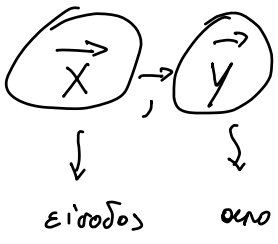
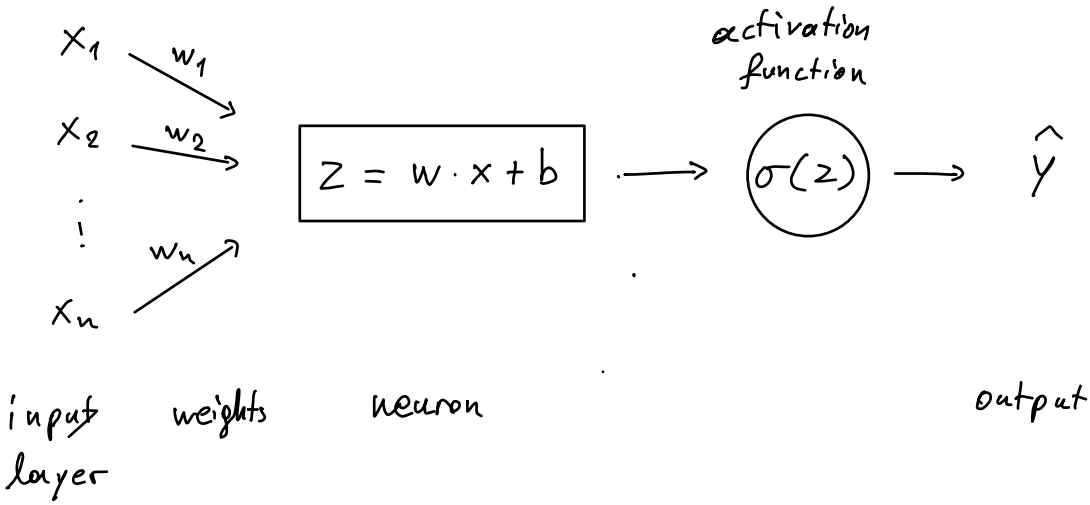
Εισαγωγή στα Νευρωνικά Δίκτυα



$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad \text{ή} \quad \vec{x} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & \ddots & & \vdots \\ \vdots & & & \\ x_{n1} & \cdots & & x_{nm} \end{bmatrix}$$

$$\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$



Βήμα 1.

Για κάθε input πολλαπλασιάζουμε το x_i με το αντίστοιχο βάρος w_i και αθροίζουμε όλες τις τιμές σε έναν αθροιστή.

Τα βάρη (weight) → strength of connection

δηλ. όσο μεγαλύτερο είναι το w τόσο σημαντικότερο είναι το x_i για το y

$w_1 > w_2$ ⇒ x_1 πιο σημαντικό από το x_2 για την πρόβλεψη του y

$$\Sigma = (x_1 \cdot w_1) + (x_2 \cdot w_2) \cdots + (x_n \cdot w_n)$$

$\vec{x}$, $\vec{w}$ $\quad i = 1 \dots n$

$$\vec{x} \cdot \vec{w} = (x_1 \cdot w_1) + (x_2 \cdot w_2) + \cdots + (x_n \cdot w_n)$$

$$\Sigma = \vec{x} \cdot \vec{w}$$

Bias b : χρησιμοποιείται για να πραγματοποιήσει συνολικό shift των τιμών.

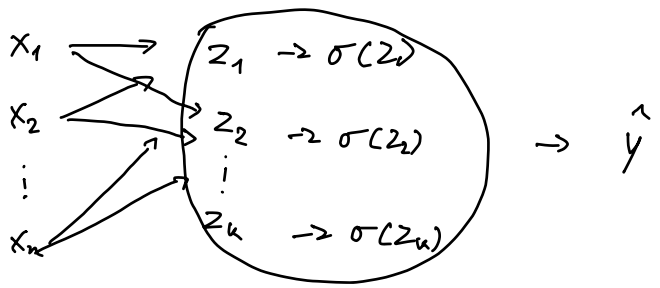
Βήμα 2

$$z = \vec{x} \cdot \vec{w} + b \quad \Longleftrightarrow \quad y$$

Παρατήρηση : με τον παραπάνω ορισμό του z η ανοχέτευση μεταξύ των inputs και των outputs είναι απολύτως γραμμική

Εφαρμόζουμε λοιπόν ένα non-linear activation function

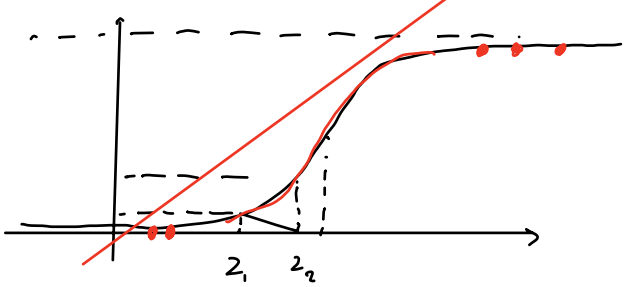
$$\hat{y} = \sigma(z) \qquad \text{όπου} \quad \sigma \text{ μη γραμμική συνάρτηση}$$



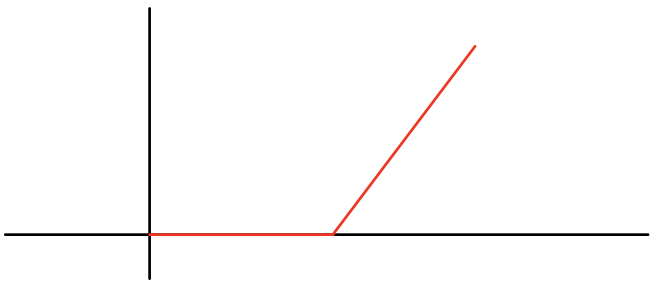
σ : μη γραμμική συνάρτηση

Συνήθεις συναρτήσεις σ (activation functions)

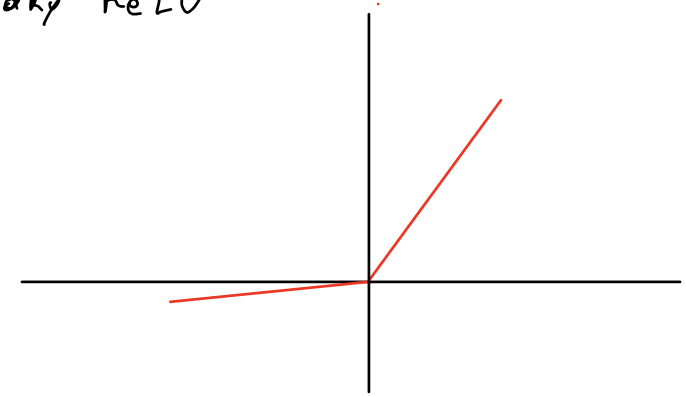
- Sigmoid



- ReLU : rectified linear unit activation function



· Leaky ReLU



activation function σ

ας πούμε ότι επιλέγουμε τη sigmoid

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

Αλγόριθμος εκπαίδευσης

1ο βήμα για την εκπαίδευση είναι η επιλογή κάποιου loss function

Τι είναι το loss function;
Είναι συναρτήσεις "διαφοράς μεταξύ επιλογών" με βάση την

οποία κρίνουμε αν το αποτέλεσμα που έχουμε προβλέψει είναι σωστά ή όχι στην πραγματική ζημιά.

Στις περισσότερες περιπτώσεις το loss function που χρησιμοποιούμε είναι η μέση τετραγωνική απόκλιση (mean squared error - MSE)

$$MSE_i = \left(y_i - \hat{y}_i\right)^2$$

Cost function : μέση ζημιά του loss function (C)

$$C = MSE = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2 = \frac{1}{n} MSE_i$$

Έχουμε ξεκινήσει με random w. Βρίσκουμε με αυτά τα τυχαία βάρη πόσο απέχει η πρόβλεψή μας από την πραγματική ζημιά.

Πρέπει να μεταβάλλουμε τα w ώστε να πετύχουμε ένα συνολικά μικρότερο C.

$$\frac{\partial C}{\partial w_i} = \frac{\partial C}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial w}$$

$$\frac{\partial C}{\partial \hat{y}} = ? \qquad \frac{\partial \hat{y}}{\partial z} = ? \qquad \frac{\partial z}{\partial w} = ?$$

$$\frac{\partial C}{\partial \hat{y}} = \frac{\partial}{\partial \hat{y}} \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = 2 \cdot \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)$$

έχουμε $\vec{y} = [y_1, y_2 \dots y_n]$, $\vec{\hat{y}} = [\hat{y}_1, \hat{y}_2 \dots \hat{y}_n]$

$$\frac{\partial C}{\partial \hat{y}} = \frac{2}{n} \sum (y - \hat{y})$$

$$\frac{\partial \hat{y}}{\partial z} = \frac{\partial}{\partial z} \hat{y} = \frac{\partial}{\partial z} \sigma(z) = \frac{\partial}{\partial z} \left(\frac{1}{1+e^{-z}} \right)$$

$$= -\frac{1}{(1+e^{-z})^2} (e^{-z})' = \frac{e^{-z}}{(1+e^{-z})^2} =$$

$$= \frac{1}{1+e^{-z}} \cdot \frac{e^{-z}}{1+e^{-z}} =$$

$$= \frac{1}{1+e^{-z}} \cdot \left(1 - \frac{1}{1+e^{-z}} \right) =$$

$$= \sigma(z) \cdot (1 - \sigma(z))$$

$$\frac{\partial z}{\partial w_i} = \frac{\partial}{\partial w_i} \sum_{i=1}^{n} (x_i \cdot w_i + b) = x_i$$

Συνολικά έχουμε:

$$\frac{\partial C}{\partial w_i} = \frac{2}{n} \cdot \sum (y - \hat{y}) \cdot \sigma(z) \cdot (1 - \sigma(z)) \, x_i$$

Βρήκαμε το $\frac{\partial C}{\partial w_i}$ το οποίο θα είναι η βάση μας για το gradient decent που θα χρησιμοποιηθεί για το optimization μέχρι ένα όριο σύγκλισης

$$w_i' = w_i - \alpha \frac{\partial C}{\partial w_i}$$

α: learning rate

όσο μεγαλύτερο είναι το α τόσο πιο γρήγορα ο αλγόριθμος προσπαθεί να μάθει

αν θελήσουμε να κάνουμε optimize και στο bias χρησιμοποιούμε αντίστοιχα το gradient descent για το bias

$$b' = b - \alpha \frac{\partial C}{\partial b}$$