a normal point
  - Parametric: fit a Gaussian (Parameters: mean and variance)
  - Non-parametric: kernel density estimation

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Anomaly Detection

- One-class classification using discriminative models:
  - Create a conservative decision boundary
  - One-class SVM: try to encompass all (normal) training data using the smallest hypersphere
  - Isolation Forests: anomalies are data points that have short path lengths in a tree
- Autoencoders:
  - Normal data have low reconstruction error
  - Abnormal data have higher reconstruction error
  - Use a histogram of the errors and decide an anomaly threshold
- Feature engineering can be very important in anomaly detection systems
- Supervised anomaly detection: way to evaluate anomaly detection systems
  - Have small amount of labelled data
  - Training set: normal data, no labels
  - CV and test sets: normal+abnormal labelled data
  - Evaluation metrics like in binary classification (TP rate, precision, AUC score,…)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Recommender Systems

- Recommender systems are systems that provide suggestions for items that are most relevant to a particular user
- Matrix completion problem:
  - sparse matrix with lots of missing values
  - predict missing values from the others
- Example: predicting movie ratings (0-5)
  - Rows: movies
  - Columns: users
  - Use different linear regression model for each user (e.g., if 1M users, we have 1M models)
  - When we have the features for each movie we can formulate a cost function for learning the parameters of all users using gradient descent
    - Supervised regression problem using the squared error loss

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Recommender Systems

- Collaborative filtering:
  - Recommend items based on ratings of users who gave similar ratings
  - Unsupervised because it does not assume knowledge of features
  - Formulates a cost function that can be used to simultaneously learn both the features and the parameters for each user using gradient descent
  - For binary labels we can use a logistic regression prediction model and a binary cross-entropy loss
  - When a new user arrives we can use mean normalization so that the predicted ratings of the new user will equal the mean of all ratings for each movie
- Content-based filtering:
  - Recommendation based on features of user and item to find good match
  - Compute embeddings (e.g., using a NN) from features which need to be of the same size
  - Predicted rating: dot product of embeddings
- Finding related items (e.g., movies related to movie i) can be done using a k-nearest neighbor search (in feature or embedding space)

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to Reinforcement Learning

- Predictions AND actions
- Interaction with environment
- Trial-and-error learning
- Learn to achieve goals
  - Goals defined using reward functions
  - Example: robot learning to escape a room
    - -1 everywhere, 0 at the exit would encourage the robot to find the shortest path
- RL: find a policy that maximizes the expected return (sum of rewards)
- Policy: behavior of the agent
  - mapping from states to actions
  - Deterministic or stochastic
- Value function: how good a state (or state-action pair) is
  - Estimate of the expected return

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Introduction to Reinforcement Learning

- Model: what the environment will do next
  - Which state will I end up if I am in state s and execute action a
  - What reward I will receive?
  - Deterministic or stochastic
- Environment: fully-observable, partially-observable
- Agent categories:
  - Value-based, Policy-based, Actor-Critic
  - Model-free, Model-based
- Prediction: learn a value function for a given policy
- Control: find the best policy
- Learning: unknown environment, interaction of agent with external environment
- Planning: when we have (or learned) a model, “interaction” of agent with the model

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Markov Decision Processes and Dynamic Programming

- MDP: (S,A,T,r)
  - S: set of states
  - A: set of actions
  - T: transition probabilities
  - r: reward function
  - T and r define the model of the environment
  - Markov means that the probability of transitioning to s_{t+1} is only affected by s_t and a_t (not the history)
- Return:
  - Undiscounted: used in episodic tasks (when there is an end)
  - Discounted: used in episodic or continuing tasks
  - Average: used typically in continuing tasks
- Discounted return: how much some future reward is worth to us right now
  - $\gamma \approx 0 \rightarrow$ myopic agent
  - $\gamma \approx 1 \rightarrow$ farsighted agent

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Markov Decision Processes and Dynamic Programming

- Value function: expected return of a given policy (for every state or state-action pair)
- Optimal value function: best value function over all policies
  - Best possible performance in an MDP
- Optimal policy: action that maximizes the optimal value function for a given state
- Bellman equations can be used to find:
  - the value functions of a given policy
  - the optimal value functions
- Dynamic programming: used to find the optimal value function
  - Policy Evaluation: find the value function of some policy (multiple iterations)
  - Policy Improvement: given some value function, take the greedy action at every state
  - Policy Iteration:
    - Start with random policy
    - Repeat: Run policy evaluation until convergence, then policy improvement step
  - Value Iteration:
    - Start with random or zero value function
    - Repeat: Run single step of policy evaluation, then policy improvement step

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model-free Reinforcement Learning

- When we know the model (T,r) we use planning to find the optimal policy
- When we don’t know the model, we use sampling
- Exploitation: go to areas that you have been before that are rewarding
- Exploration: sample new experiences
- Exploration vs Exploitation tradeoff: how to choose between these two?
- Multi-armed bandits setting: only actions, no states
  - Goal: find the action that results in the highest expected return
- ε-greedy action selection:
  - Select random action with probability ε (exploration)
  - Select the greedy action with probability 1-ε (exploitation)
- Model-free RL:
  - Monte Carlo Learning
  - Temporal Difference Learning

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model-free Reinforcement Learning

- Monte Carlo algorithms
  - Only work in episodic tasks: update after the end of the episode
  - Updates can be noisy
- Temporal Difference learning algorithms
  - Work in episodic and continuing tasks: learn after every step
  - Use bootstrapping like in dynamic programming:
    - Update the value estimate of a state (or state-action pair) based on another estimate (of the value of the next state or state-action pair)
    - Single-step value estimates can be inaccurate
      - We can use multi-step TD to mitigate this
- TD for control:
  - use state-action (Q) value functions
  - On-policy: learn about some behavior policy while following that policy
  - Off-policy: follow some policy, but learn about a different policy

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Model-free Reinforcement Learning

- SARSA is on-policy:
  - $Q(s_t, a_t) = Q(s_t, a_t) + \eta\big(R_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\big)$
  - Converges to the optimal action-value function if the policy is greedy in the limit of infinite exploration (e.g., ε-greedy starting with high epsilon and decreasing it)
- Q-learning is off-policy:
  - $Q(s_t, a_t) = Q(s_t, a_t) + \eta\left(R_{t+1} + \gamma \max_b Q(s_{t+1}, b) - Q(s_t, a_t)\right)$
  - Learns about the greedy policy while following some other policy (e.g., random)
  - Converges to the optimal action-value function if all state-action pairs are visited infinitely often
- Initializing the action-value function optimistically and acting greedily allows the agent to explore all state-action pairs.
  - Optimistic initialization:
    - maximum possible expected return from each state-action pair
    - $Q_{init} = r_{max}/1 - \gamma$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Looking forward

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

ML for Natural Language Processing

- How do we process and analyze natural language data?

- Tasks:
  - speech recognition
  - text2speech
  - dialogue generation
  - automatic summarization
  - machine translation
  - sentiment analysis
  - natural language understanding
  - natural language generation
  - text2image generation

- Models: BERT, GPT3, PaLM,...

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

ML for Computer Vision

- How do we process images, video, point clouds?

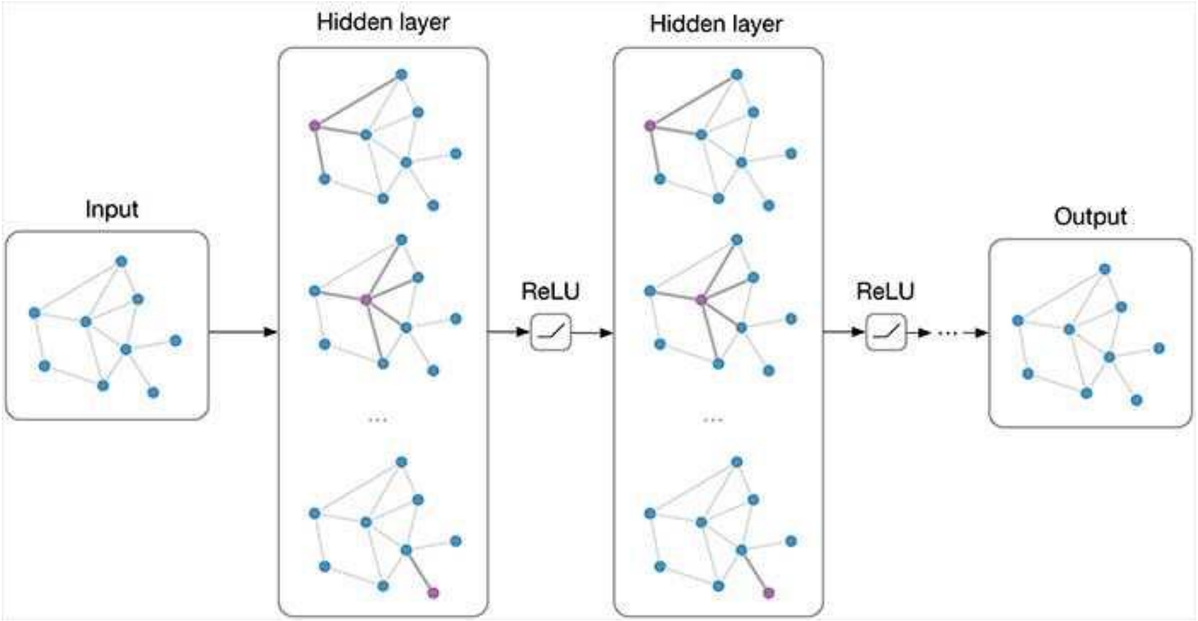
- Tasks:
  - object recognition
  - object segmentation
  - object tracking
  - pose estimation
  - activity recognition
  - scene reconstruction
  - image captioning
  - face recognition
  - style transfer
  - text2image generation

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

ML for graph data

- Graph data:
  - social networks
  - molecules
  - images as graphs
  - text as graphs

- Graph neural networks



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Generative Models

Can generate images, audio, music, text, …

Models:

- Variational Autoencoders
- Generative Adversarial Networks
- Flows
- Diffusion models

Go to:
https://this-person-does-not-exist.com/

SYSTEM PROMPT (HUMAN-WRITTEN)	*In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.*
MODEL COMPLETION (MACHINE-WRITTEN, 10 TRIES)	The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science. Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved. Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Meta-learning

- **Learning to learn fast**

- Learn from a distribution of tasks how to adapt quickly to a new task (drawn from the same distribution)
- Various methods:
  - learning an optimizer (instead of using backprop)
  - learning good initial values for NNs coupled with gradient based optimization

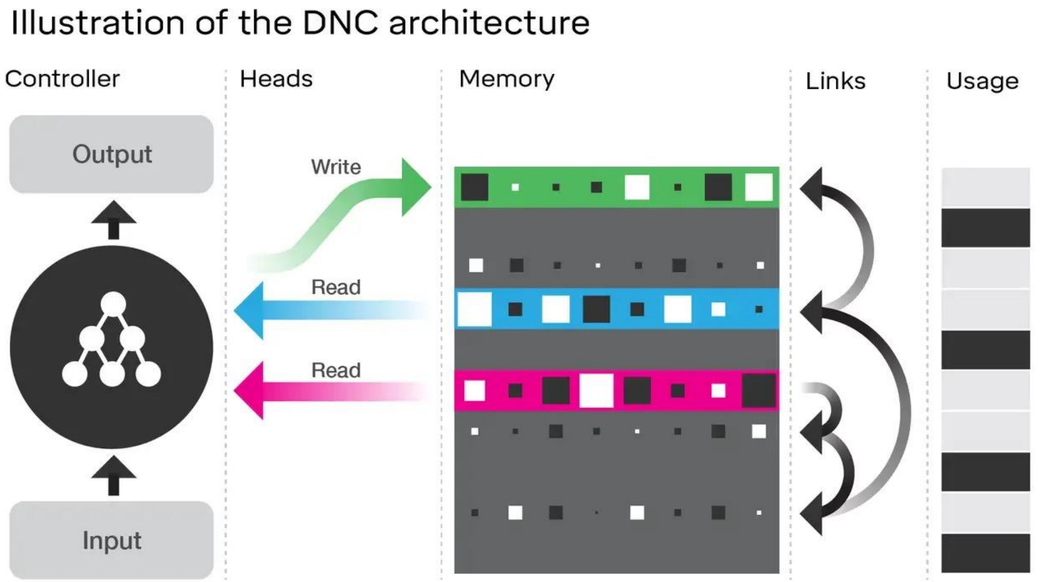
- Can help generalization
- Can help with data efficiency

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Memory-augmented NNs

- **Decoupling memory from computation**
  - CPU + RAM
  - Learn how to read/write from/to memory, and modify memory

- Examples:
  - Neural Turing Machines
  - Memory Networks
  - Differentiable Neural Computers



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Self-supervised Learning

- No labelled data
- Extract supervisory signals from the data and use supervised techniques to learn representations
  - e.g. predicting parts of images from other parts
- Often used with data augmentation
- Promising approach to learn better representations

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

(More on) Reinforcement Learning

- Deep Reinforcement Learning
  - RL : learning to act
  - DL : learning good representations
- High dimensional input and output spaces
  - e.g., self-driving car, humanoid robot
- How do we learn in the absence of external rewards?
  - e.g., curiosity, intrinsic motivation
- How do we structure the learning process to facilitate learning?
  - automatic curriculum, open-ended learning
- How do we combine learning and planning?
  - learning a model of the world
- ...

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Multiagent Learning

- MDP framework assumes a single agent trying to maximize reward
- The world consists of multiple agents interacting and learning
- Competition: e.g., creating superhuman AI for playing games
- Coordination: how do we create agents that coordinate their behavior to solve a complex problem (e.g. push a block towards a certain position)
- Mixture of competition/coordination: e.g., robot soccer, most of real-world interactions
- Communication: e.g., for resolving conflicts, avoiding deadlocks
- Agents teaching other agents:
  - agent societies
  - how human intelligence arose

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Explainable ML

- ML models are often viewed as black boxes
  - know the inputs and outputs but not the internals
  - especially the case of NNs
- Often models learn the wrong association, which however results in good performance
  - e.g., learning to recognize cows
- How do we create systems that explain their predictions or decisions?
  - e.g., through textual descriptions or visualizations
- Enable counterfactural reasoning/explanations

Class: White Necked Raven

Counter-Class: American Crow

This is a ***White Necked Raven*** because this is a black bird with a white nape and a large beak. This is not an ***American Crow*** because it does not have a pointy black beak.

source

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Continual Learning

- Transfer learning: train on task T1, adapt/fine-tune on task T2
  - we take a model trained on task 1, and fine-tune it with data from task 2, where tasks 1 and 2 are expected to be similar
  - doing so is expect to learn task 2 without requiring a lot of data
  - we don't care about solving T1 with the new model
- Multi-task learning: train a model simultaneously on tasks {T1,T2,...,Tn}
- Meta-learning: train on tasks {T1,T2,..,Tn}, adapt on task Tn+1
- Sequential multi-task learning: train on task 1, then train on task 2, then train on task 3,...
  - naive transfer learning will result in catastrophic forgetting, i.e., knowledge about previous tasks will get overwritten

- Continual learning: how to tackle the problem of catastrophic forgetting when learning sequentially
  - how does the brain do it?
  - one of the grand AI challenges!

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Thank you

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423