MAI4CAREU Master programmes in Artificial Intelligence 4 Careers in Europe

Русенски университет

ИНТЕЛИГЕНТНИ КОМПЮТЪРНИ СИСТЕМИ

Светлана Стефанова

Септември, 2022

Co-financed by the European Union
Connecting Europe Facility

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

ЛЕКЦИЯ 11

ПЛИТКИ НЕВРОННИ МРЕЖИ

СЪДЪРЖАНИЕ

1. Въведение
2. Скрит слой
3. Математическо моделиране
4. Дефиниране на теглата

Co-financed by the European Union
Connecting Europe Facility

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

СЪДЪРЖАНИЕ 1

Определение

Под НМ си представяме структура, състояща се от много слоеве, включително скрити. Но има и тип НМ само с няколко броя скрити слоеве.

Плитките невронни мрежи се състоят само от 1 скрит слой. Разбирането им дава представа какво точно се случва в дълбока невронна мрежа.

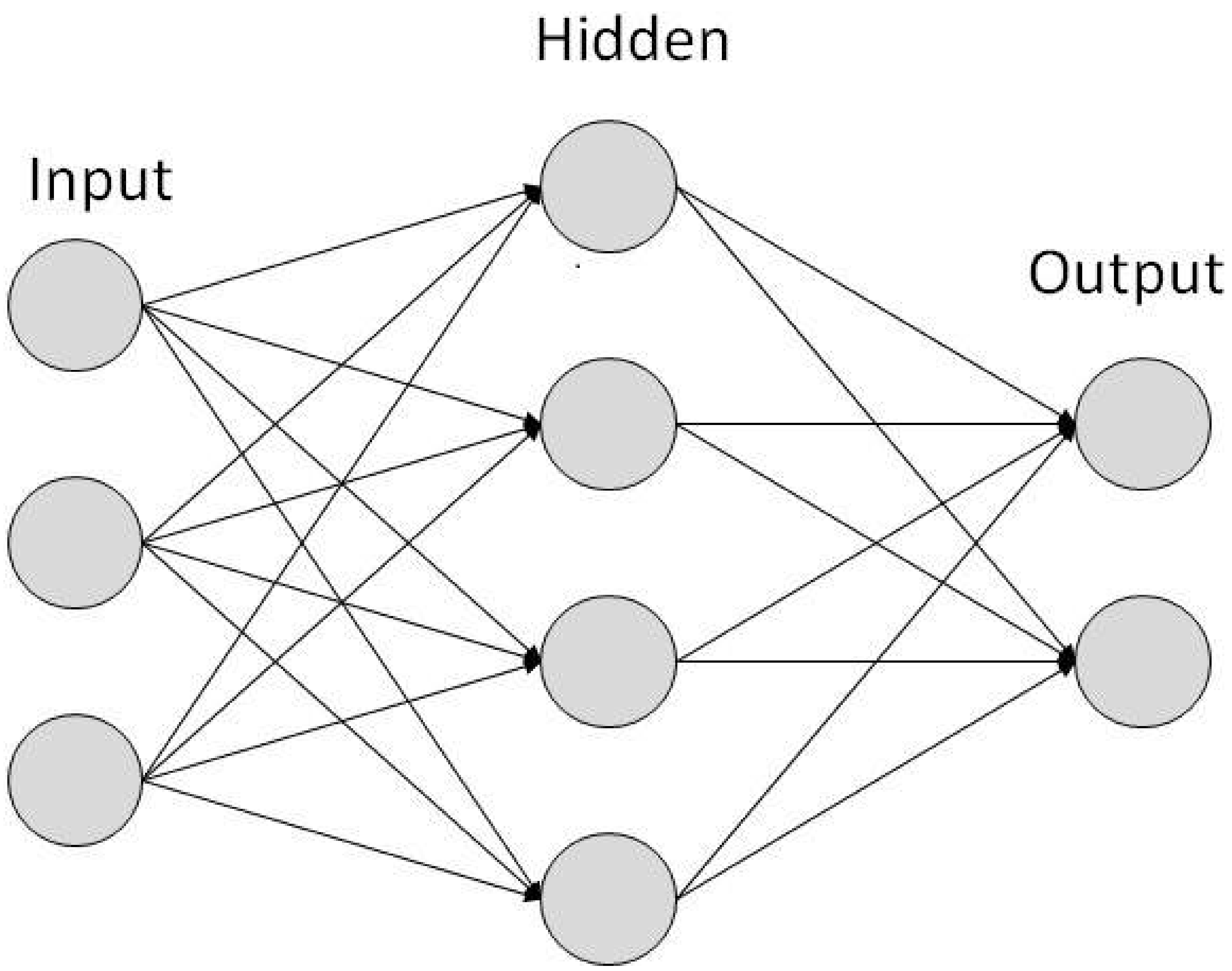
Co-financed by the European Union
Connecting Europe Facility

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

СЪДЪРЖАНИЕ 1

Пример

Плитка невронна мрежа с 1 скрит слой, 1 входен слой и 1 изходен слой.



Co-financed by the European Union
Connecting Europe Facility

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

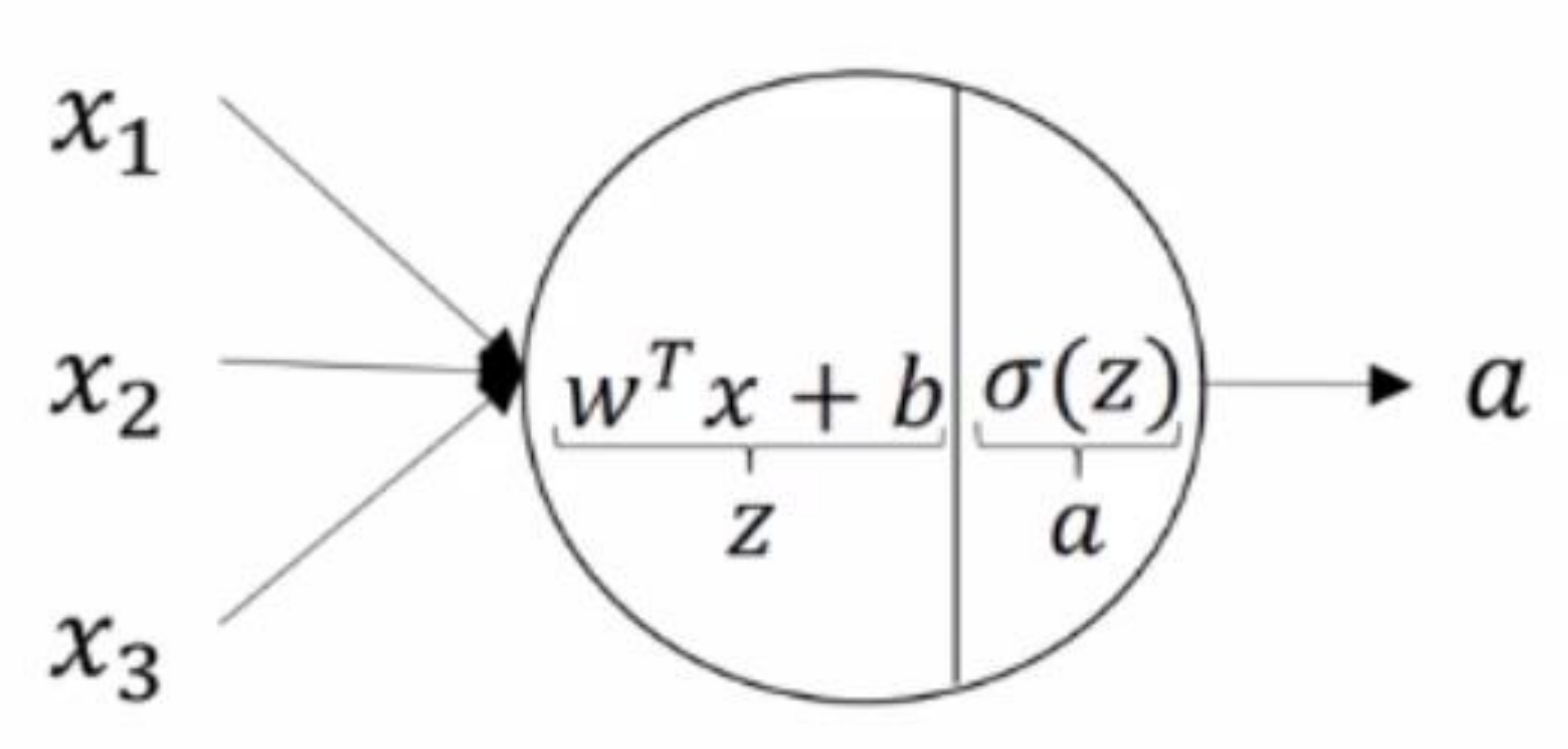
СЪДЪРЖАНИЕ 1

Модел на неврон

Невронът е градивната единица на НМ. При зададен вход, той осигурява изхода и предава този изход като вход към следващия слой. Невронът може да се разглежда като комбинация от 2 части:

Първата част - изчислява изхода **z**, като използва входовете и теглата им.

Втората част - извършва активирането на **z**, за да даде крайния изход **a** на неврона.

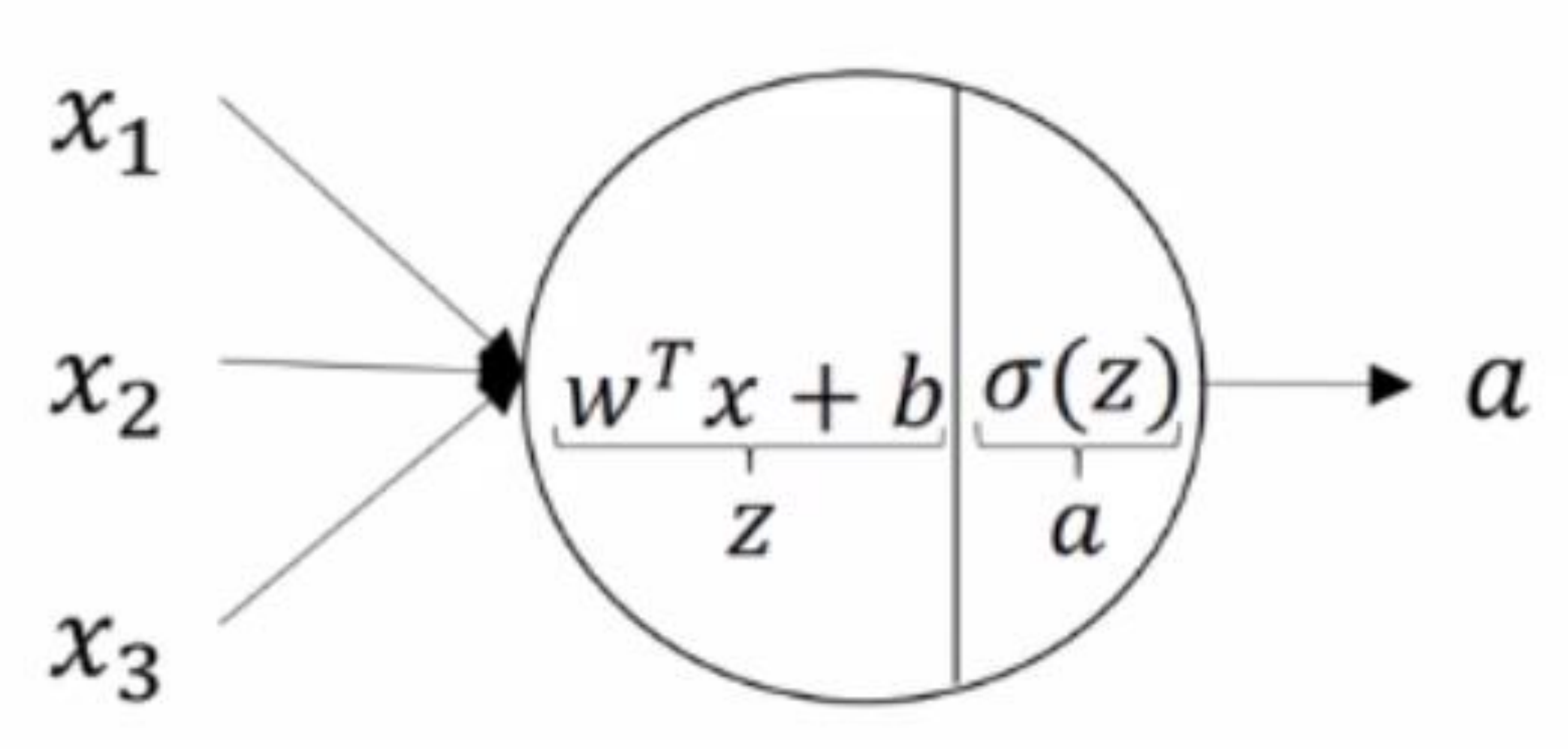


Скритият слой

Скритият слой се състои от различни неврони, всеки от които извършва 2-те последователни изчисления.

Нека имаме 4 неврона в скрития слой на нашата плитка НМ, изчисляващи следното:

$$z_1^{[1]} = w_1^{[1]T} x + b_1^{[1]}, a_1^{[1]} = \sigma\left(z_1^{[1]}\right)$$
$$z_2^{[1]} = w_2^{[1]T} x + b_2^{[1]}, a_2^{[1]} = \sigma\left(z_2^{[1]}\right)$$
$$z_3^{[1]} = w_3^{[1]T} x + b_3^{[1]}, a_3^{[1]} = \sigma\left(z_3^{[1]}\right)$$
$$z_4^{[1]} = w_4^{[1]T} x + b_4^{[1]}, a_4^{[1]} = \sigma\left(z_4^{[1]}\right)$$



Co-financed by the European Union
Connecting Europe Facility

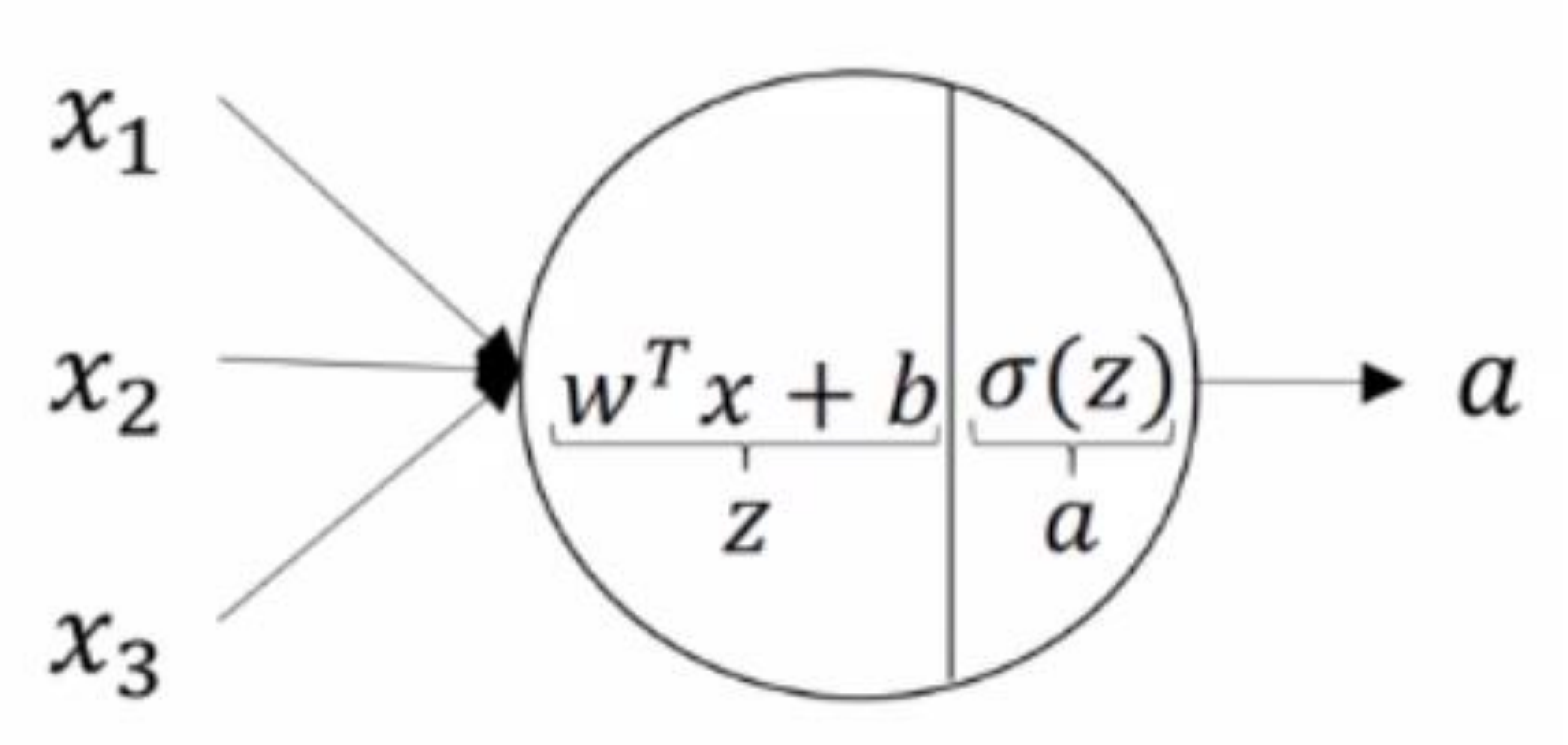
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

СЪДЪРЖАНИЕ 2

Скритият слой

В горните уравнения,

- горният индекс $^{[i]}$ обозначава номера на слоя, а индексният номер $_j$ означава невронния номер в определен слой.
- ***X*** е входният вектор, състоящ се от 3 характеристики.
- $W^{[i]}_j$ е теглото, свързано с неврон ***j***, присъстващ в слоя ***i***.
- $b^{[i]}_j$ е отклонението, свързано с неврон ***j***, присъстващ в слоя ***i.***

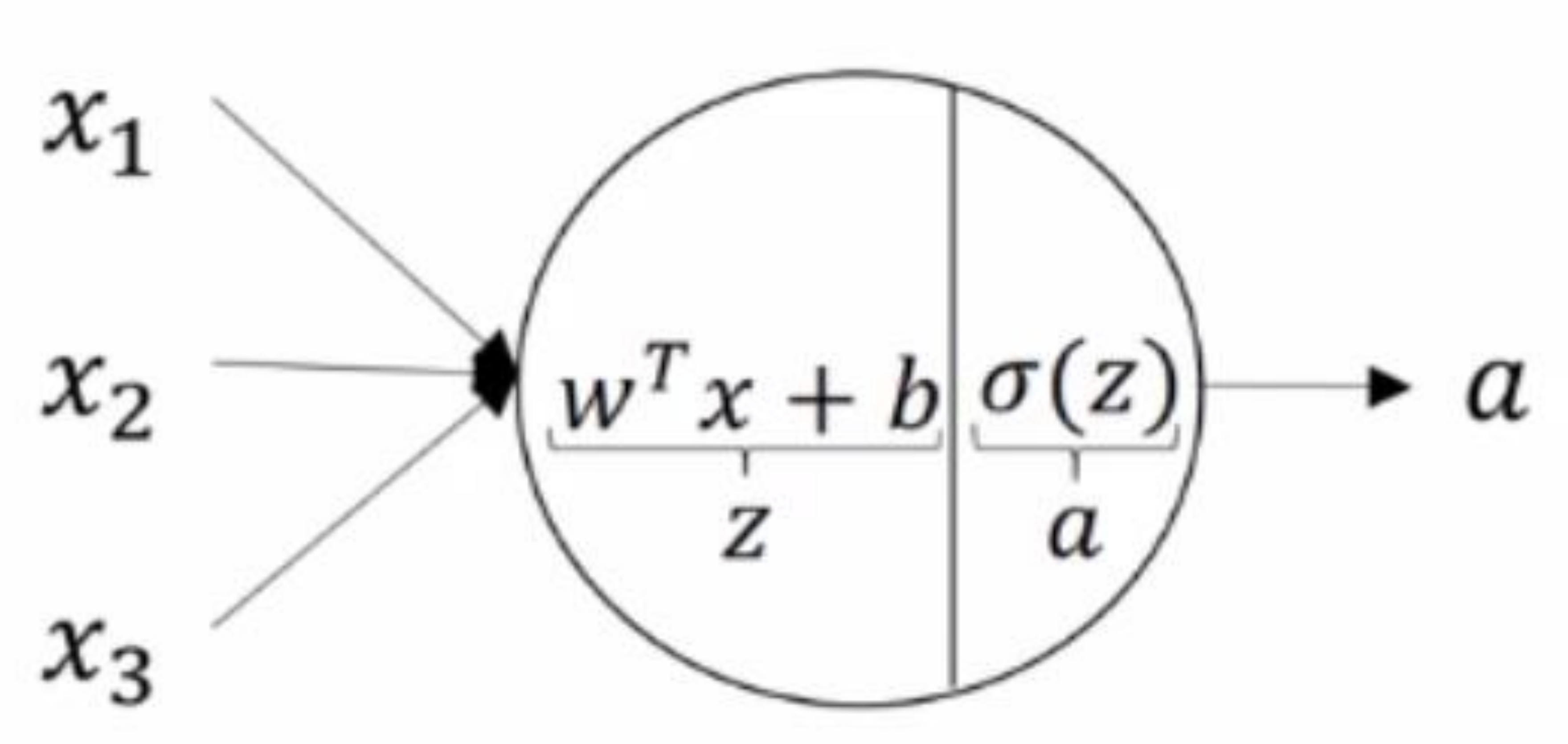


СЪДЪРЖАНИЕ 2

Скритият слой

- $Z^{[i]}_j$ е междинният изход, свързан с неврон j, присъстващ в слоя i.
- $a^{[i]}_j$ е крайният изход, свързан с неврон j, присъстващ в слоя i.
- σ е функцията за активиране (напр. сигмоидна). Математически се дефинира:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



Co-financed by the European Union
Connecting Europe Facility

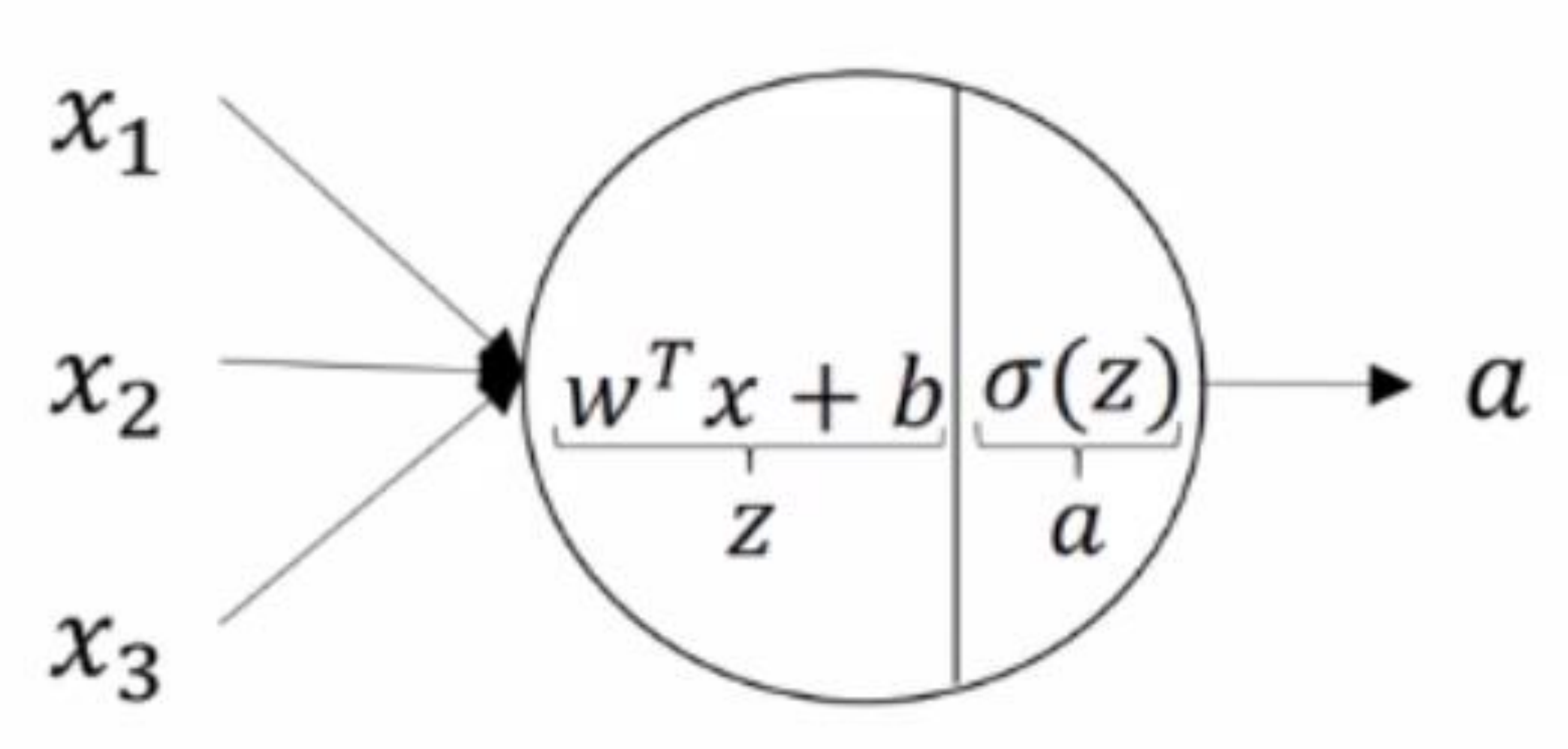
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Скритият слой

Можем да векторизираме уравненията:

$$Z^{[1]} = W^{[1]T} X + b^{[1]}$$

$$A^{[1]} = \sigma\left(Z^{[1]}\right)$$



Първото уравнение изчислява всички междинни изходи ***Z*** при единично матрично умножение.

Второто уравнение изчислява всички активирания ***A*** при единично матрично умножение.

Уравнения за разпространение напред

- 1-то уравнение изчислява междинния изход $Z^{[1]}$ на първия скрит слой.
- 2-то уравнение изчислява крайния изход $A^{[1]}$ на първия скрит слой.
- 3-то уравнение изчислява междинния изход $Z^{[2]}$ на изходния слой.
- 4-то уравнение изчислява крайния изход $A^{[2]}$ на изходния слой, който е и крайният изход на цялата невронна мрежа.

$$Z^{[1]} = W^{[1]T} X + b^{[1]}$$

$$A^{[1]} = \sigma\left(Z^{[1]}\right)$$

$$Z^{[2]} = W^{[2]T} A^{[1]} + b^{[2]}$$

$$\hat{y} = A^{[2]} = \sigma\left(Z^{[2]}\right)$$

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

СЪДЪРЖАНИЕ 3

Активационна функция

НМ е основно набор от математически уравнения и тегла. За да я направим стабилна, така че да се представя добре в различни сценарии, използваме функциите за активиране. Те въвеждат нелинейни свойства в НМ.

Нека опитаме да разберем защо функциите за активиране са от решаващо значение за всяка НМ с помощта на плитката невронна мрежа.

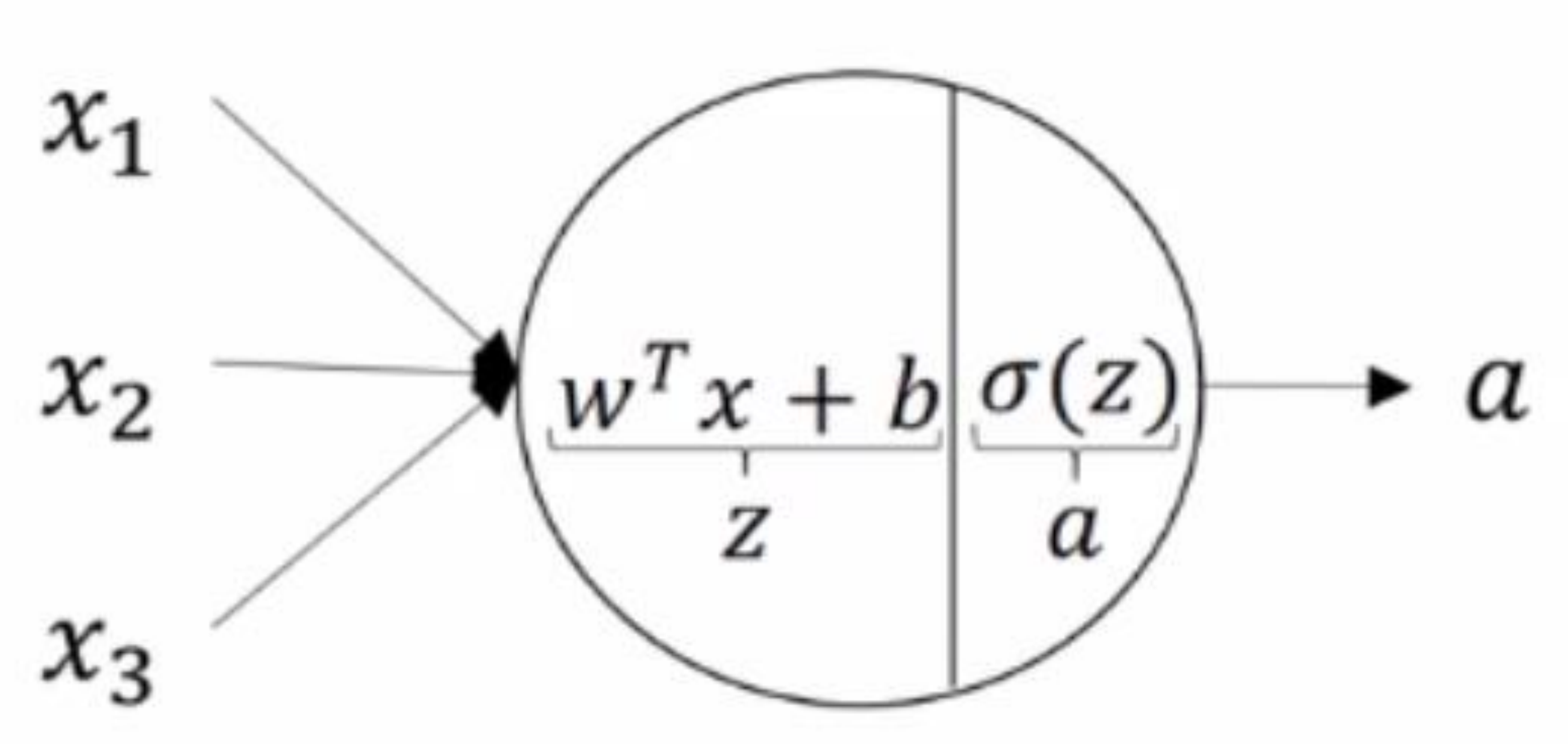
Co-financed by the European Union
Connecting Europe Facility

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

СЪДЪРЖАНИЕ 3

Активационна функция

Без функциите за активиране нашата плитка невронна мрежа може да се представи:



$$Z^{[1]} = W^{[1]T} X + b^{[1]}$$

$$\hat{y} = Z^{[2]} = W^{[2]T} Z^{[1]} + b^{[2]}$$

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

СЪДЪРЖАНИЕ 3

Активационна функция

Ако заменим стойността на $Z^{[1]}$ от уравнение 1 в уравнение 2, получаваме следното:

$$Z^{[1]} = W^{[1]T}X + b^{[1]}$$
$$\hat{y} = Z^{[2]} = W^{[2]T}W^{[1]T}X + W^{[2]T}b^{[1]} + b^{[2]}$$
$$\hat{y} = Z^{[2]} = W_{new}X + b_{new}$$

СЪДЪРЖАНИЕ 3

Активационна функция

Както виждаме, изходът ще се превърне в линейна комбинация от нова матрица за тежести ***W***, вход ***X*** и ново отклонение ***b***, което означава, че невроните, присъстващи в скрития слой, и теглата им остават без значение.

Следователно:

- за да въведем нелинейност в мрежата, трябва да използваме функциите за активиране;
- не е задължително да се използва една функция за активиране за всички слоеве.

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

СЪДЪРЖАНИЕ 4

Инициализиране на теглата

Матрицата на теглата ***W*** на НМ се инициализира **на случаен принцип**.

Защо не може да се инициализира с 0 или някаква конкретна стойност? - да разберем това с помощта на плитката невронна мрежа.

Инициализиране на теглата

Нека

W^1 - матрицата на теглата на слой 1 и

W^2 - матрицата на теглата на слой 2

се инициализират с 0 или друга стойност.

Ако матриците на теглата са еднакви - активирането на невроните в скрития слой ще бъде същото. Освен това производните на активациите биха били еднакви. Следователно невроните в скрития слой ще модифицират теглата по подобен начин, т.е. няма да има значение дали има повече от 1 неврон в един скрит слой.

Ние не искаме това - искаме всеки неврон в скрития слой да бъде уникален, да има различно тегло и да работи като уникална функция. Затова инициализираме теглата на случаен принцип.

Метод на Хавиер

Най-добрият метод за инициализация е методът на Хавиер.
Математически се дефинира като:

$$W^{[l]} \sim \mathcal{N}\left(\mu = 0, \sigma^2 = \frac{1}{n^{[l-1]}}\right)$$

$$b^{[l]} = 0$$

Той гласи, че матрицата на теглото ***W*** на определен слой ***l*** се избира произволно от нормално разпределение със средната стойност ***μ = 0*** и дисперсия ***σ ² = мултипликативната обратна на броя на невроните в слой l − 1***.

Отклонението ***b*** на всички слоеве се инициализира с **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

Градиентно спускане

Вече знаем, че теглата на невронната мрежа се инициализират произволно.

За да използваме невронната мрежа за правилни прогнози, трябва да актуализираме тези тегла.

Методът, чрез който актуализираме теглата, е известен като **градиентно спускане**.

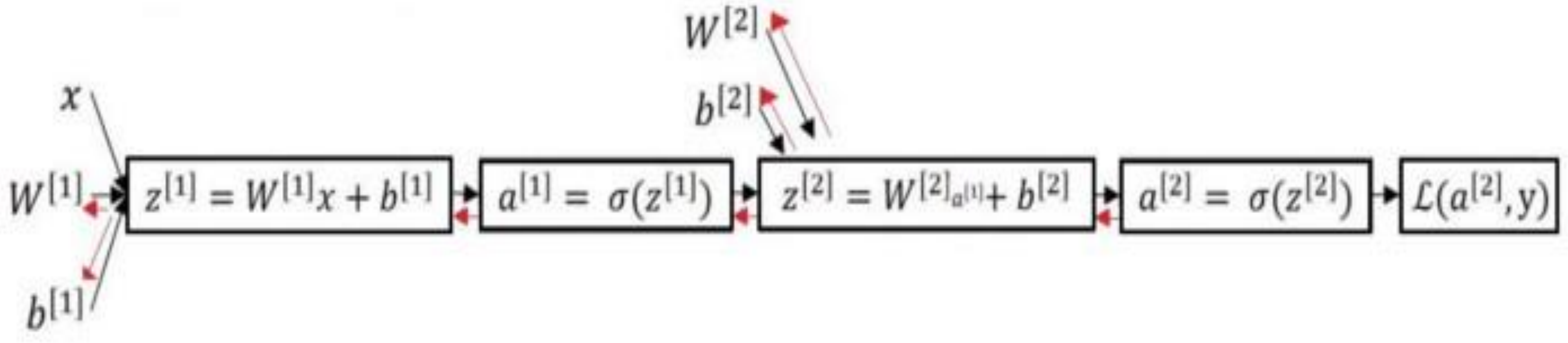
Co-financed by the European Union
Connecting Europe Facility

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

СЪДЪРЖАНИЕ 4

Градиентно спускане

- **Разпространение напред** (**черните** линии) - използва се за изчисляване на изхода за даден вход ***X***.
- **Обратното разпространение** (червените линии) използва се за актуализиране на тегловните матрици $W^{[1]}$, $W^{[2]}$ и отклонения $b^{[1]}$, $b^{[2]}$. Прави се чрез изчисляване на производни на входовете на всяка стъпка.



Co-financed by the European Union
Connecting Europe Facility

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

СЪДЪРЖАНИЕ 4

Грешка в резултата

Грешката ***L*** се определя математически:

$$L(\hat{y}, y) = -\left[y \log \hat{y} + (1-y) \log (1-\hat{y})\right]$$

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

Обратното разпространение

Използвайки уравнението за грешка ***L*** и сигмоидна функция за активиране на скрития и изходния слой, с помощта на верижно правило на производни, изчисляваме:

$$dA^{[2]} = \frac{\delta L\left(A^{[2]},Y\right)}{\delta A^{[2]}} = \frac{-Y}{A^{[2]}} + \frac{1-Y}{1-A^{[2]}}$$

$$dZ^{[2]} = \frac{\delta L\left(A^{[2]},y\right)}{\delta Z^{[2]}} = \frac{\delta L\left(A^{[2]},y\right)}{\delta A^{[2]}} * \frac{\delta A^{[2]}}{\delta Z^{[2]}} = A^{[2]} - Y$$

$$dW^{[2]} = \frac{\delta L\left(A^{[2]},y\right)}{\delta W^{[2]}} = \frac{\delta L\left(A^{[2]},y\right)}{\delta Z^{[2]}} * \frac{\delta Z^{[2]}}{\delta W^{[2]}} = dZ^{[2]} A^{[1]T}$$

$$db^{[2]} = \frac{\delta L\left(A^{[2]},y\right)}{\delta b^{[2]}} = \frac{\delta L\left(A^{[2]},y\right)}{\delta Z^{[2]}} * \frac{\delta Z^{[2]}}{\delta b^{[2]}} = dZ^{[2]}$$

$$dA^{[1]} = \frac{\delta L\left(A^{[2]},Y\right)}{\delta A^{[1]}} = \frac{\delta L\left(A^{[2]},Y\right)}{\delta Z^{[2]}} * \frac{\delta Z^{[2]}}{\delta A^{[1]}} = dZ^{[2]} W^{[2]}$$

$$dZ^{[1]} = \frac{\delta L\left(A^{[2]},y\right)}{\delta Z^{[1]}} = \frac{\delta L\left(A^{[2]},y\right)}{\delta A^{[1]}} * \frac{\delta A^{[1]}}{\delta Z^{[1]}} = W^{[2]T} dZ^{[2]} * \sigma' \left(Z^{[1]}\right)$$

$$dW^{[1]} = \frac{\delta L\left(A^{[2]},y\right)}{\delta W^{[1]}} = \frac{\delta L\left(A^{[2]},y\right)}{\delta Z^{[1]}} * \frac{\delta Z^{[1]}}{\delta W^{[1]}} = dZ^{[1]} X^{T}$$

$$db^{[1]} = \frac{\delta L\left(A^{[2]},y\right)}{\delta b^{[1]}} = \frac{\delta L\left(A^{[2]},y\right)}{\delta Z^{[1]}} * \frac{\delta Z^{[1]}}{\delta b^{[1]}} = dZ^{[1]}$$

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

MAI4CAREU

Master programmes in Artificial Intelligence 4 Careers in Europe

Благодаря ви.

Co-financed by the European Union
Connecting Europe Facility

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