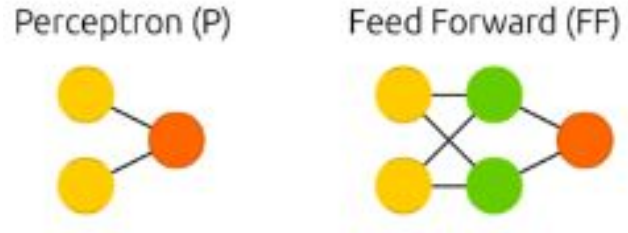
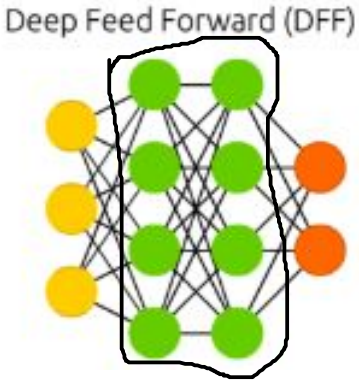
Нейронні мережі. Лекція 2. Навчання нейронних мереж

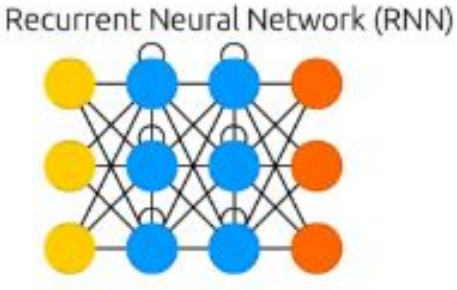
кафедра програмної інженерії ЗНУ

Типи нейромереж

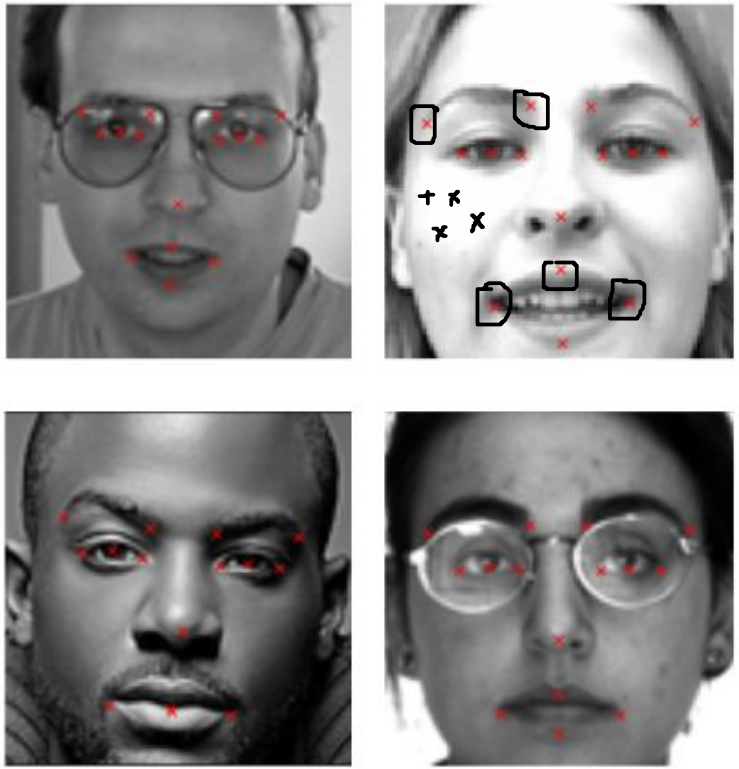
http://www.asimovinstitute.org/neural-network-zoo/







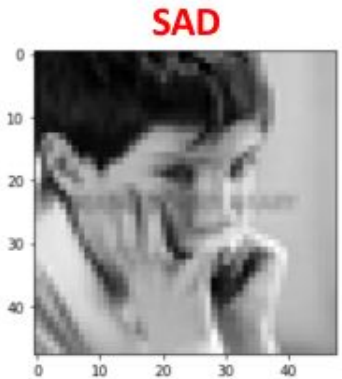
Препроцесінг: визначення ключових точок



Data Source: https://www.kaggle.com/c/facial-keypoints-detection/data

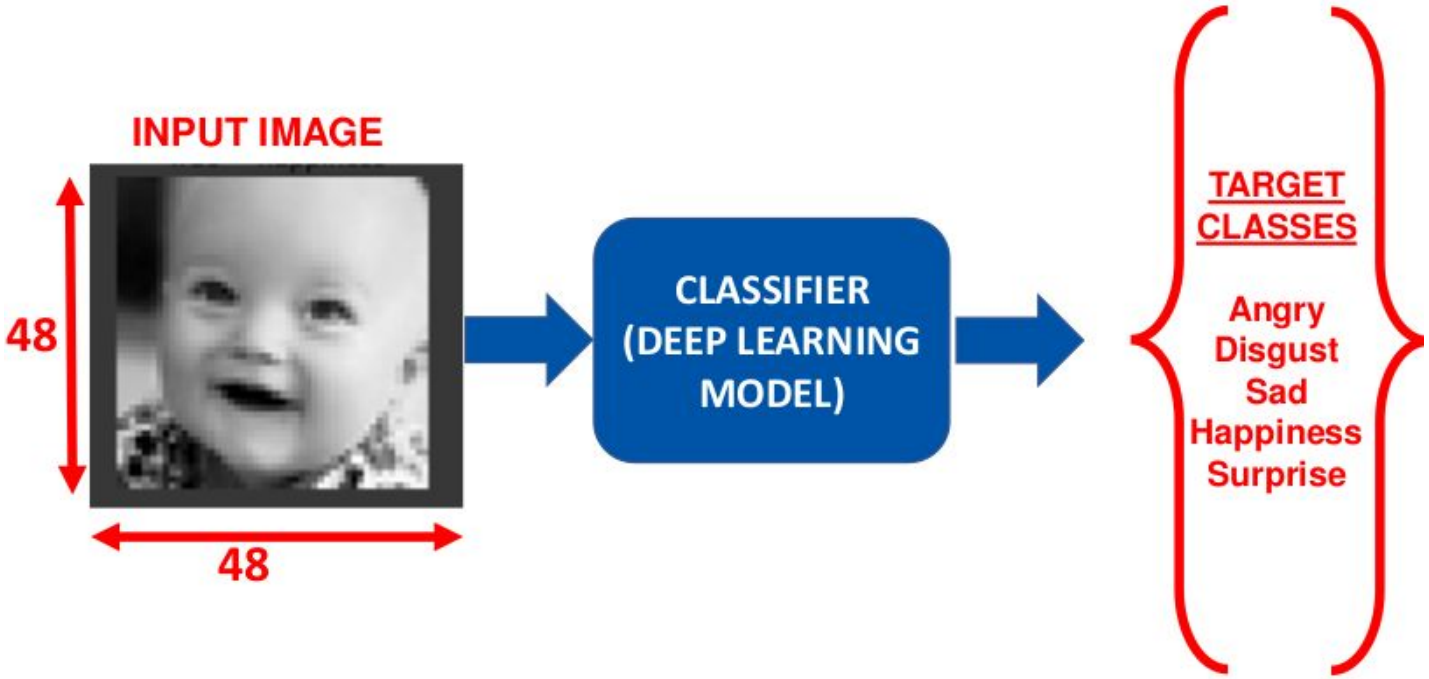
Класифікація зображень

- 0 = Angry
- 1 = Disgust
- 2 = Sad
- 3 = Happy
- 4 = Surprise

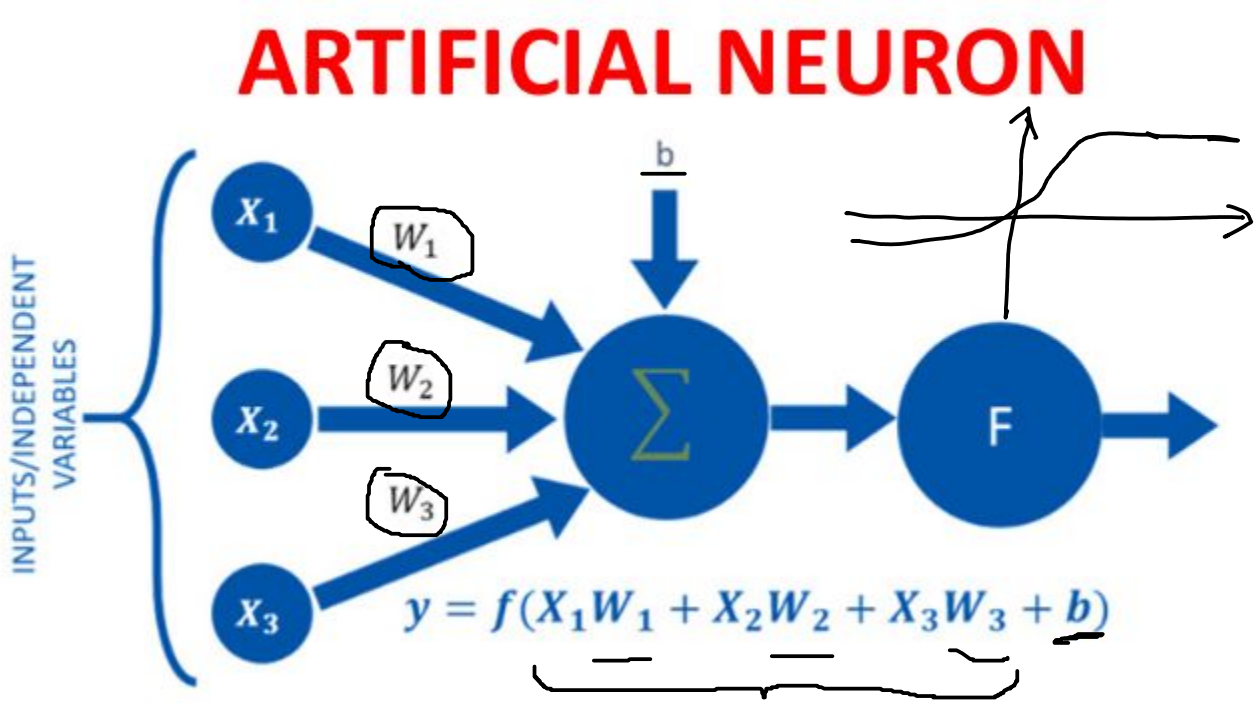


Data Source: [Challenges in Representation Learning: Facial Expression Recognition Challenge | Kaggle]

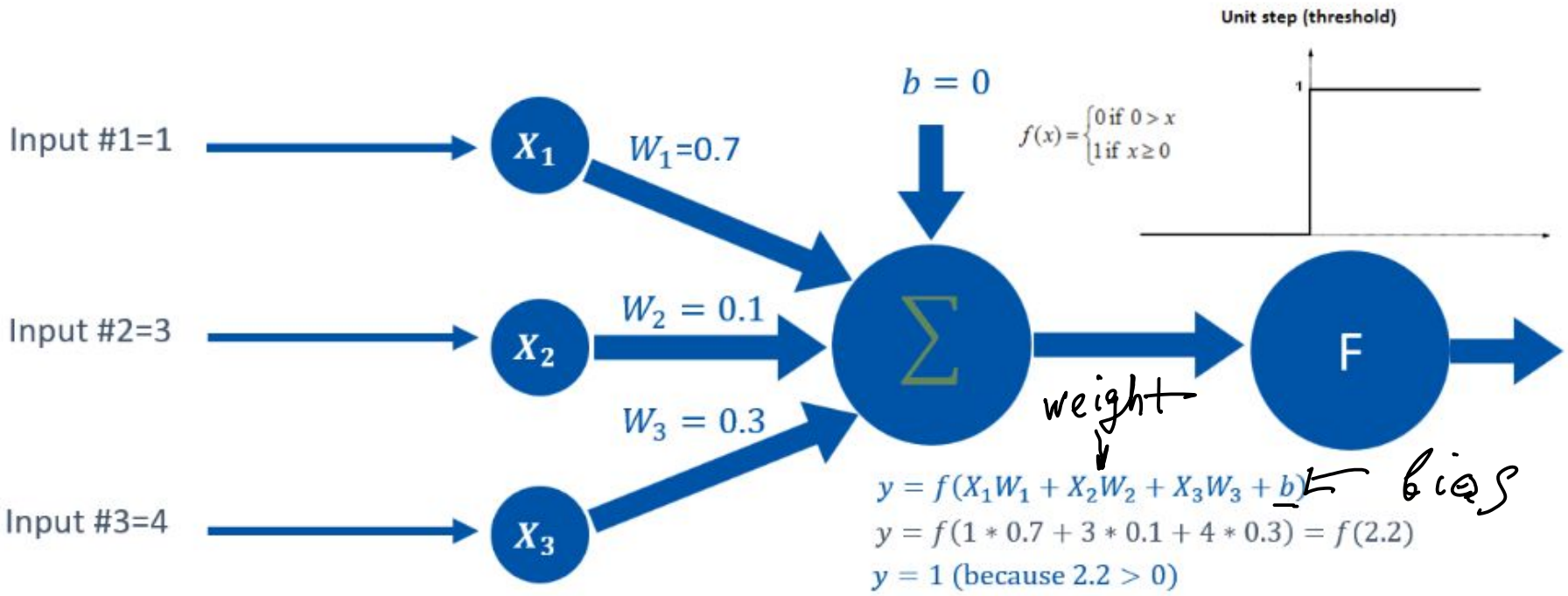
Класифікація зображень



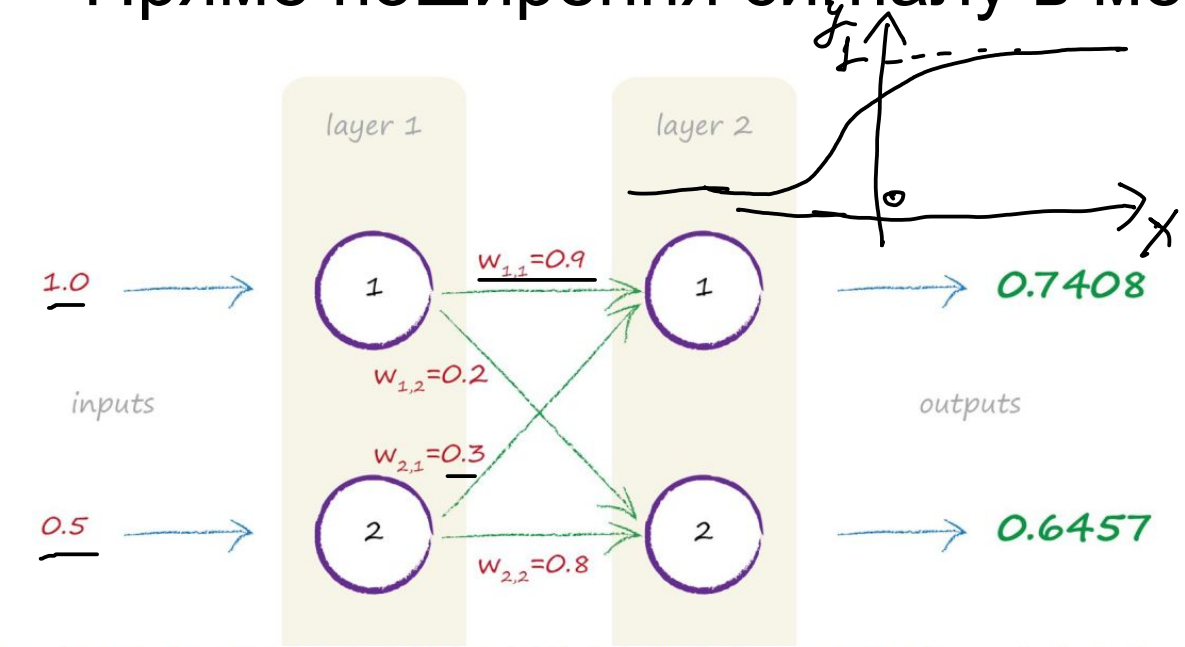
Штучний нейрон

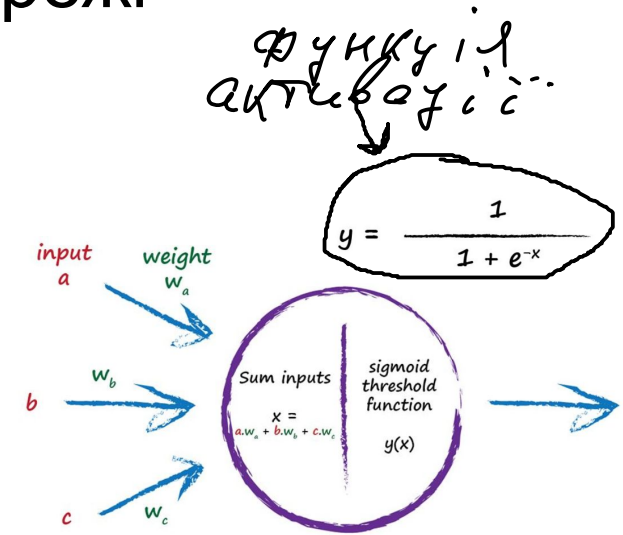


Штучний нейрон



Пряме поширення сигналу в мережі





x = (output from first node * link weight) + (output from second node * link weight)

x = (1.0 * 0.9) + (0.5 * 0.3)

x = 0.9 + 0.15

x = 1.05

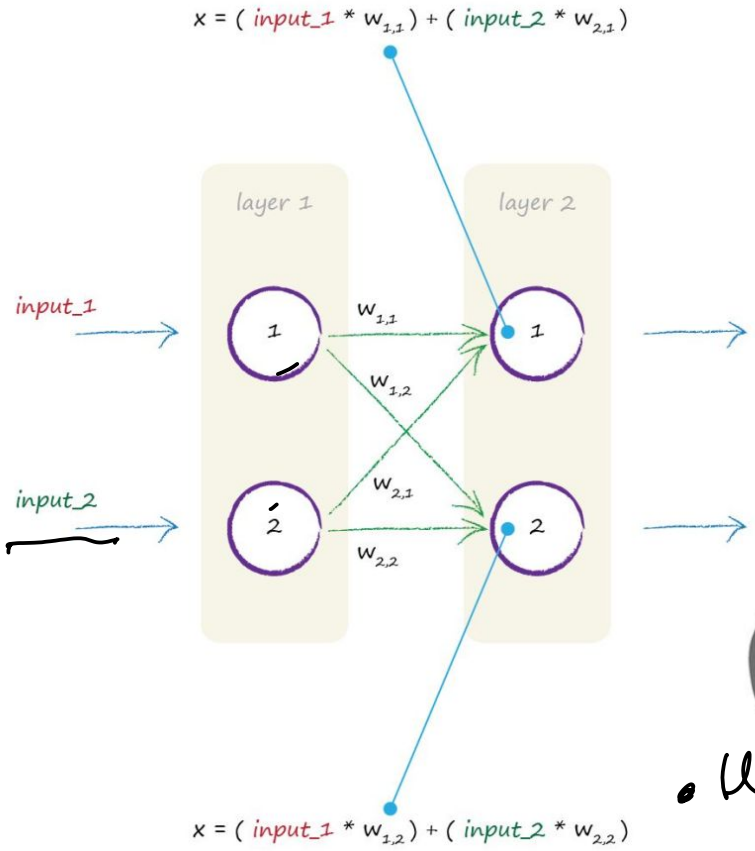
x = (output from first node * link weight) + (output from second node * link weight)

x = (1.0 * 0.2) + (0.5 * 0.8)

x = 0.2 + 0.4

x = 0.6

Пряме поширення сигналу в мережі



$$\begin{pmatrix} a & b & .. \\ c & d & .. \end{pmatrix} \begin{pmatrix} e & f \\ g & h \\ .. & .. \end{pmatrix} = \begin{pmatrix} (a*e)+(b*g)+... & (a*f)+(b*h)+... \\ (c*e)+(d*g)+... & (c*f)+(d*h)+... \end{pmatrix}$$

$$= \begin{pmatrix} ae+bg+... & af+bh+... \\ ce+dg+... & cf+dh+... \end{pmatrix}$$

$O(n^3)$

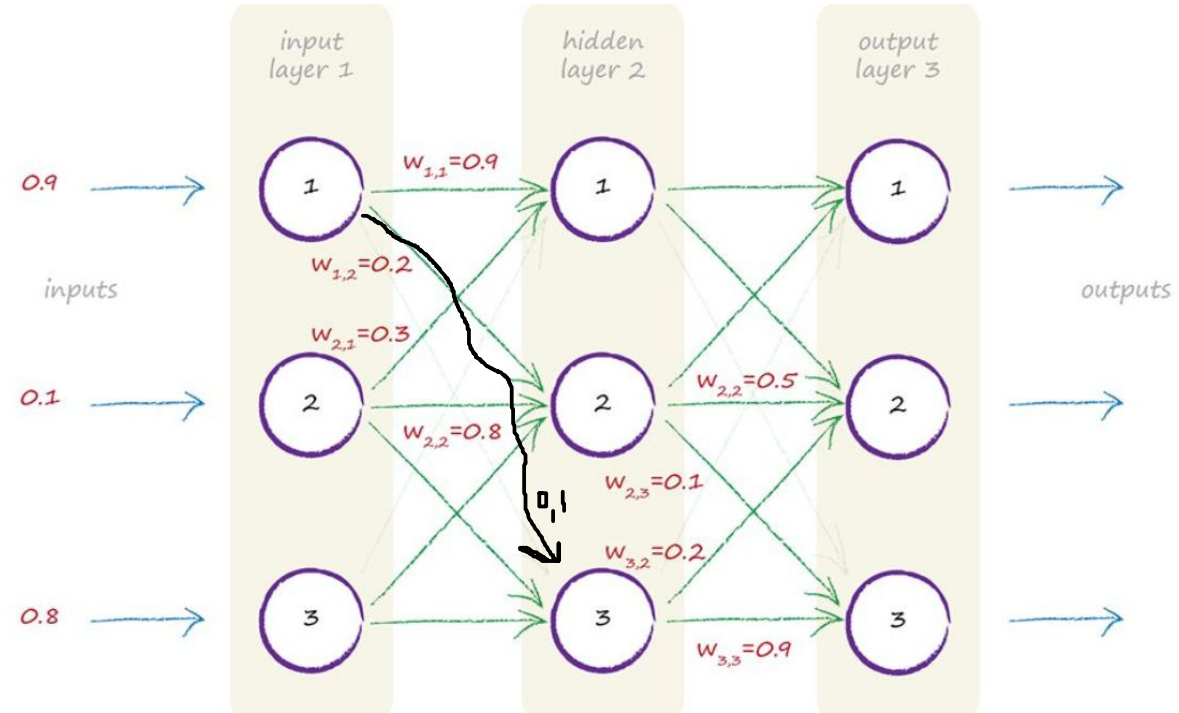
numpy - бібліотека матриць

$W \cdot X = out$

$$\begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix} \begin{pmatrix} input\_1 \\ input\_2 \end{pmatrix} = \begin{pmatrix} (input\_1 * w_{1,1}) + (input\_2 * w_{2,1}) \\ (input\_1 * w_{1,2}) + (input\_2 * w_{2,2}) \end{pmatrix}$$

- Не використовувати for

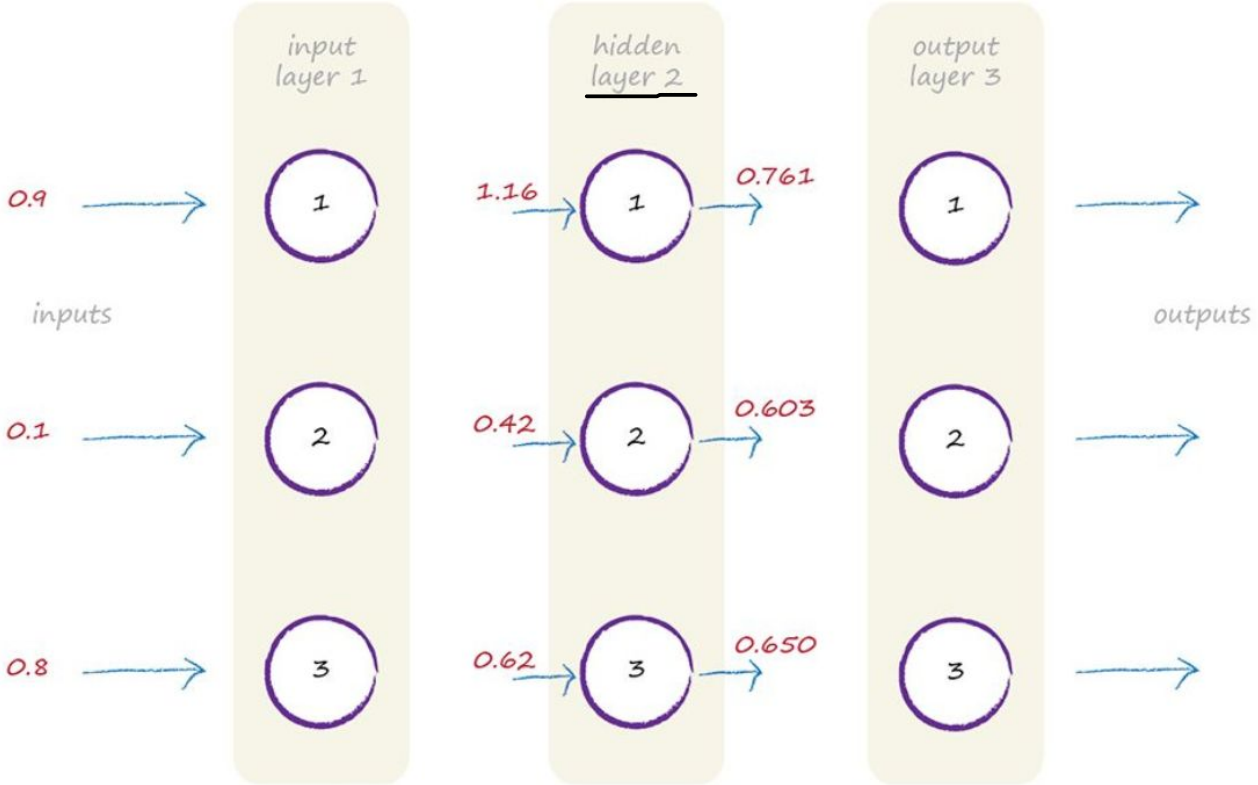
Пряме поширення сигналу в мережі



$$X_{hidden} = \begin{pmatrix} 0.9 & 0.3 & 0.4 \\ 0.2 & 0.8 & 0.2 \\ 0.1 & 0.5 & 0.6 \end{pmatrix} \cdot \begin{pmatrix} 0.9 \\ 0.1 \\ 0.8 \end{pmatrix}$$

$$X_{hidden} = \begin{pmatrix} 1.16 \\ 0.42 \\ 0.62 \end{pmatrix}$$

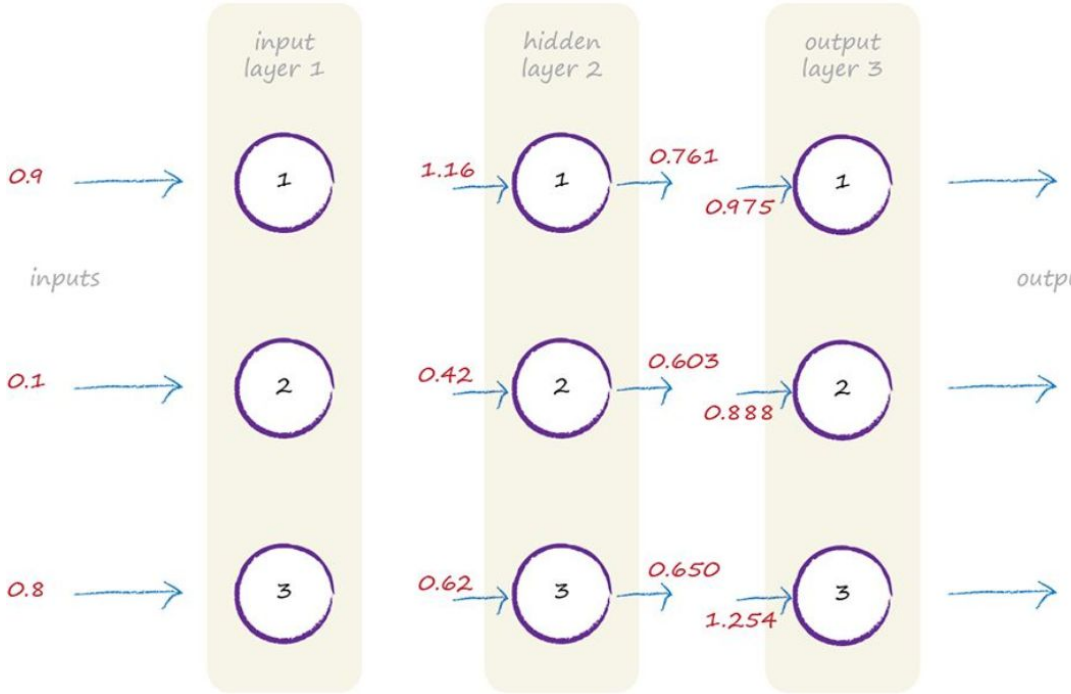
Пряме поширення сигналу в мережі



$$O_{hidden} = sigmoid \begin{pmatrix} 1.16 \\ 0.42 \\ 0.62 \end{pmatrix}$$

$$O_{hidden} = \begin{pmatrix} 0.761 \\ 0.603 \\ 0.650 \end{pmatrix}$$

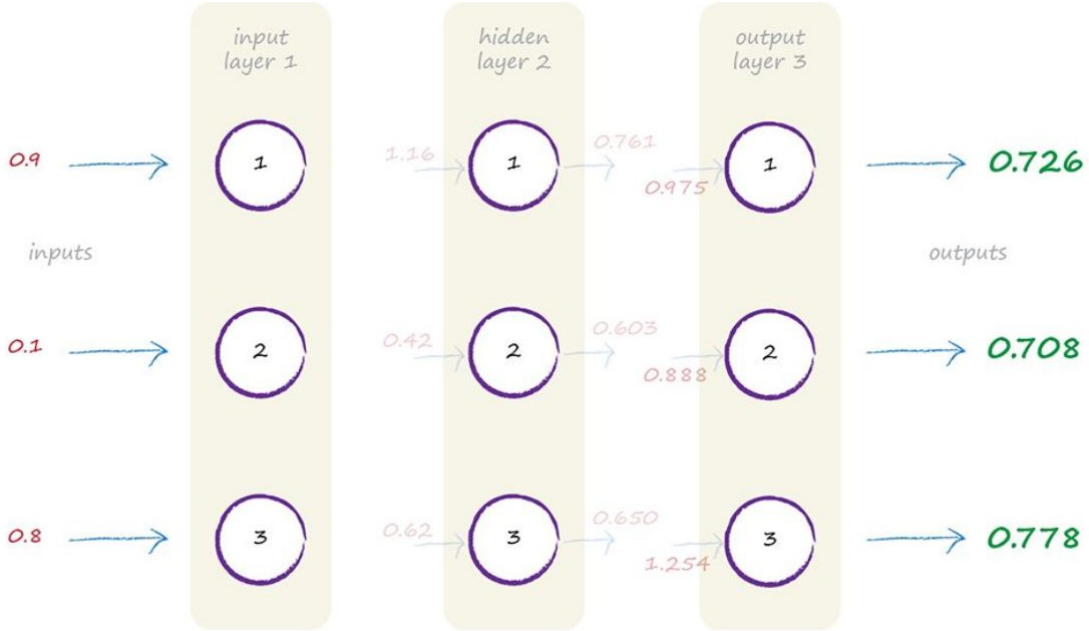
Пряме поширення сигналу в мережі



$$X_{output} = \begin{pmatrix} 0.3 & 0.7 & 0.5 \\ 0.6 & 0.5 & 0.2 \\ 0.8 & 0.1 & 0.9 \end{pmatrix} \cdot \begin{pmatrix} 0.761 \\ 0.603 \\ 0.650 \end{pmatrix}$$

$$X_{output} = \begin{pmatrix} 0.975 \\ 0.888 \\ 1.254 \end{pmatrix}$$

Пряме поширення сигналу в мережі

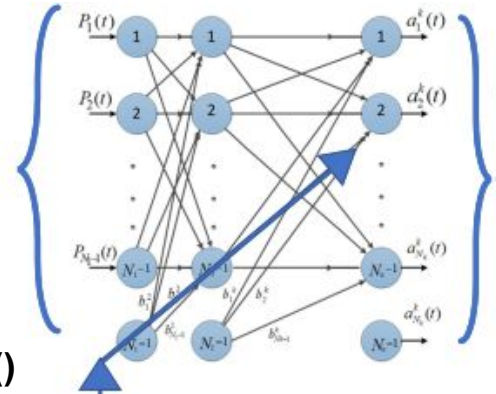


$$O_{output} = signmoid \begin{pmatrix} 0.975 \\ 0.888 \\ 1.254 \end{pmatrix}$$

$$O_{output} = \begin{pmatrix} 0.726 \\ 0.708 \\ 0.778 \end{pmatrix}$$

Навчання штучних нейронних мереж

$y = \{0, 1\}$

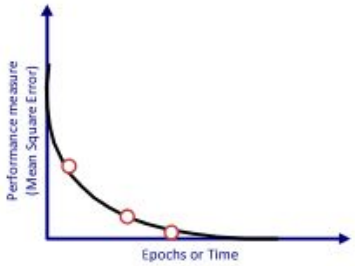


Training inputs X

Dense()

Predicted Output $\hat{Y}$

Desired (True) Output Y



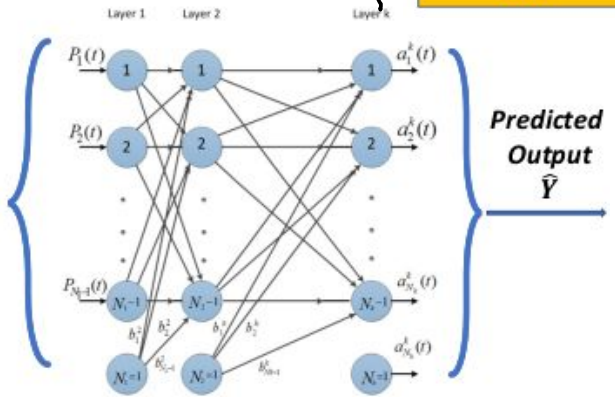
Error = $\hat{Y} - Y$

Update Network Weights

!!Навчання з вчителем

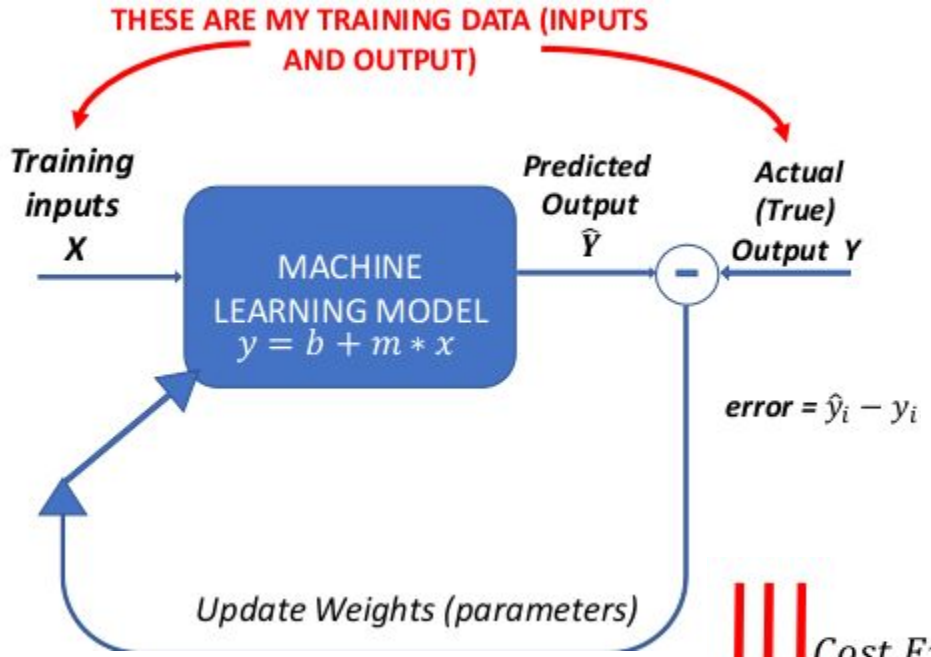
FREEZE NETWORK WEIGHTS AND TEST THE NETWORK WITH NEW DATA THAT THE MODEL HAS NEVER SEEN BEFORE

Testing inputs X



Predicted Output $\hat{Y}$

Навчання штучних нейронних мереж

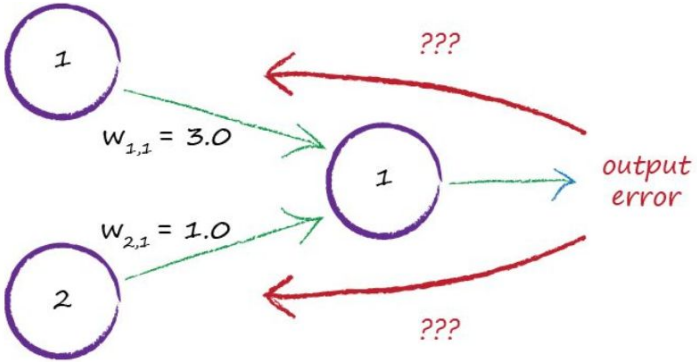


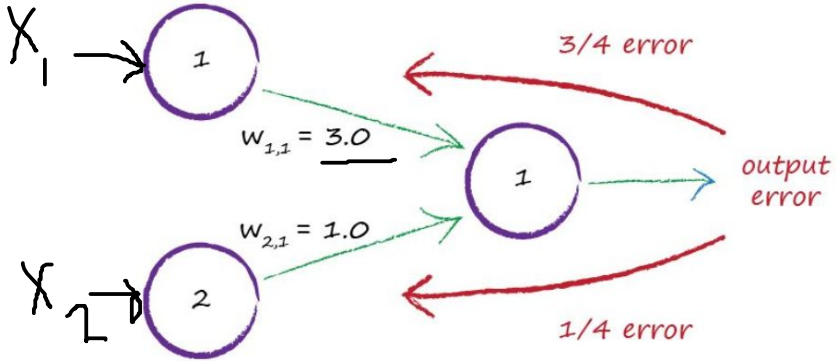
$$y = b + m * x$$

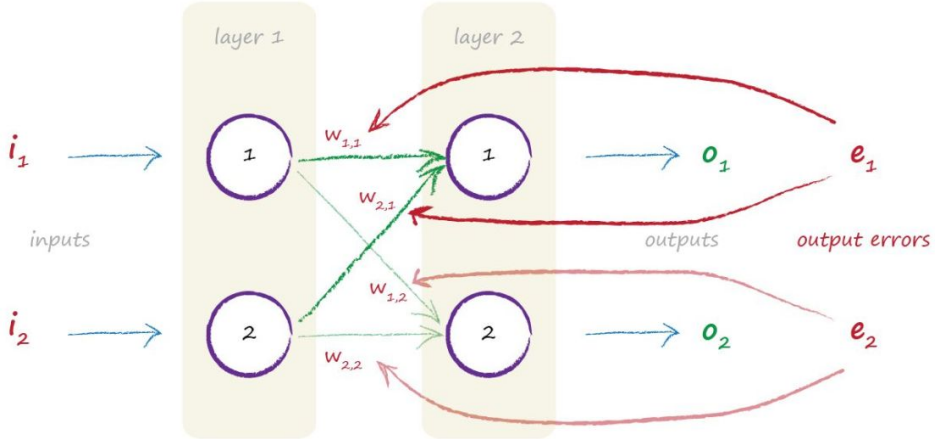
GOAL IS TO FIND BEST PARAMETERS

$$Cost\ Function\ f(m,b) = \frac{1}{N}\sum_{i=1}^{n}(error)^2 = \frac{1}{N}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

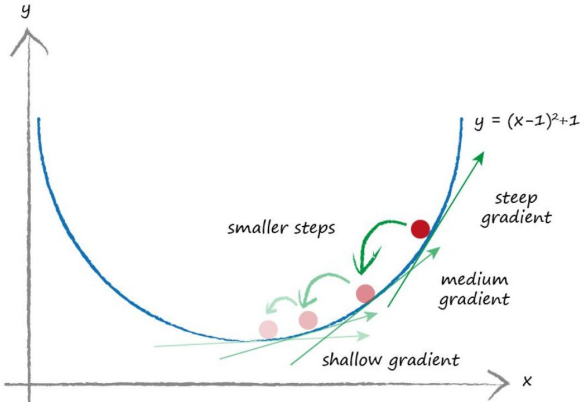
Зворотне поширення помилки





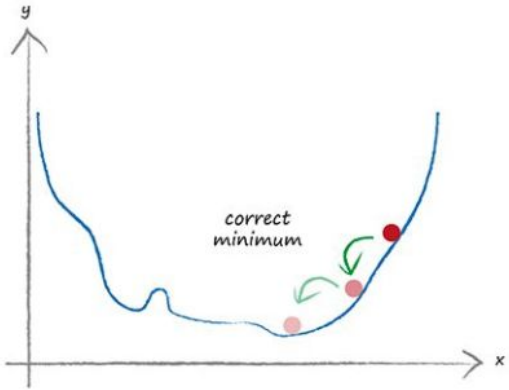


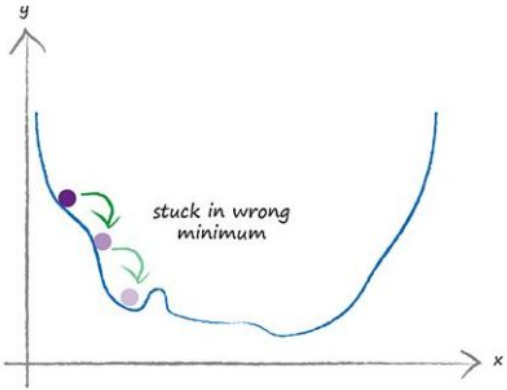
Оптимізація параметрів мережі

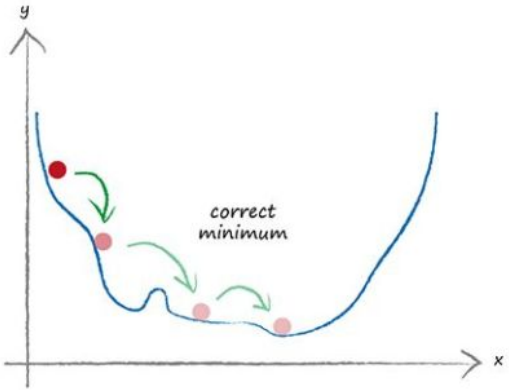


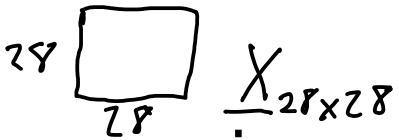
[Градієнт — Вікіпедія]

[Похідна — Вікіпедія]





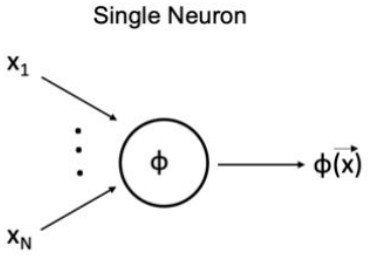


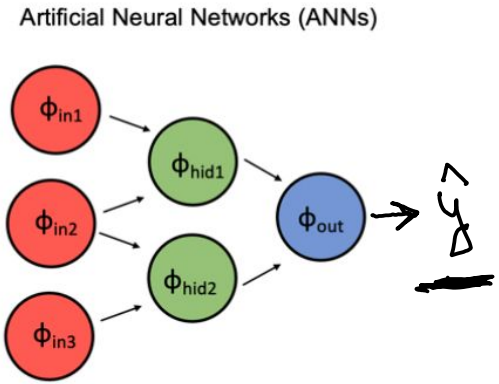


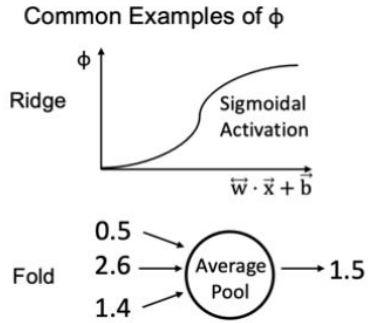
Навчальна вибірка

X	$x_{1\dots m}^{(1)}$	$x_{1\dots m}^{(2)}$	$x_{1\dots m}^{(3)}$	…
y	y_1	y_2	y_3	…

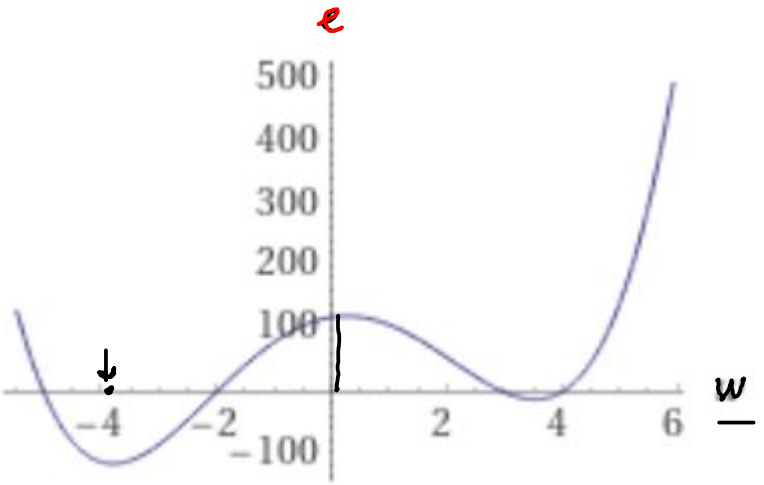
Оптимізація параметрів мережі







$$\phi_{out}(\dots) = \phi_{out}(\phi_{hid1}, \phi_{hid2}) = \phi_{out}(\phi_{hid1}(\phi_{in1}, \phi_{in2}), \phi_{hid2}(\phi_{in2}, \phi_{in3}))$$



$e = w^4 - 27w^2 + 14w + 120$

помилка прогнозування e мінімізується під час навчання

$$e(w) = \sum_i (\hat{y}_i - y_i)^2 = \sum_i (w_i x_i - y_i)^2 \to \min$$

Метод градієнтного спуску оптимізації функції $e(w)$

$e = w^4 - 27w^2 + 14w + 120$

$\frac{de}{dw} = 4w^3 - 54w + 14 \equiv de$

$w_0 = random(\dots)$

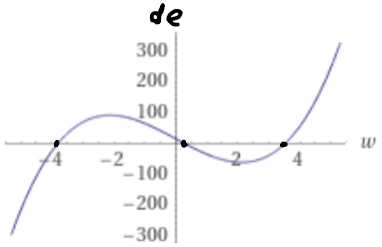
Основна формула градієнтного спуску:

$$w_{n+1} = w_n - \lambda de(w_n)$$

$w_1 \geq w_0 - \lambda de(w_0)$

де λ – швидкість навчання

n – кількість ітерацій алгоритму (кількість епох навчання)



критичні точки функції $de(w)$:

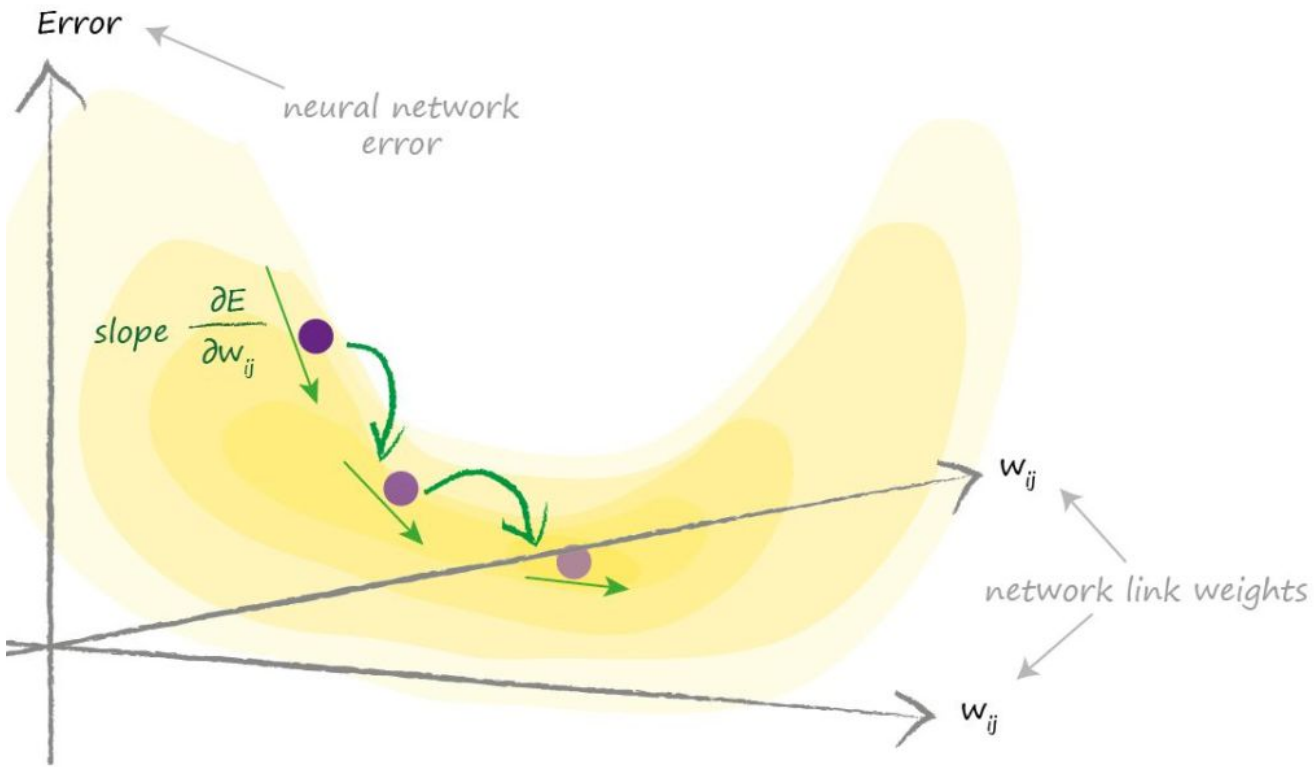
$w \approx -3,7976$ ⇐

$w \approx 0,2606$

$w \approx 3,5370$

https://colab.research.google.com/drive/1Smr751jgNi4M4qHxkPiNAZ6KSS7TzcFT?usp=sharing

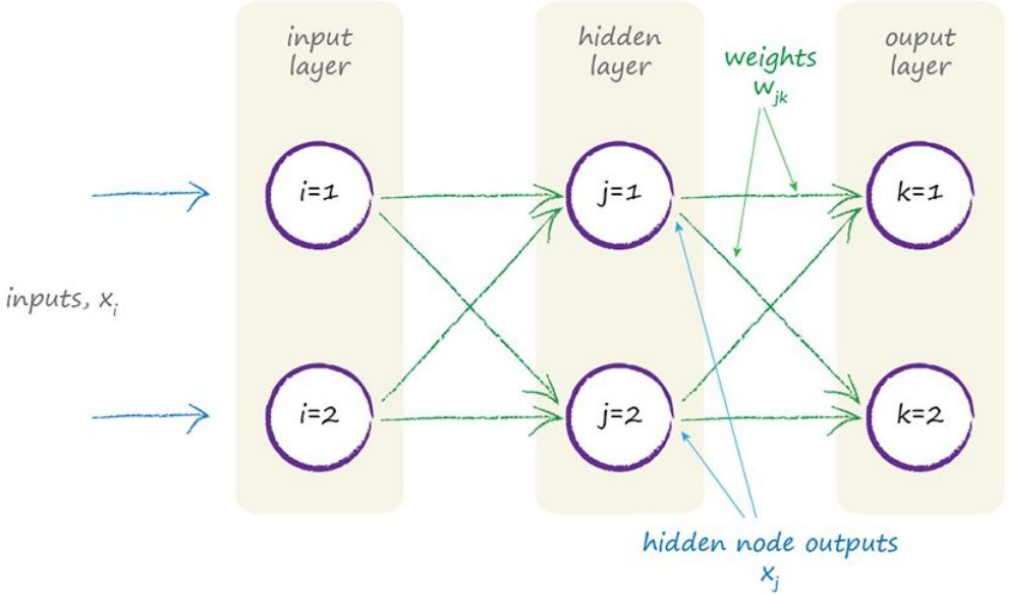
Оптимізація параметрів мережі



Градієнт – Вікіпедія

Похідна — Вікіпедія

Оптимізація параметрів мережі



node error = target - actual

$e_k = t_k - o_k$

outputs, o_k

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial}{\partial w_{jk}} (t_k - o_k)^2$$

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial o_k} \cdot \frac{\partial o_k}{\partial w_{jk}}$$

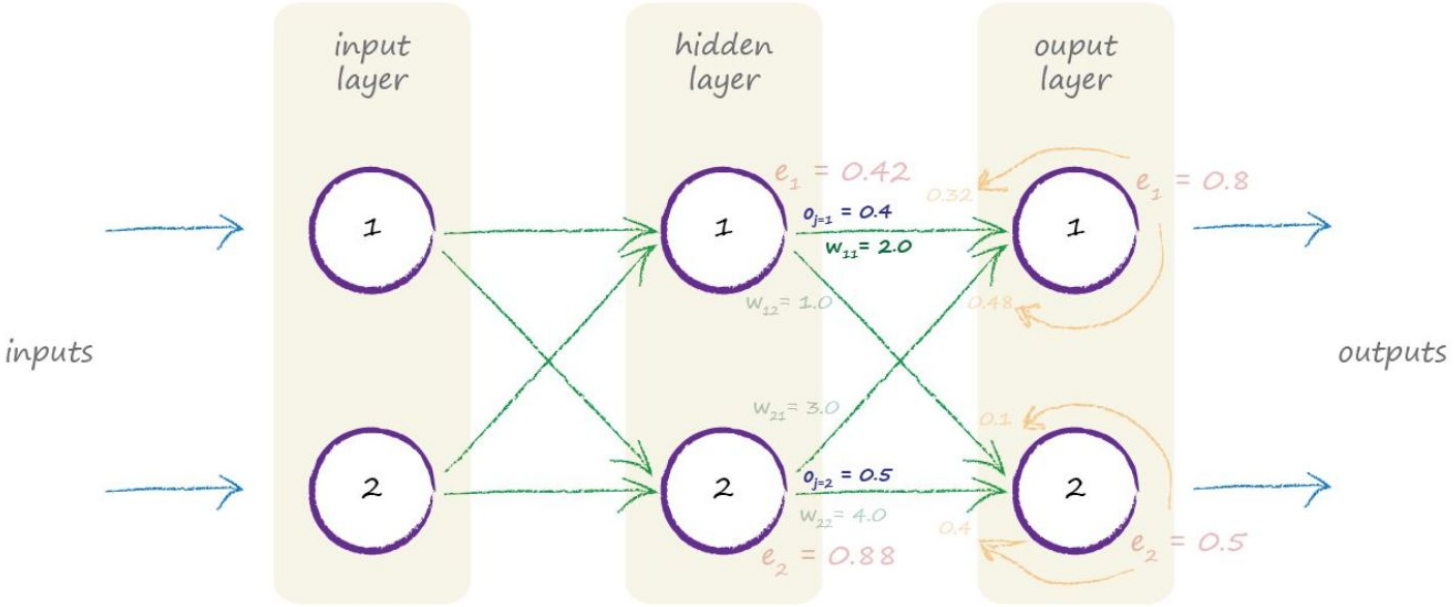
$$\frac{\partial E}{\partial w_{jk}} = -2(t_k - o_k) \cdot \frac{\partial o_k}{\partial w_{jk}}$$

$$\frac{\partial E}{\partial w_{jk}} = -2(t_k - o_k) \cdot \frac{\partial}{\partial w_{jk}} \text{sigmoid}(\Sigma_j w_{jk} \cdot o_j)$$

$$\frac{\partial}{\partial x} \text{sigmoid}(x) = \text{sigmoid}(x)\left(1 - \text{sigmoid}(x)\right)$$

$$\frac{\partial E}{\partial w_{jk}} = -(t_k - o_k) \cdot \text{sigmoid}(\Sigma_j w_{jk} \cdot o_j)(1 - \text{sigmoid}(\Sigma_j w_{jk} \cdot o_j)) \cdot o_j$$

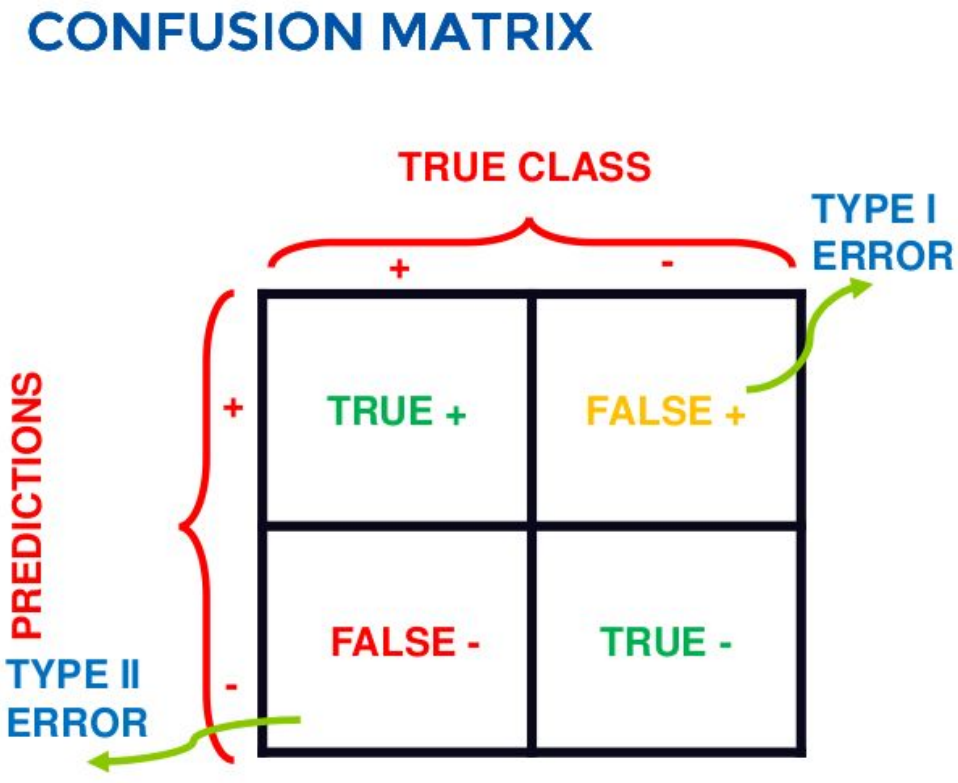
Метод градієнтного спуску



$$\frac{\partial E}{\partial w_{ij}} = -(e_j) \cdot sigmoid(\Sigma_i w_{ij} \cdot o_i)(1 - sigmoid(\Sigma_i w_{ij} \cdot o_i)) \cdot o_i$$

$$new\ w_{jk} = old\ w_{jk} - \alpha \cdot \frac{\partial E}{\partial w_{jk}}$$

Метрики якості навчання: матриця розбіжностей



[sklearn.metrics.confusion_matrix — scikit-learn 1.2.1 documentation]

[Матриця невідповідностей — Вікіпедія]