

# NEURAL NETWORK: FROM NEURON TO BRAIN

---

Come costruire la propria rete neurale.

Risolvi il seguente calcolo:

$$\underline{9 \times 2} + \underline{15 : 3} + 4 + \underline{12 : 4 \times 2} - \underline{3 \times 5 \times 2} =$$

$$18 + 5 + 4 + \underline{3 \times 2} - \underline{15 \times 2} =$$

$$\underline{18} + \underline{5} + 4 + 6 - 30 =$$

$$\underline{23} + 4 + 6 - 30 =$$

$$\underline{27} + 6 - 30 =$$

$$33 - 30 = 3$$

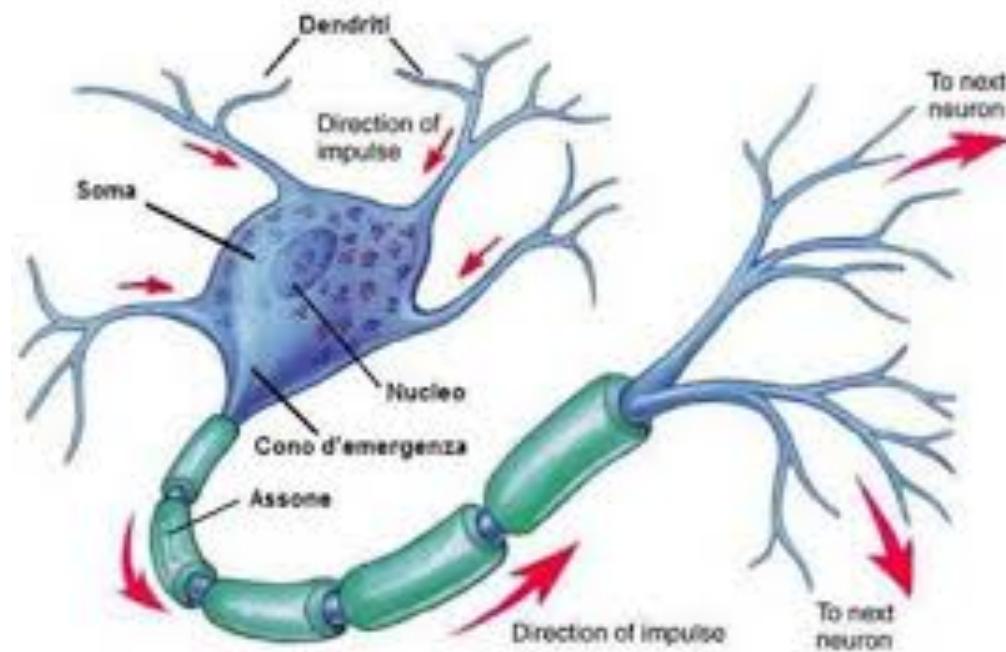
# Quante persone ci sono nella foto?



Problema	Computer	Uomo
Svolgere calcoli	<i>Facile</i>	<i>Difficile</i>
Semantica parole e immagini	<i>Difficile</i>	<i>Facile</i>

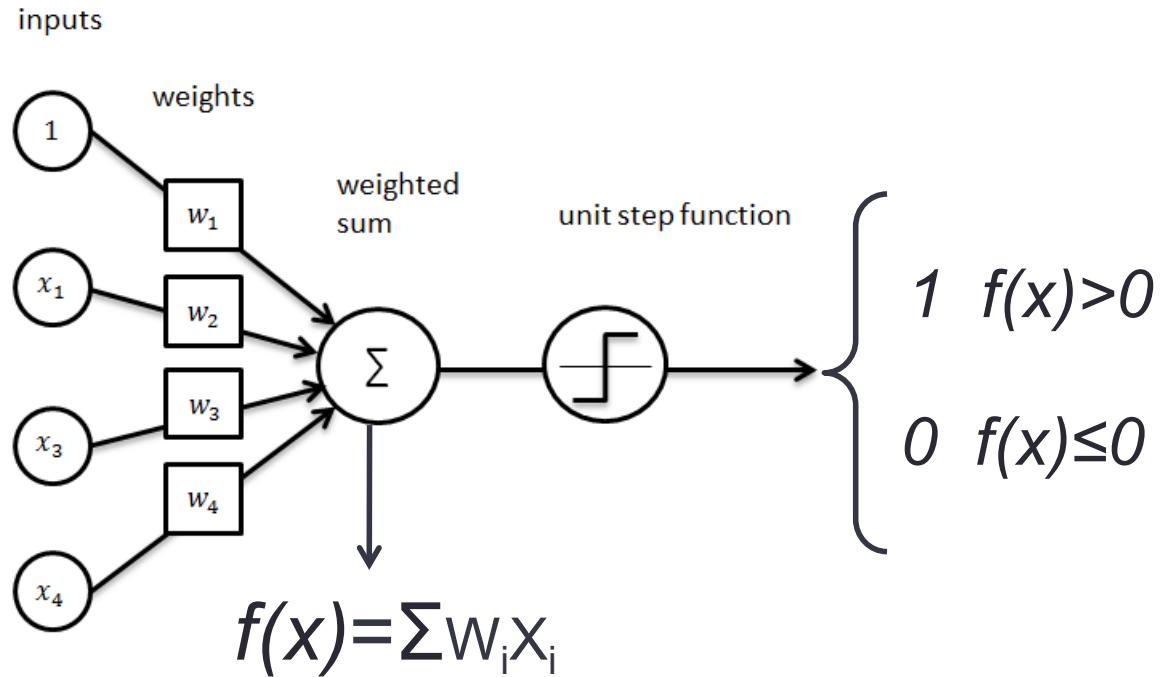
# Neurone

- INPUT
- ATTIVAZIONE
- OUTPUT



# Perceptron

- **INPUT**
- **ATTIVAZIONE**
- **OUTPUT**

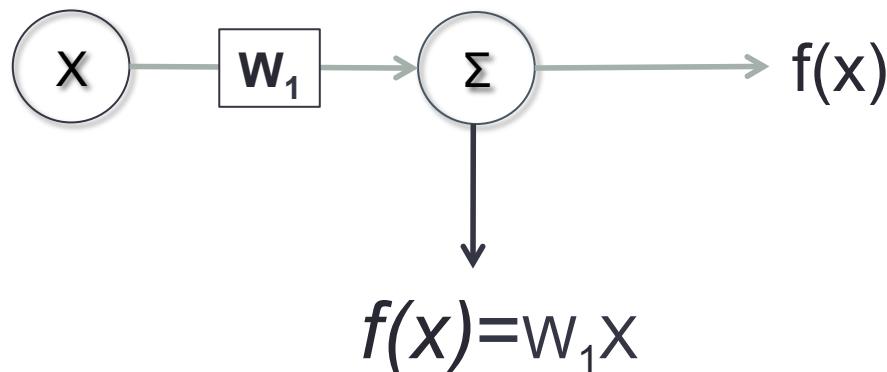


# Esempio

**Vogliamo definire il fattore di conversione tra chilometro e miglio:**

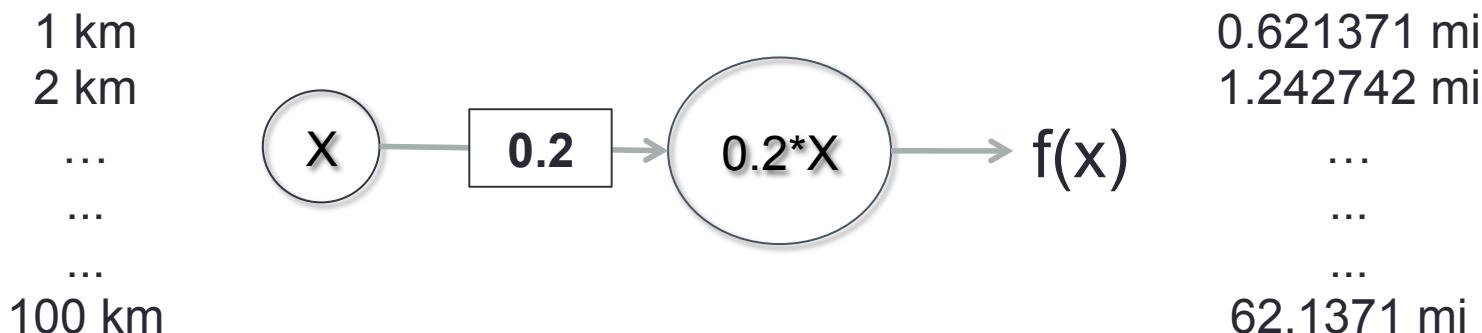
$$1 \text{ Km} = 0.621371 \text{ mi} \rightarrow \text{km}/0.621371 = \text{mi}$$

1. **Definisco INPUT (X il valore in km)**
2. **Definisco il numero di Nodi della Rete**
3. **Definisco la mia funzione di ATTIVAZIONE**
4. **Definisco OUTPUT (f(x) il mio valore in mi)**

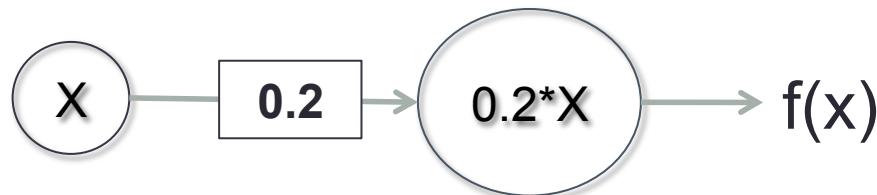


# DATASET E TRAINING

Quando noi definiamo la rete, i pesi ( $w$ ) sono numeri randomici. Dobbiamo trovare un insieme di dati (DATASET) che risolvono il problema per allenare la nostra rete

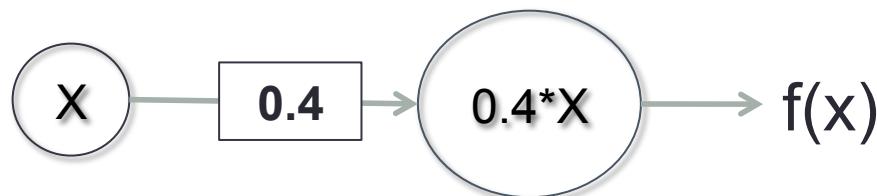


# TRAINING

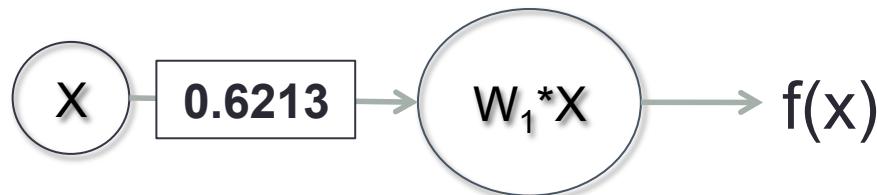


Dato che i valore di OUTPUT differiscono occorre modificare i pesi tramite una Funzione di Back Propagation :

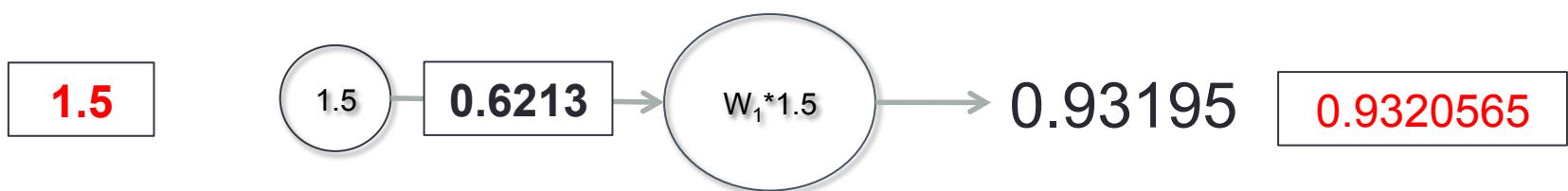
$$w_{\text{new}} = w_{\text{old}} - \alpha \delta E / \delta W$$



# TESTSET E TESTING



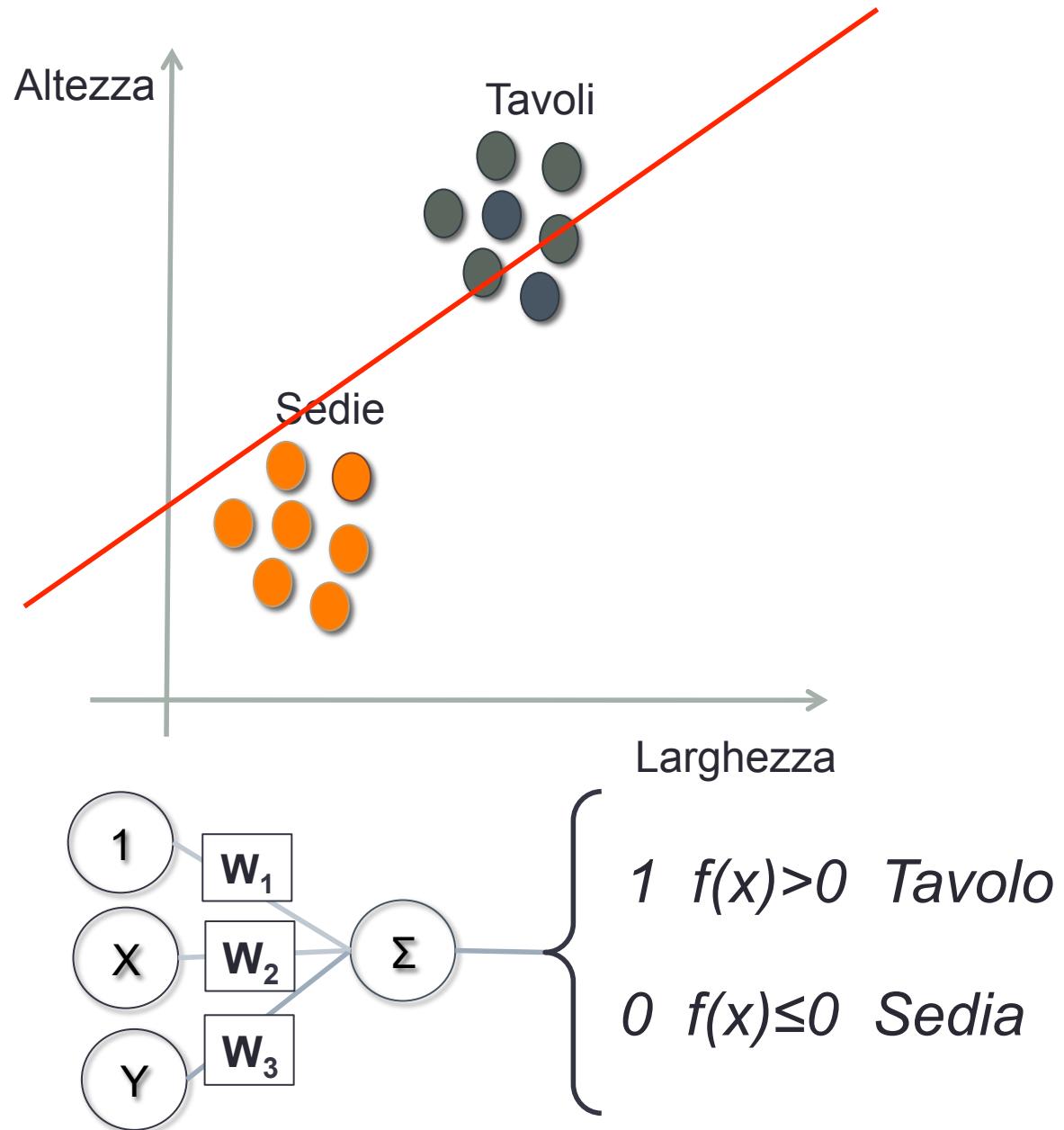
Una volta che la rete è pronta possiamo ne testiamo il funzionamento con un set di dati.



Se l'errore tra il dato di uscita e il valore reale ci soddisfa possiamo interrompiamo  
Il training altrimenti dobbiamo continuarlo

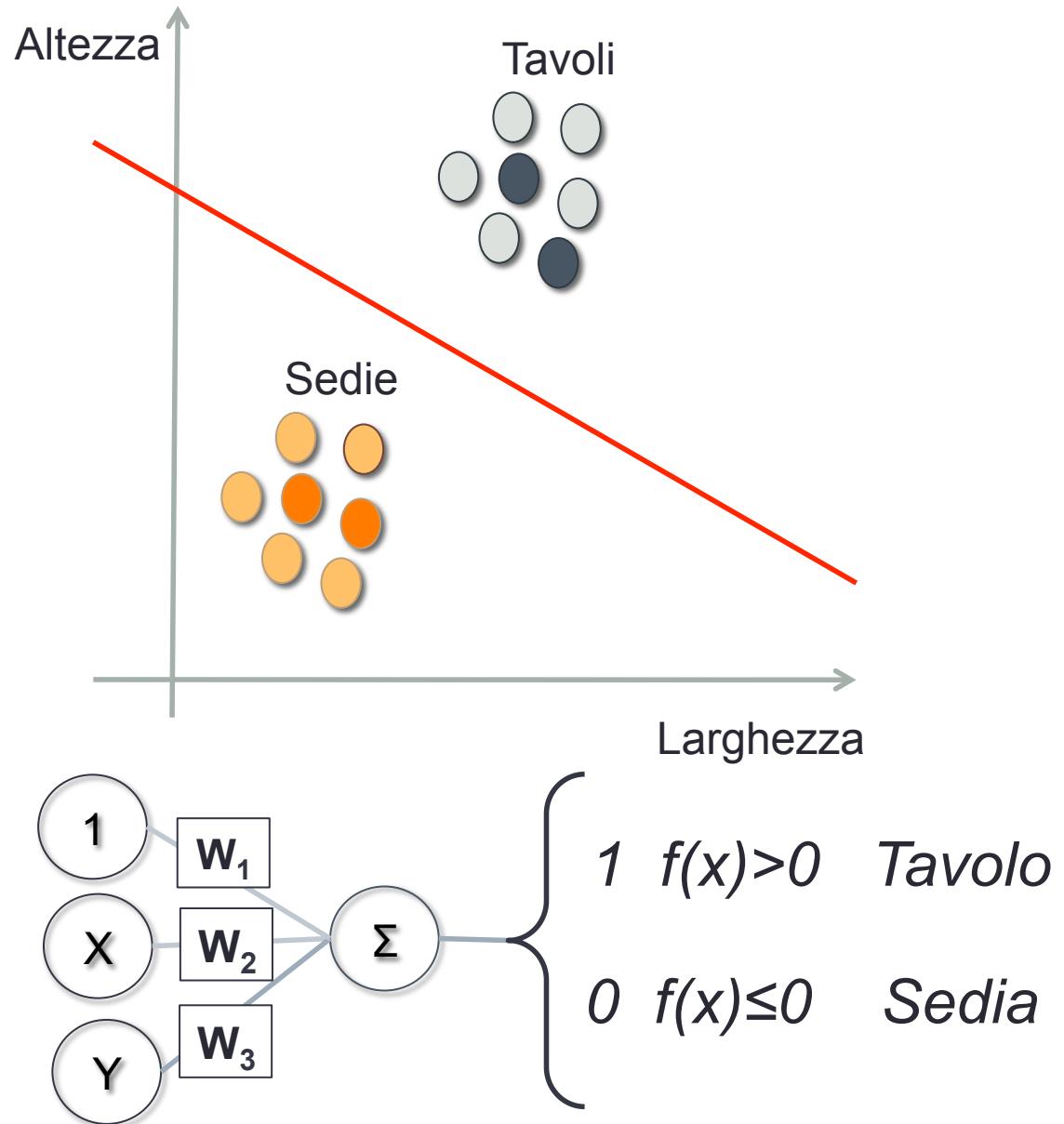
## Perceptron

- INPUT
- ATTIVAZIONE
- OUTPUT

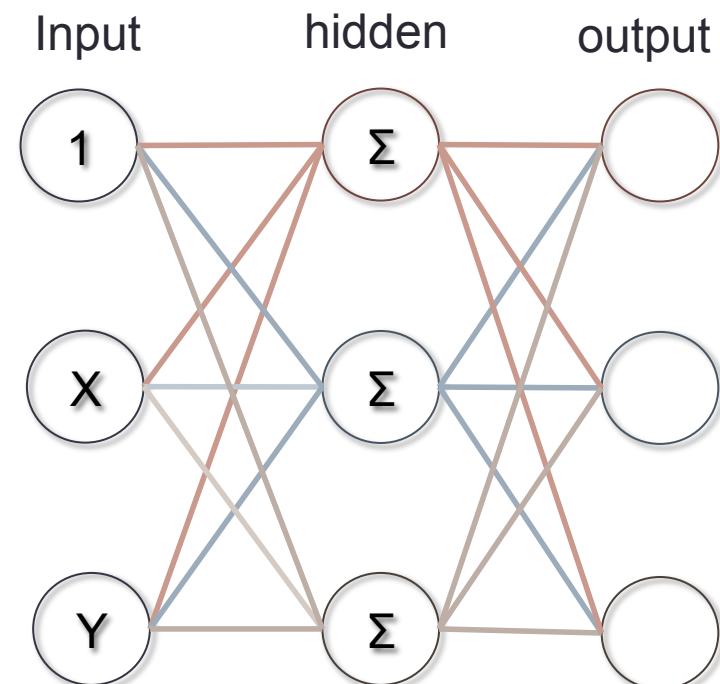
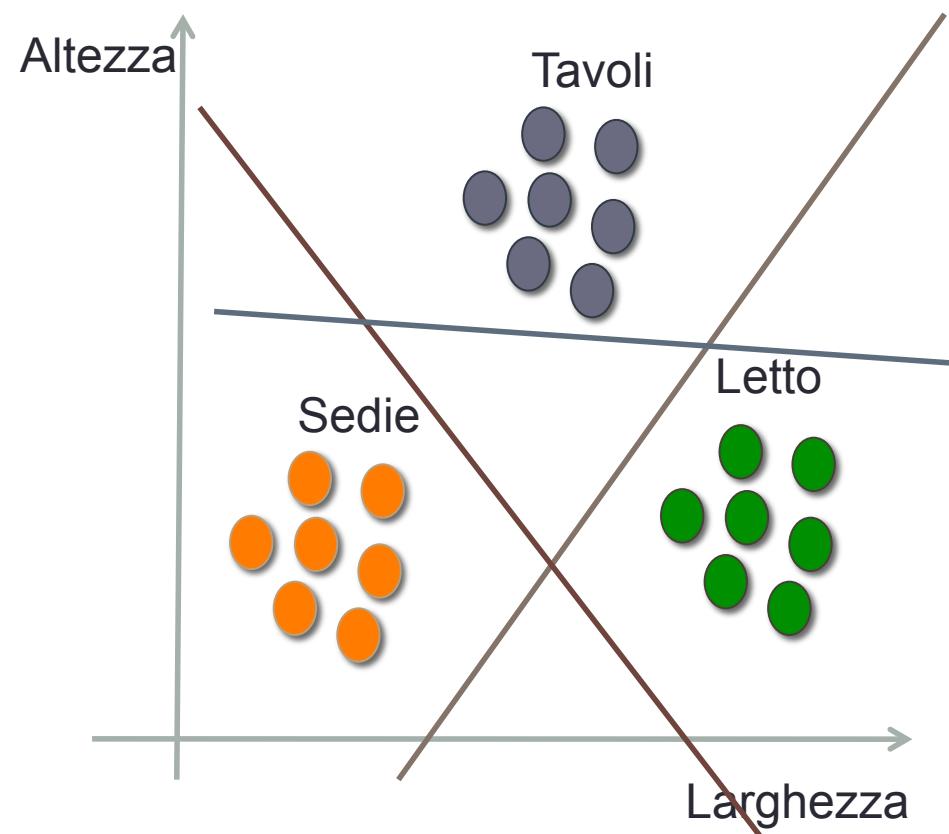


## Perceptron

- DATASET
- TRAINING
- TESTSET



# Neural Network



## MNIST

## • INPUT

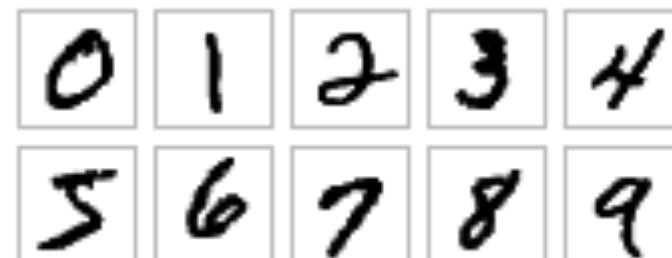
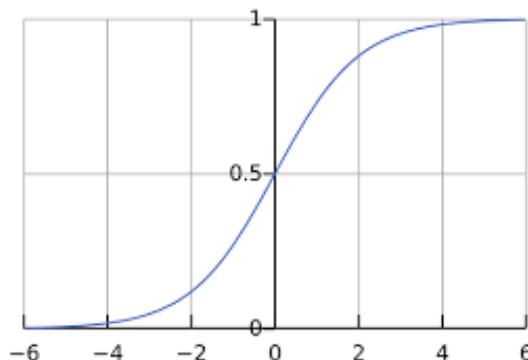
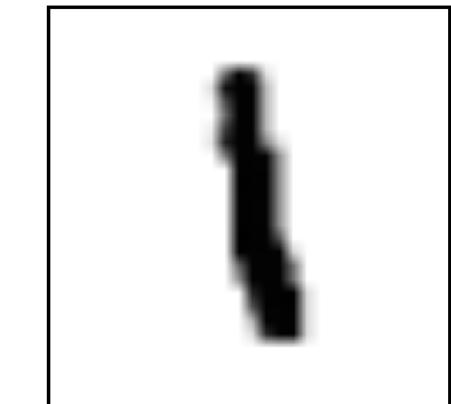
**Separo l'immagine in  
una matrice 28X28=784**

## • ATTIVAZIONE

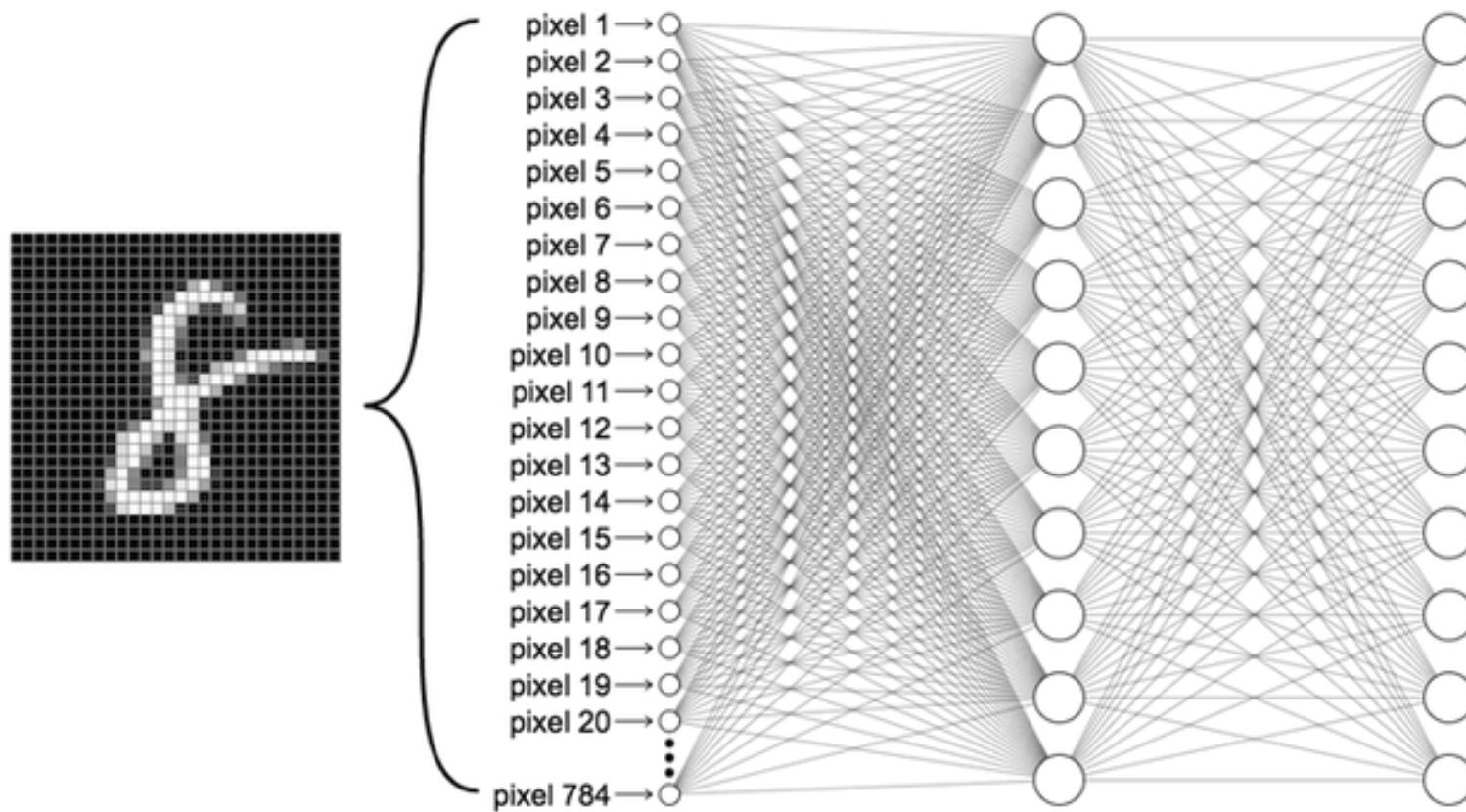
## Funzione matematica *softmax*

## • **OUTPUT**

**Ci sono solo 10 possibili uscite**



# MNIST: Neural Network



# Google: doodle classification

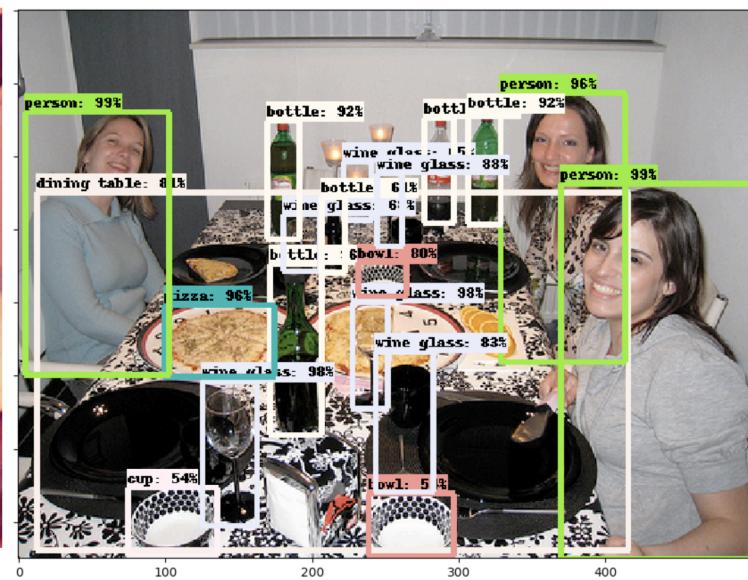
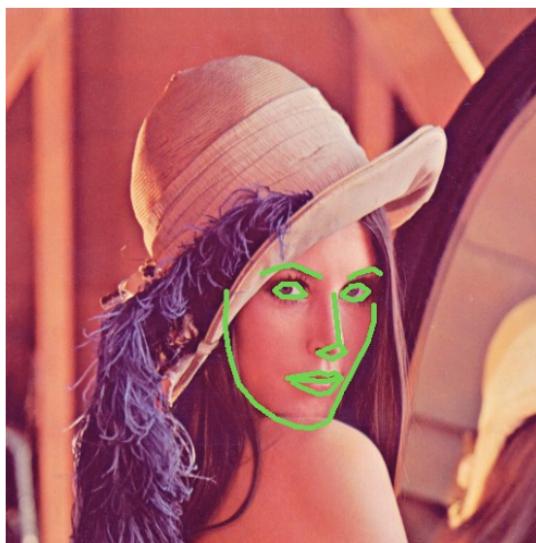
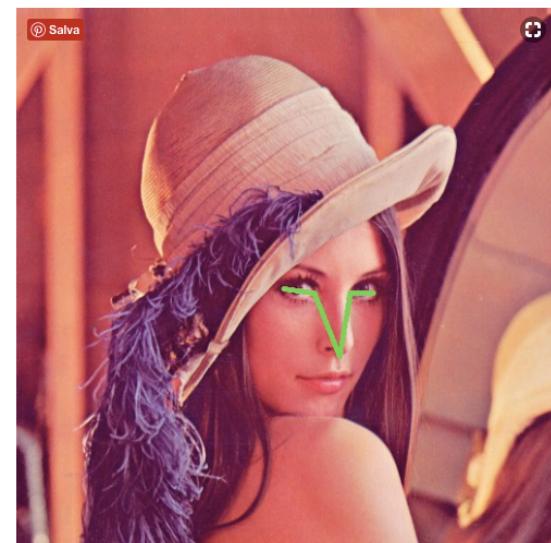


Una rete neurale può imparare a riconoscere i disegni?

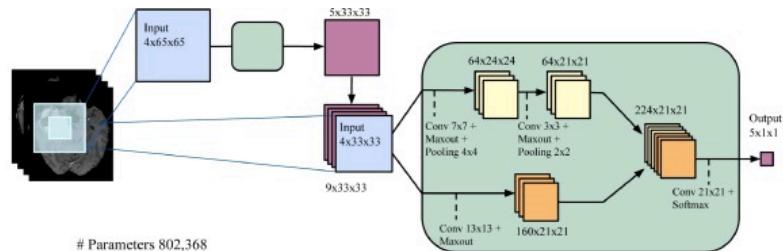
Agevola il suo apprendimento aggiungendo i tuoi disegni alla [più vasta raccolta di disegni al mondo](#), pubblicamente condivisa per contribuire alla ricerca sul machine learning.

[Disegniamo!](#)

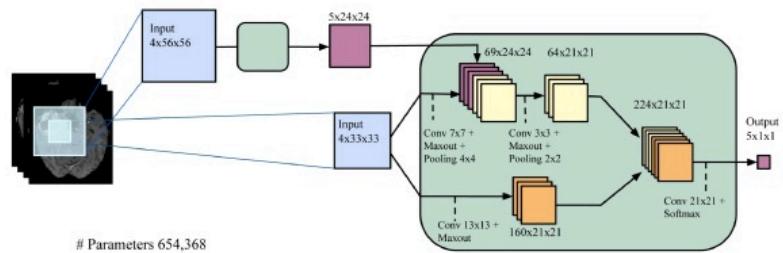
# Neural Network



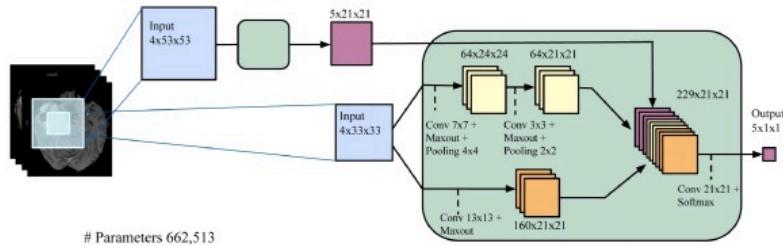
# Neural Network



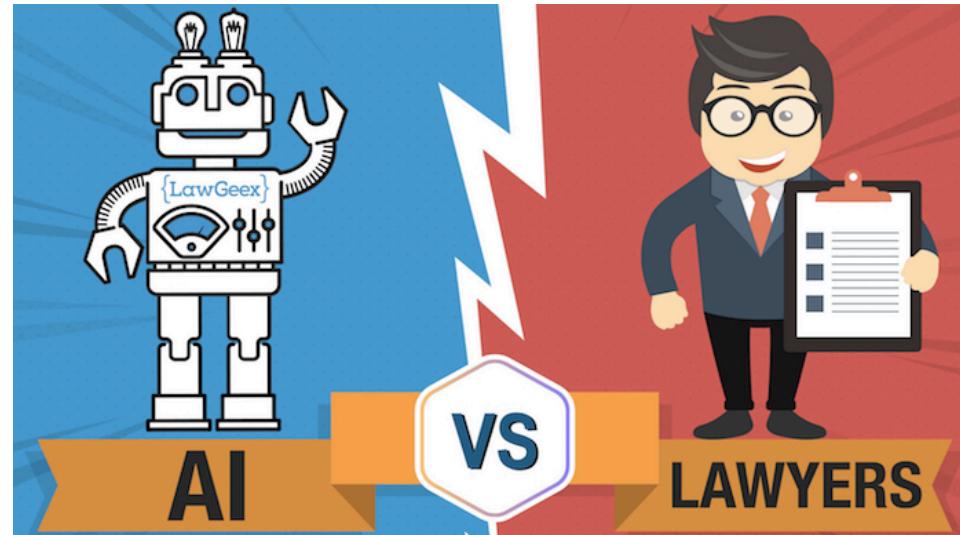
(a) Cascaded architecture, using input concatenation (INPUTCASCADECNN).



(b) Cascaded architecture, using local pathway concatenation (LOCALCASCADECNN).



(c) Cascaded architecture, using pre-output concatenation, which is an architecture with properties similar to that of learning using a limited number of mean-field inference iterations in a CRF (MFCASCADECNN).



# Flappy Bird

- **INPUT**

**Posizione X del tubo più vicino**

**Posizione Y inizio apertura**

**Posizione Y della fine apertura**

**Posizione Y dell'uccello**

**Velocità Y dell'uccello**

- **ATTIVAZIONE**

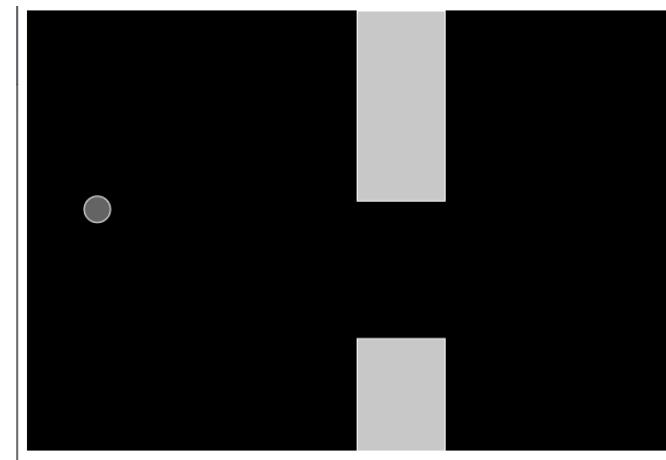
**Funzione matematica *softmax***

- **OUTPUT**

**Due OUTPUT probabilità di saltare o non saltare.**



L'obiettivo è quello di totalizzare il punteggio più alto possibile (il giocatore guadagna un punto per ogni coppia di tubi attraversata), facendo volare un uccello attraverso una serie di tubi, evitando di farlo scontrare con essi o di farlo cadere per terra. L'uccello, quando non riceve comandi si abbassa e cade verso il fondo.



- Nel nostro caso semplificato l'uccellino non può avanzare nella mappa e la sua posizione in X è fissa

# Contatti e link materiale

[petrangolini.paolo@gmail.com](mailto:petrangolini.paolo@gmail.com)

**Coding train**

[https://www.youtube.com/channel/UCvjqXvBlbQiydffZU7m1\\_aw](https://www.youtube.com/channel/UCvjqXvBlbQiydffZU7m1_aw)

<http://natureofcode.com/book>

<http://natureofcode.com/book/chapter-10-neural-networks/>

**Gli esempi potete scaricarli da:**

<https://github.com/CodingTrain/Toy-Neural-Network-JS>

GRAZIE  
DELL'ATTENZIONE