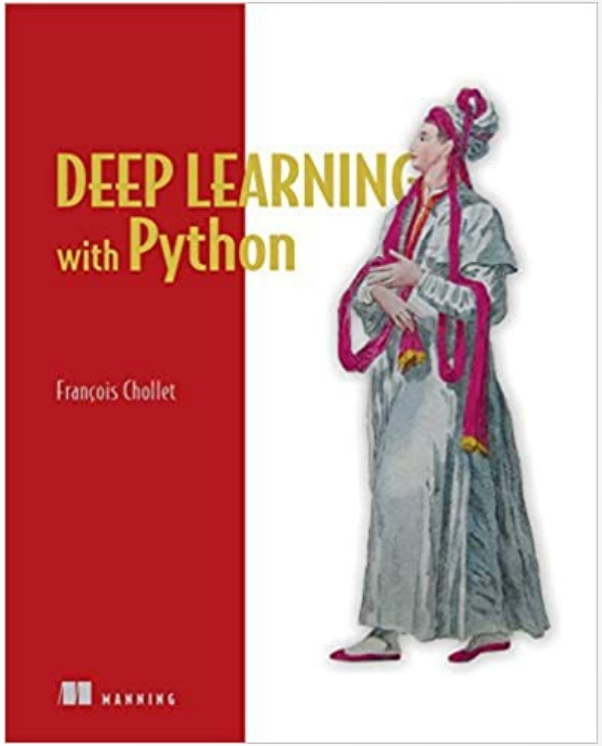
Deep Learning con Keras

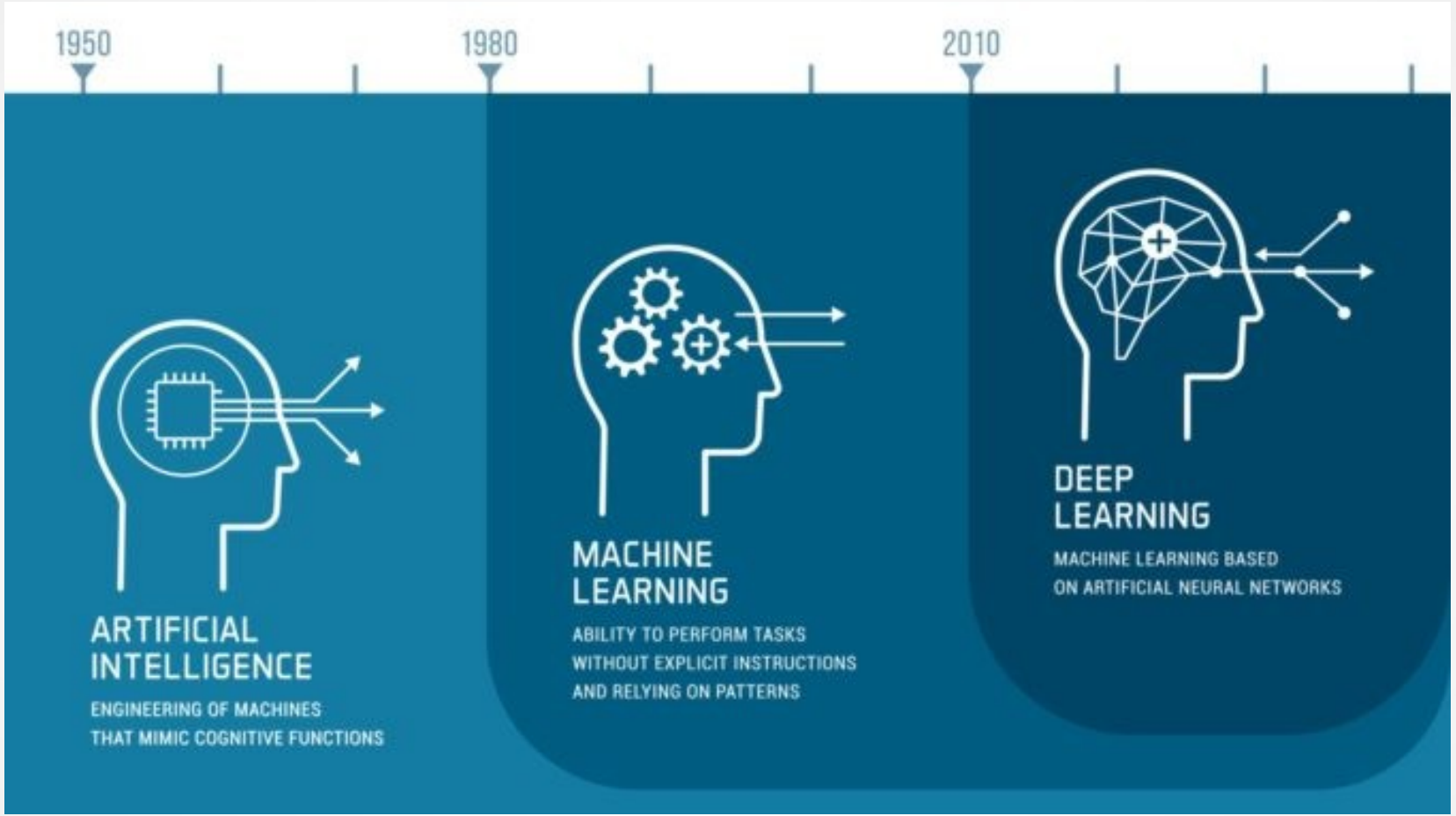
Prof. Gianluca Amato
Dipartimento di Economia
Università “G. d’Annunzio” di Chieti-Pescara
Ultimo aggiornamento: 22 mar 2022

Deep Learning con Keras

- **Deep Learning con Keras** è un breve ciclo di seminari sull'utilizzo della libreria Keras per il deep learning.
- I seminari sono basati sul libro:
  - FRANÇOIS CHOLLET
    Deep Learning with Python
    Manning

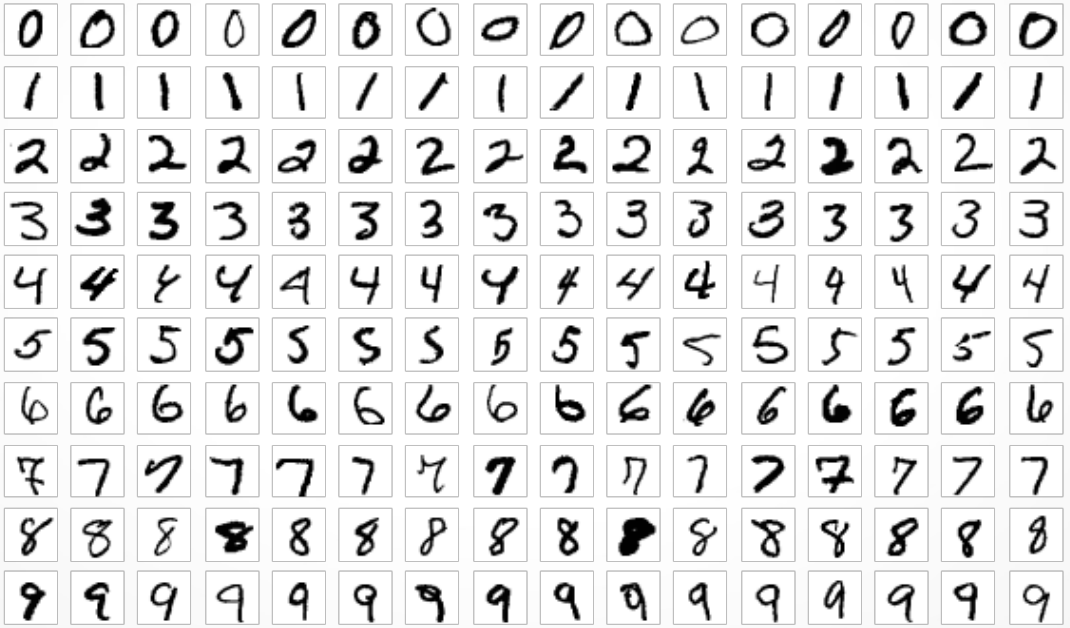


Cosa è il deep learning



Riconoscimento di immagini (1)

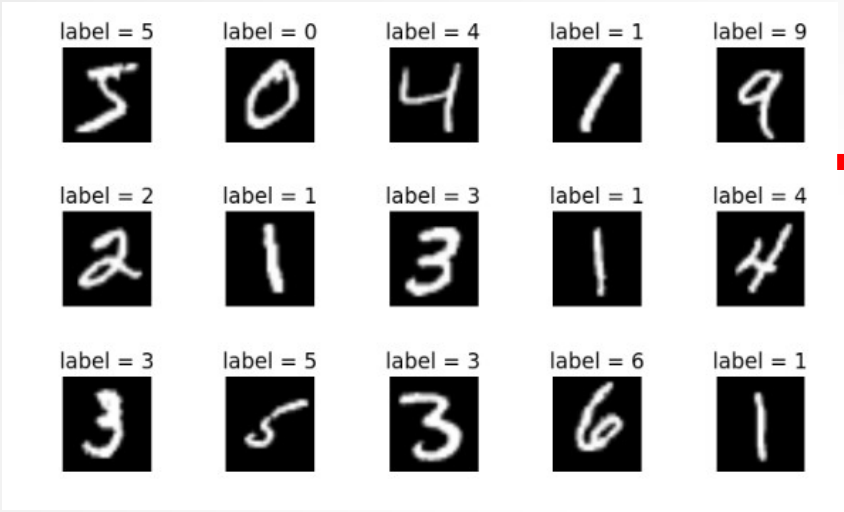
- Problema: data una immagine di una cifra scritta a mano, riconoscere di quale cifra si tratta.
- 70.000 immagini prese dal database MNIST



Riconoscimento di immagini (2)

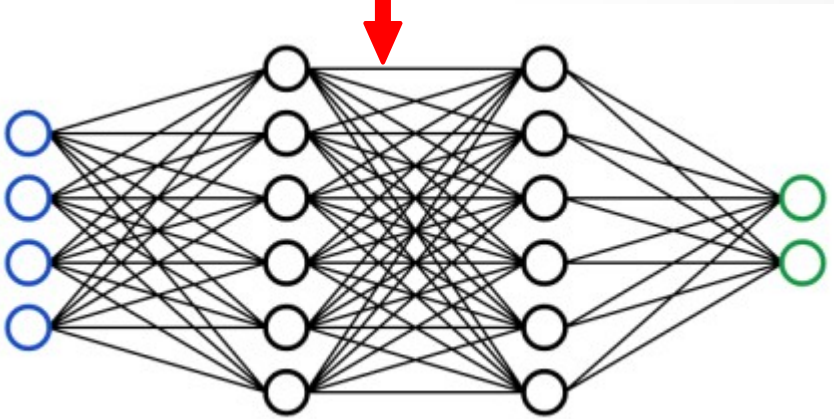
- Problema banale per un essere umano, ma è una cosa che facciamo in maniera inconscia.
- Esempio: Chiediamoci cosa è un 9 ?
  - Intuizione: un 9 ha un cappio in cima e un tratto verticale in basso a destra
  - Difficile da esprimere algoritmicamente
  - Se cerchiamo di essere più precisi, anneghiamo in un mare di eccezioni e casi particolari.

Machine learning (addestramento)



Immagini di addestramento
(con etichette)

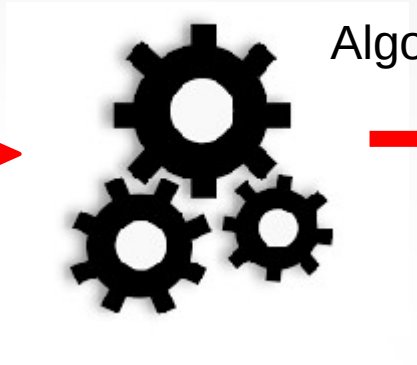
Algoritmo di
apprendimento



Rete neurale (o altro modello)

Machine learning (inferenza)

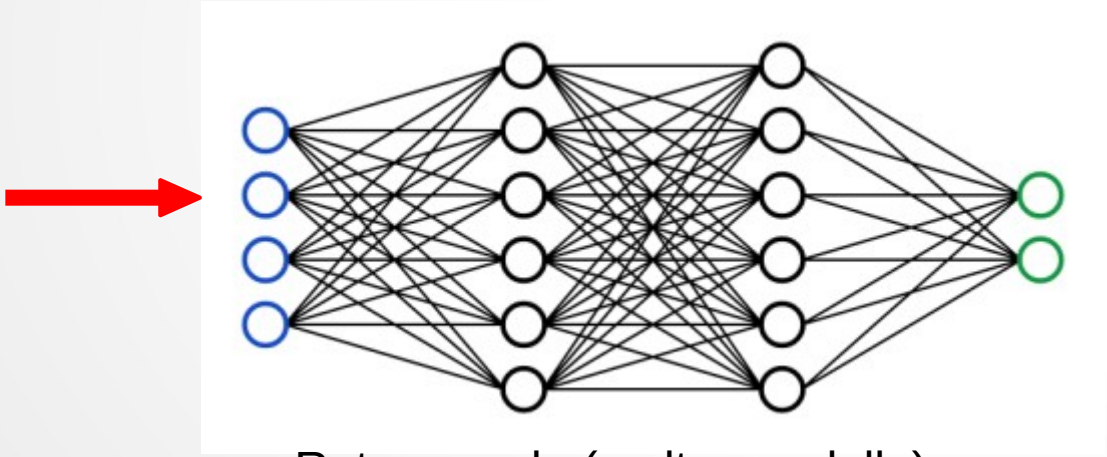
Immagini nuove senza etichetta



Algoritmo di inferenza

9 8 1 5 1
4 4 1 4 9

Predizioni

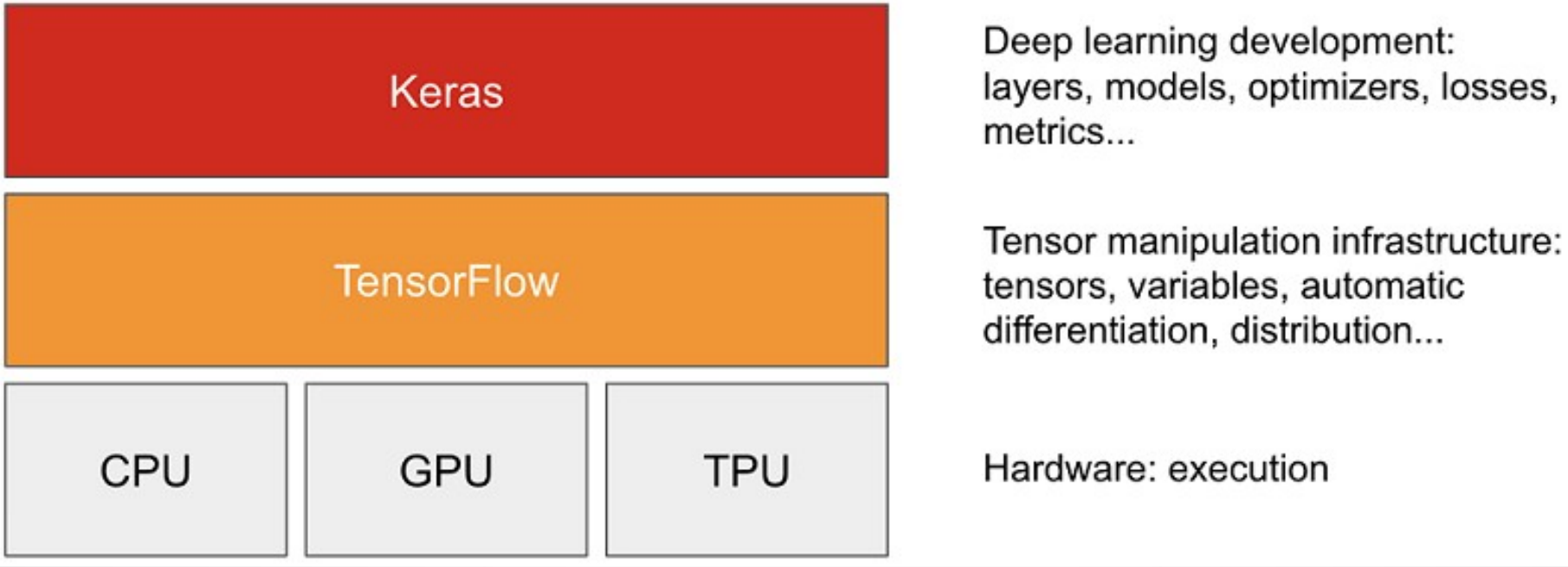


Rete neurale (o altro modello)

= errore

Keras e Tensorflow

- Keras è una libreria per il deep learning
- Si basa su un’altra libreria, Tensorflow, per il calcolo su tensori



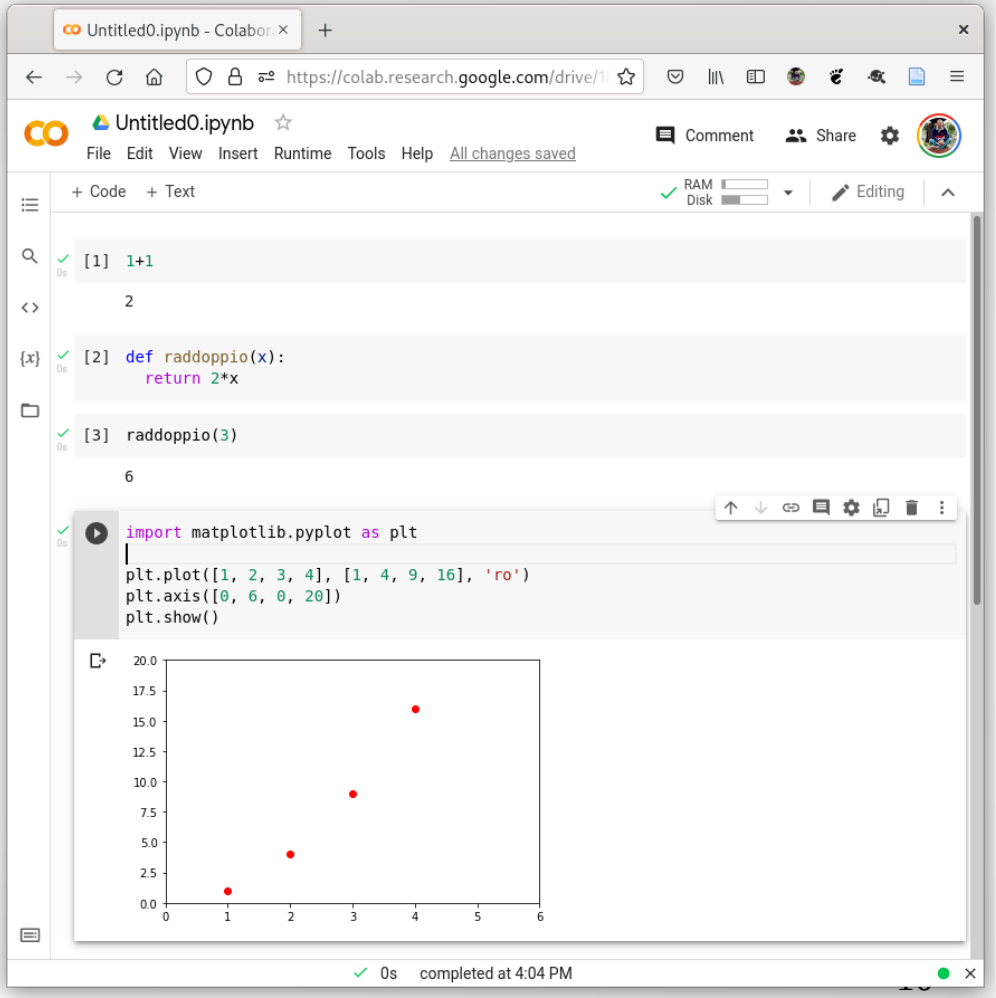
Python

- Python è il linguaggio di programmazione principe per il deep learning

```
# Esempio di codice Python che esegue la
# somma dei numeri da 0 fino ad n.
n = 10
sum = 0
for y in range(n+1):
  sum = sum + y
  print(y)
print(f"La somma dei primi {n} numeri è {sum}")
```

Notebook

- Normalmente durante la fase di sviluppo di un modello di machine learning si usa un approccio interattivo
  - do un comando
  - guardo il risultato
- Piuttosto che un ambiente di sviluppo classico per la scrittura di programmi, come Eclipse, usiamo ambienti basati sul concetto del notebook.

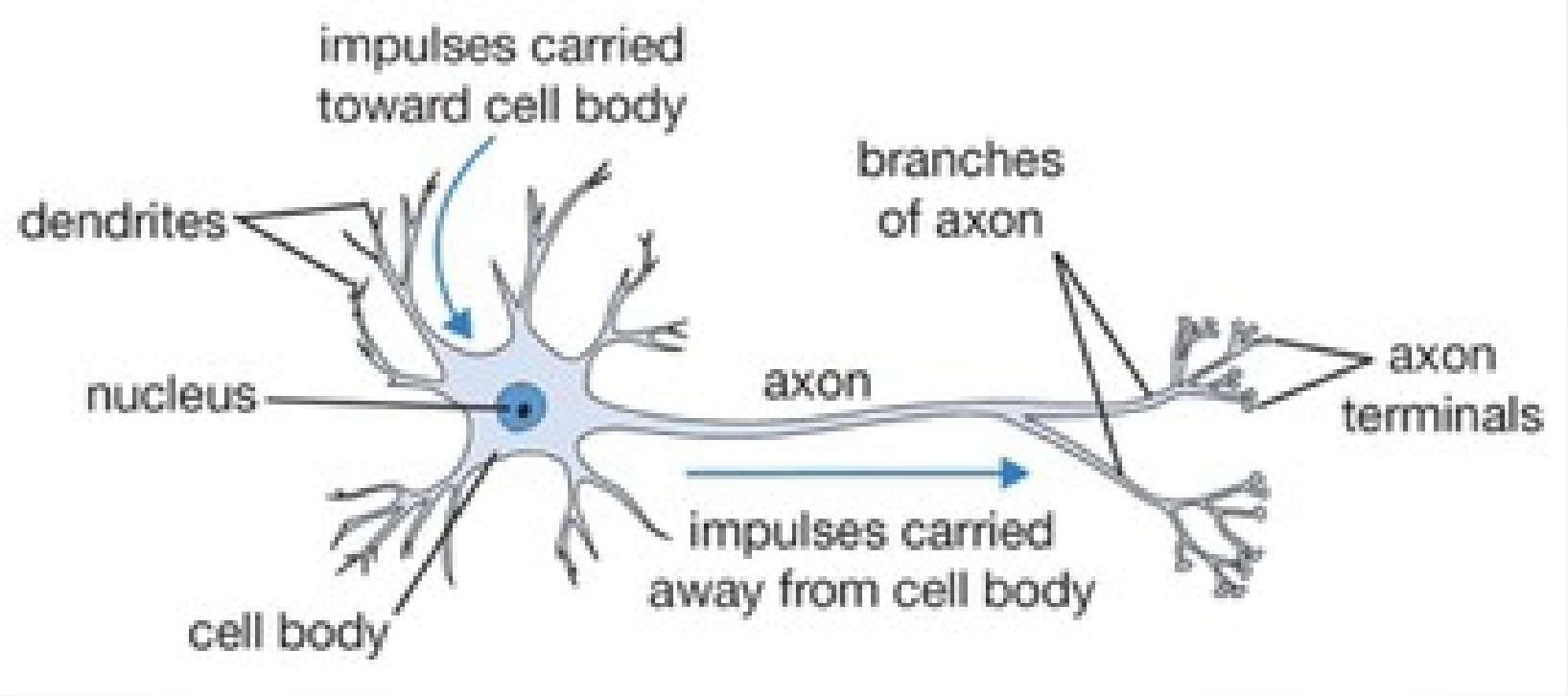


Google Colab e Kaggle

- Google Colab: un ambiente Python basato su notebook
  - https://colab.research.google.com/
  - gratuito (è richiesto solo un account Google)
  - accesso limitato a GPU e TPU
- Kaggle: una comunità dedicata al machine learning
  - https://www.kaggle.com/
  - corsi on-line di machine learning
  - competizioni con premi

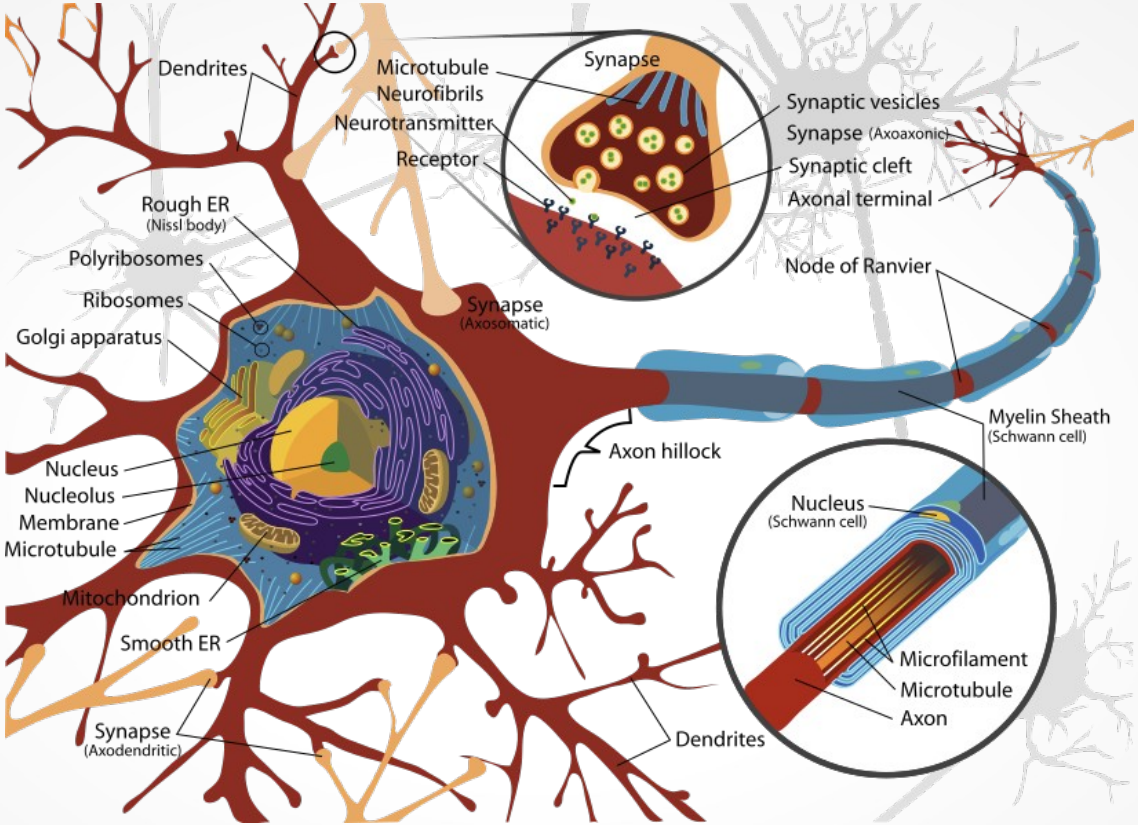
Reti neurali

Il neurone biologico



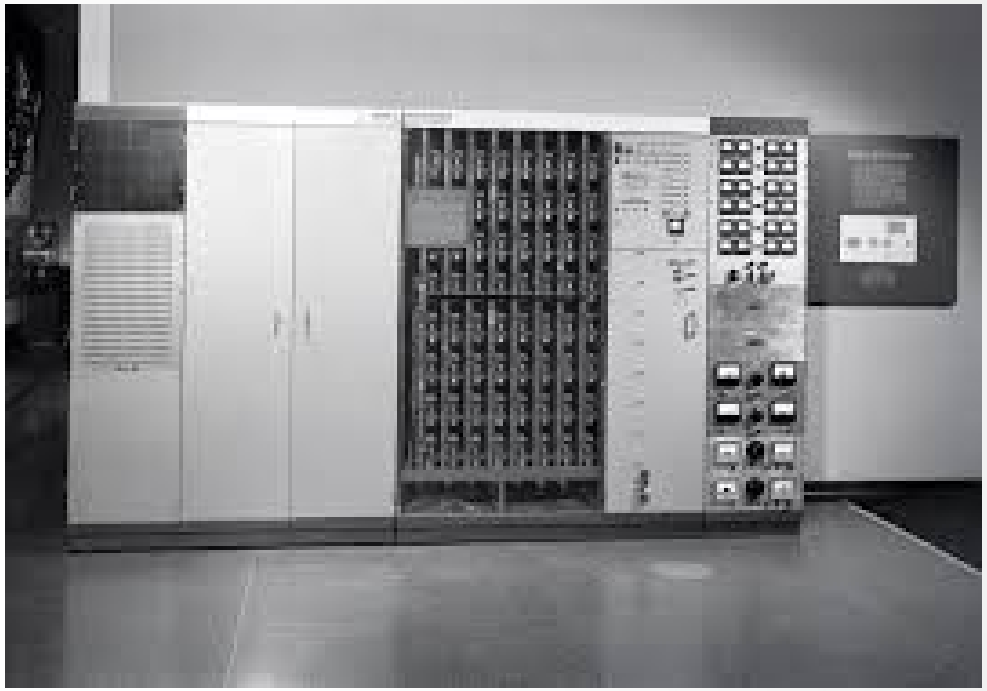
- Il neurone somma i segnali provenienti dai dendriti e “spara” un impulso dagli assoni quando si supera una certa soglia.

Il vero neurone biologico

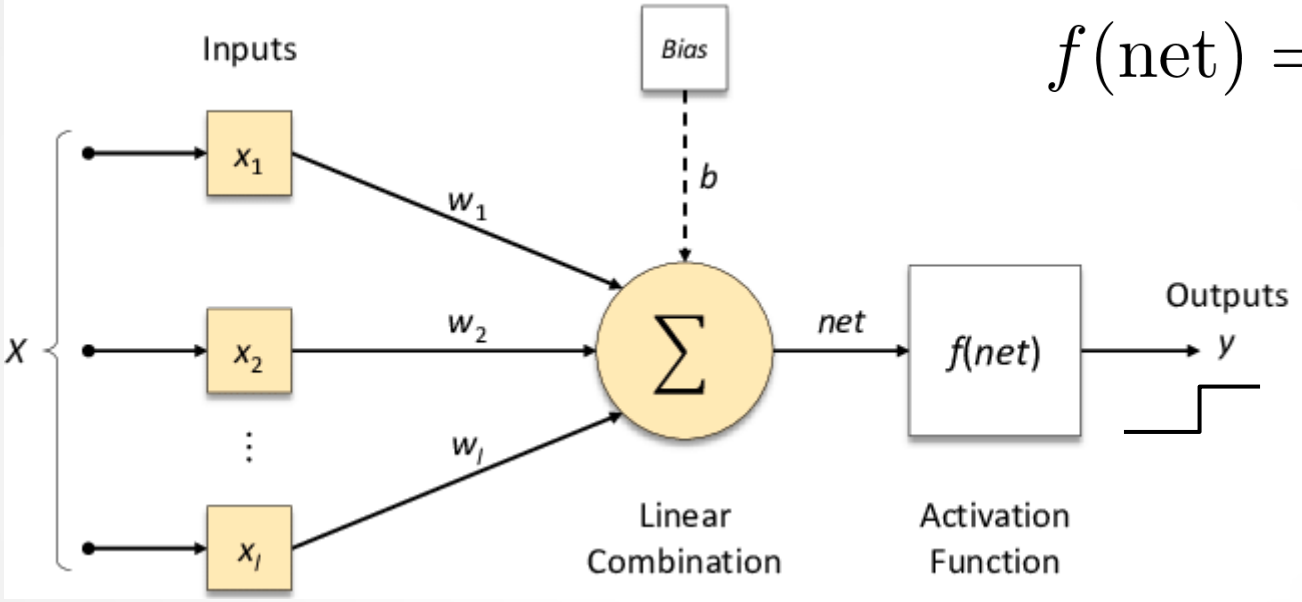


Nascita del percettrone

- 1957, Perceptron, Frank Rosenblatt: la prima *rete neurale artificiale*
- Il New York Times dell’8 luglio 1958 scrive:
  - *The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.*



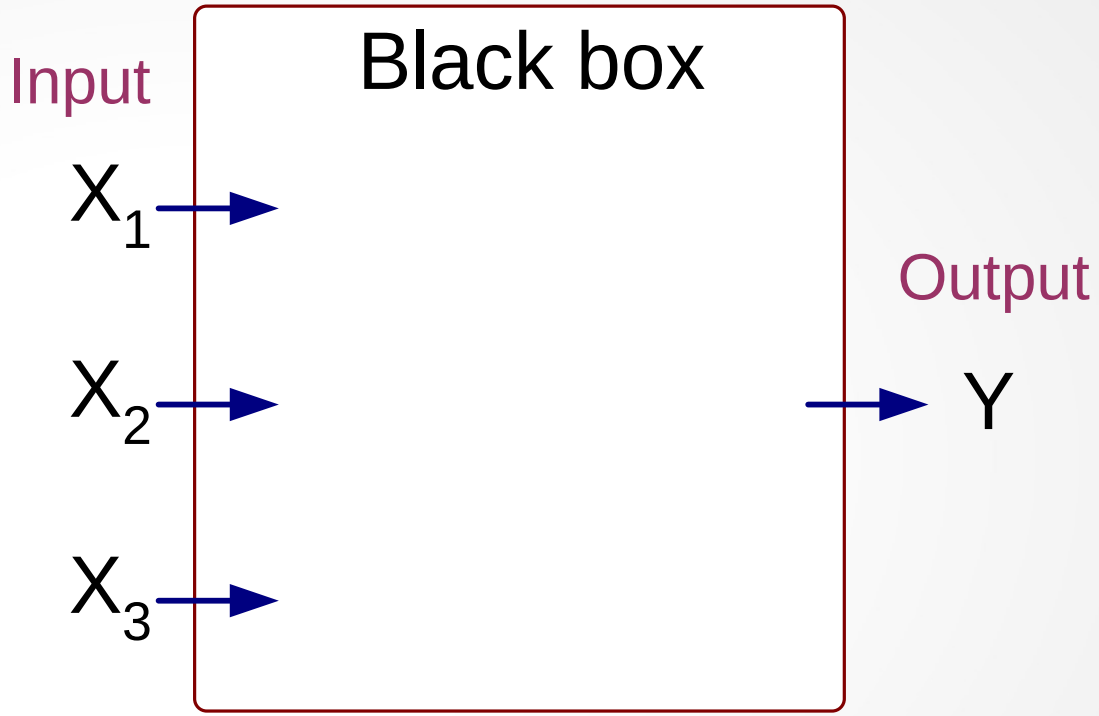
Il percettrone



$$f(\text{net}) = \begin{cases} 1 & \text{if net} > 0 \\ -1 & \text{otherwise} \end{cases}$$

Percettrone: un esempio (1)

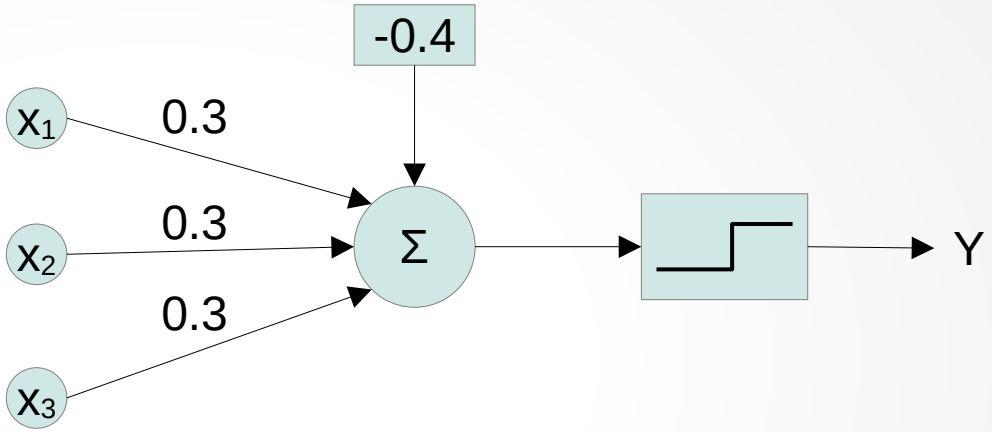
X_1	X_2	X_3	Y
1	0	0	-1
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	-1
0	1	0	-1
0	1	1	1
0	0	0	-1



Il risultato è 1 se almeno due dei tre attributi X sono ad 1, altrimenti è -1

Percettrone: un esempio (2)

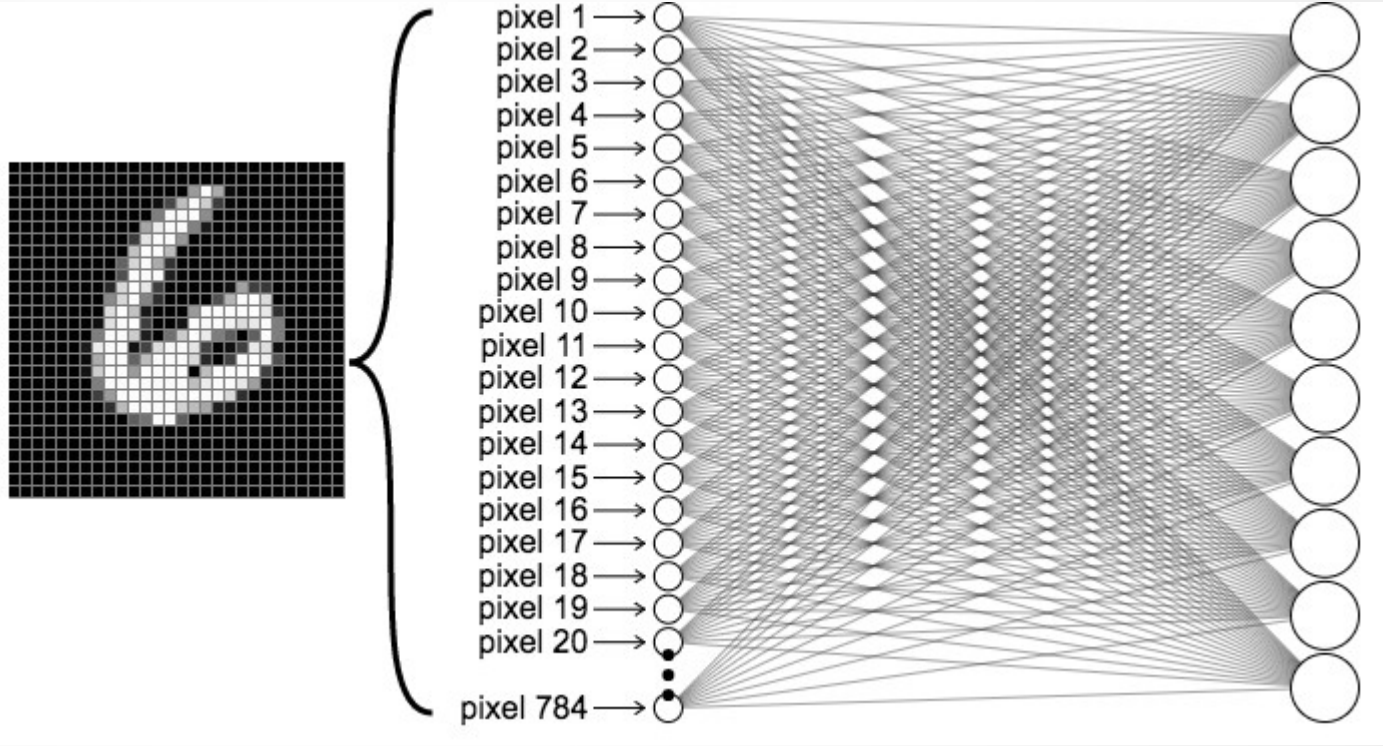
X_1	X_2	X_3	Y
1	0	0	-1
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	-1
0	1	0	-1
0	1	1	1
0	0	0	-1



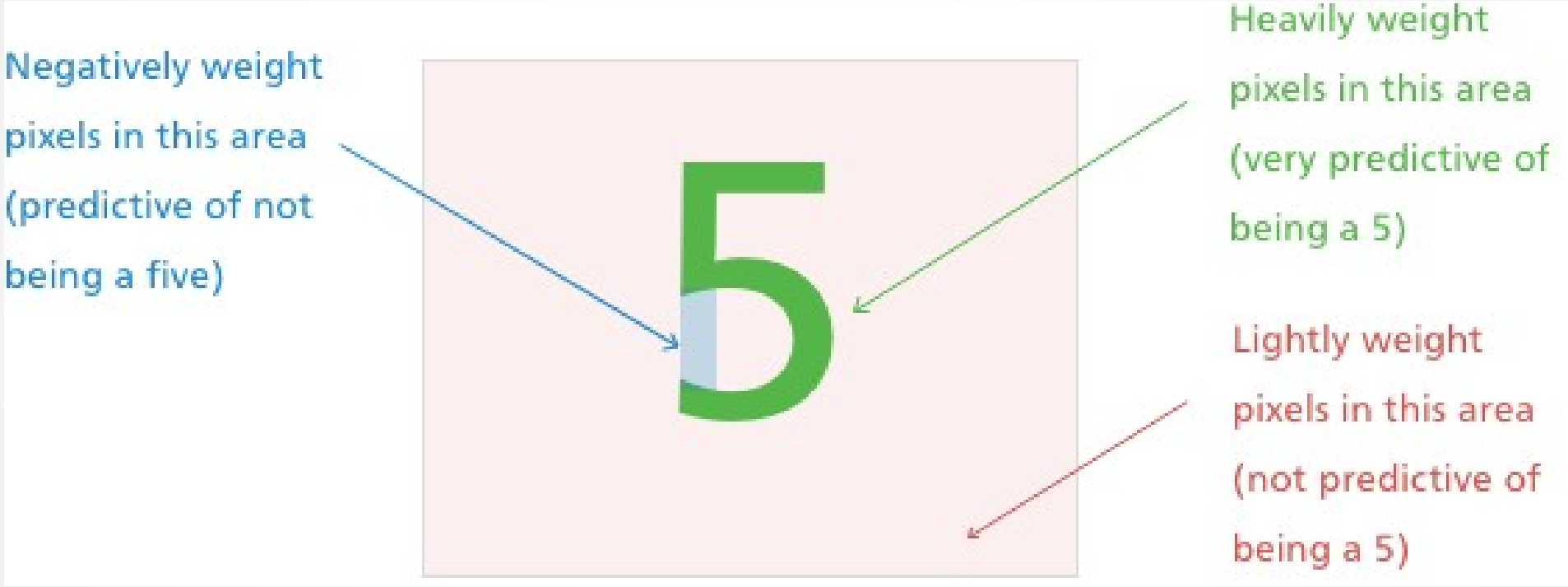
Il risultato è 1 se almeno due dei tre attributi X sono ad 1, altrimenti è -1

Percettrone per il riconoscimento di immagini (1)

- Un neurone per ogni cifra.
- w_i : uno per input/pixel.
  - Alcuni positivi, per attivarsi con la cifra cercata.
  - Alcuni negativi, per escludere altre cifre.

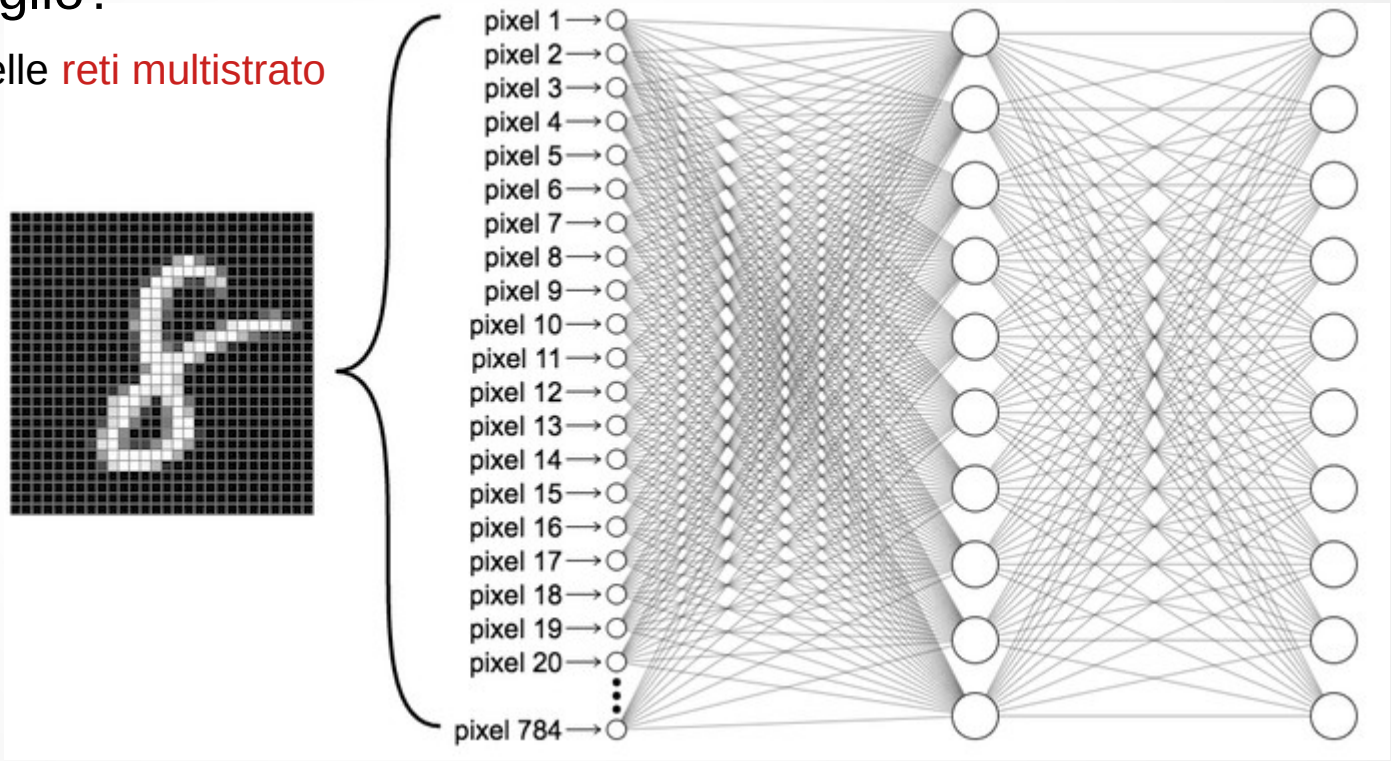


Percettrone per il riconoscimento di immagini (2)



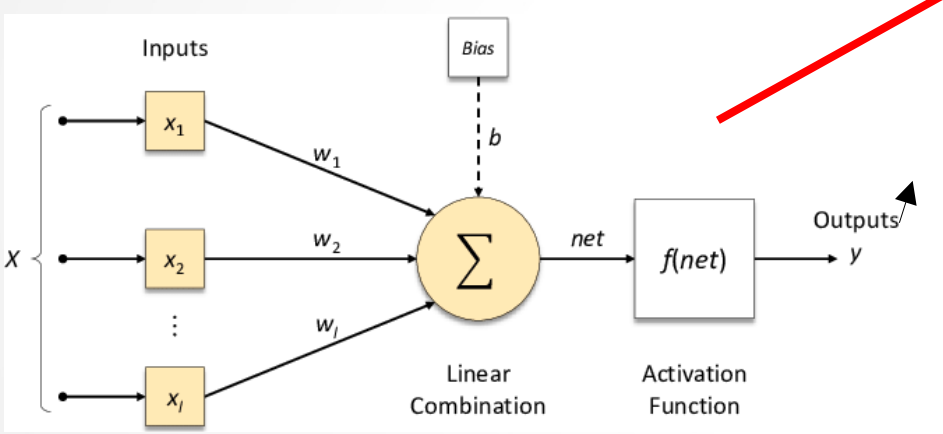
Quanto funziona in pratica?

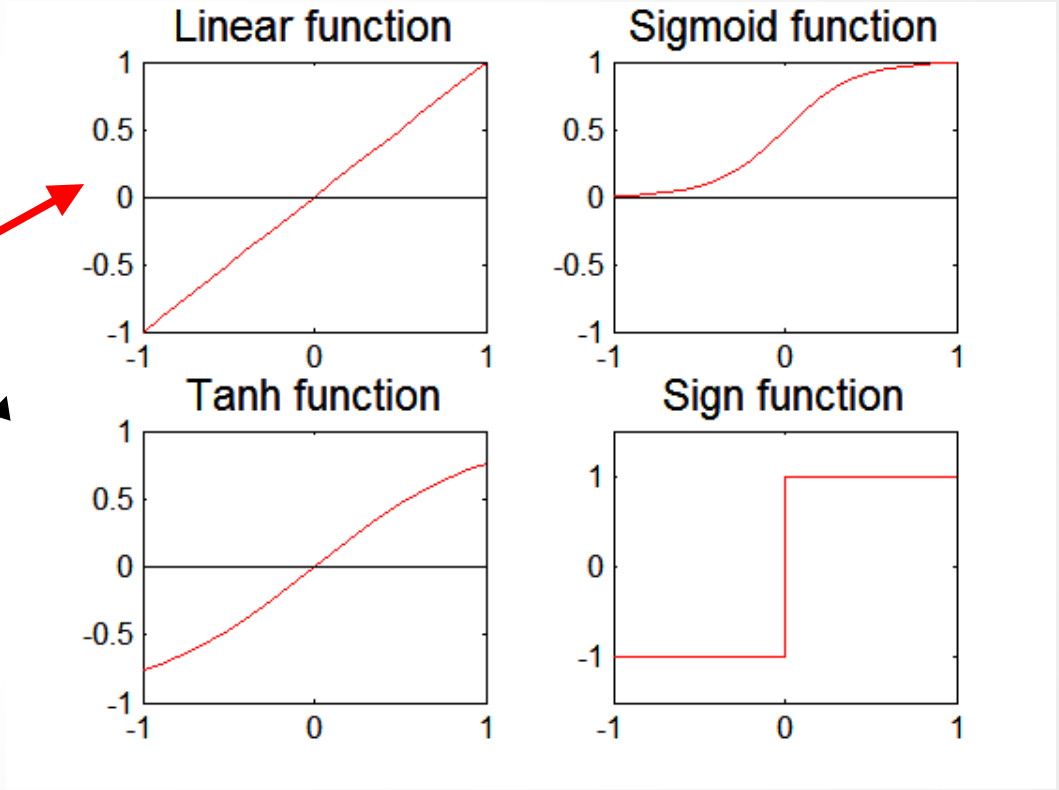
- Nel riconoscimento delle immagini MNIST, il tasso di errore è del 12%
- Si può fare di meglio?
  - Sì, ma servono delle reti multistrato



Funzioni di attivazione

- Nelle reti multistrato, la funzione di attivazione a scalino è spesso rimpiazzata da altre
- Funzioni di attivazioni più comuni:



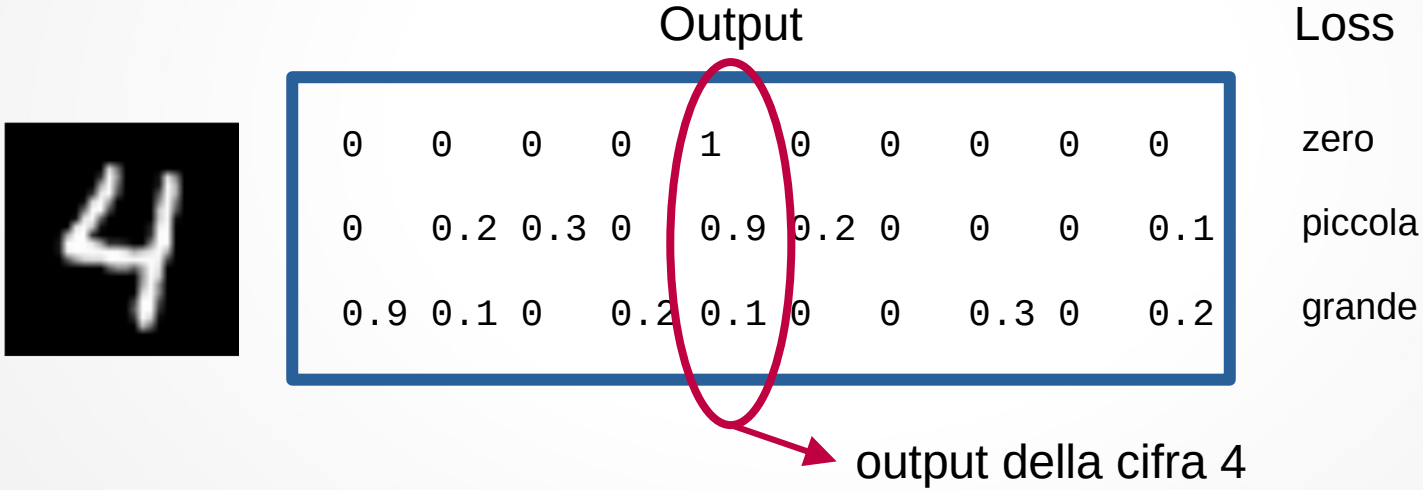


Apprendimento automatico dei pesi (1)

- I dettagli sono un po' complessi, ma si tratta di un caso speciale di un metodo generale noto come "discesa del gradiente"
- Definizioni
  - $\mathbf{x}_i$ è il vettore degli input della i-esima istanza (immagine i-esima)
  - y_i è il risultato da associare alla i-esima istanza (cifra dell'immagine i-esima)
  - $\widehat{y_i} = f(\mathbf{w}_i, \mathbf{x}_i)$ è il risulato della rete neurale per la i-esima immagine
  - Una funzione di errore (*Loss*) determina la distanza tra il risultato desiderati e quello ottenuto dalla rete

Esempio funzione di loss

- Tornando al nostro esempio MNIST, supponiamo la rete neurale abbia 10 output corrispondenti alle 10 cifre
- La cifra predetta dalla rete è quella che ha l'output più alto
- Allora la funzione di loss dovrebbe comportarsi così



Apprendimento automatico dei pesi (2)

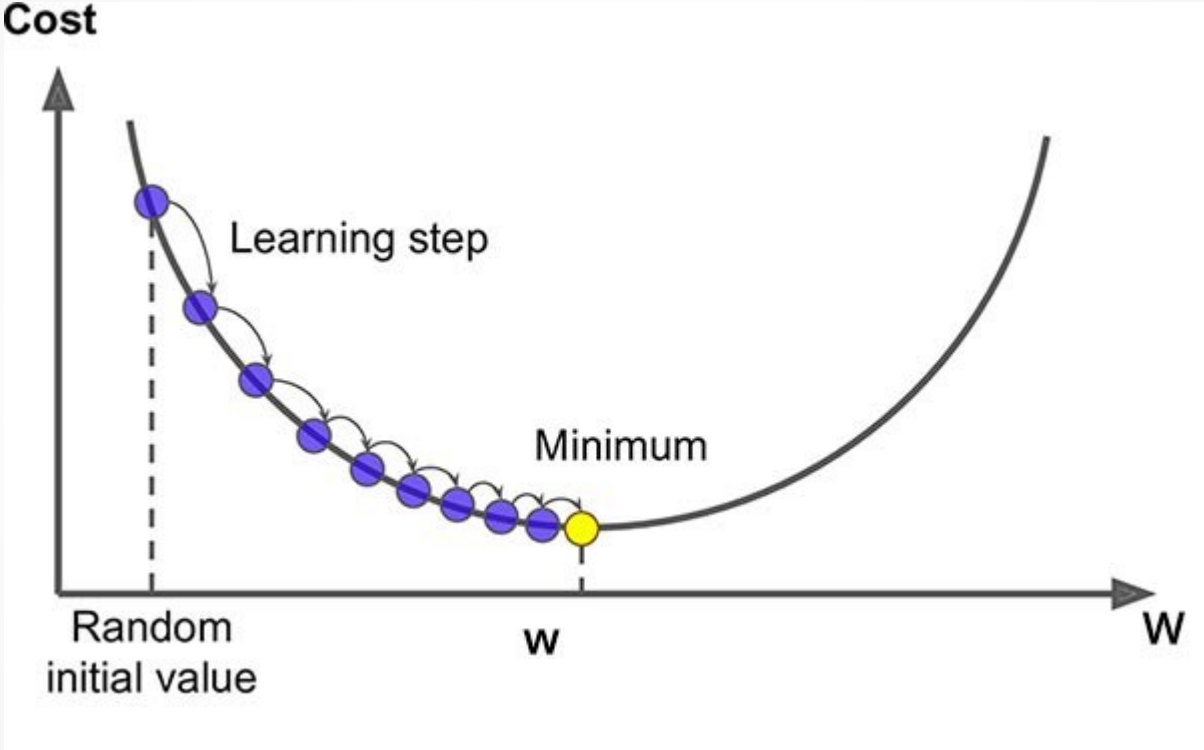
- Definizioni
  - L’errore complessivo della rete è la somma di tutti gli errori per le singole istanze di input

$$E = \sum_{i=1}^{N} Loss(y_i, \hat{y}_i)$$

  - L’errore dipende dai pesi (**w**) della rete. Dobbiamo cambiare i pesi in modo da rendere minimo l’errore!
  - Non si può risolvere in maniera analitica, la formula completa dell’errore è troppo complessa.
  - Si usano metodi di approssimazione, come la **discesa del gradiente**

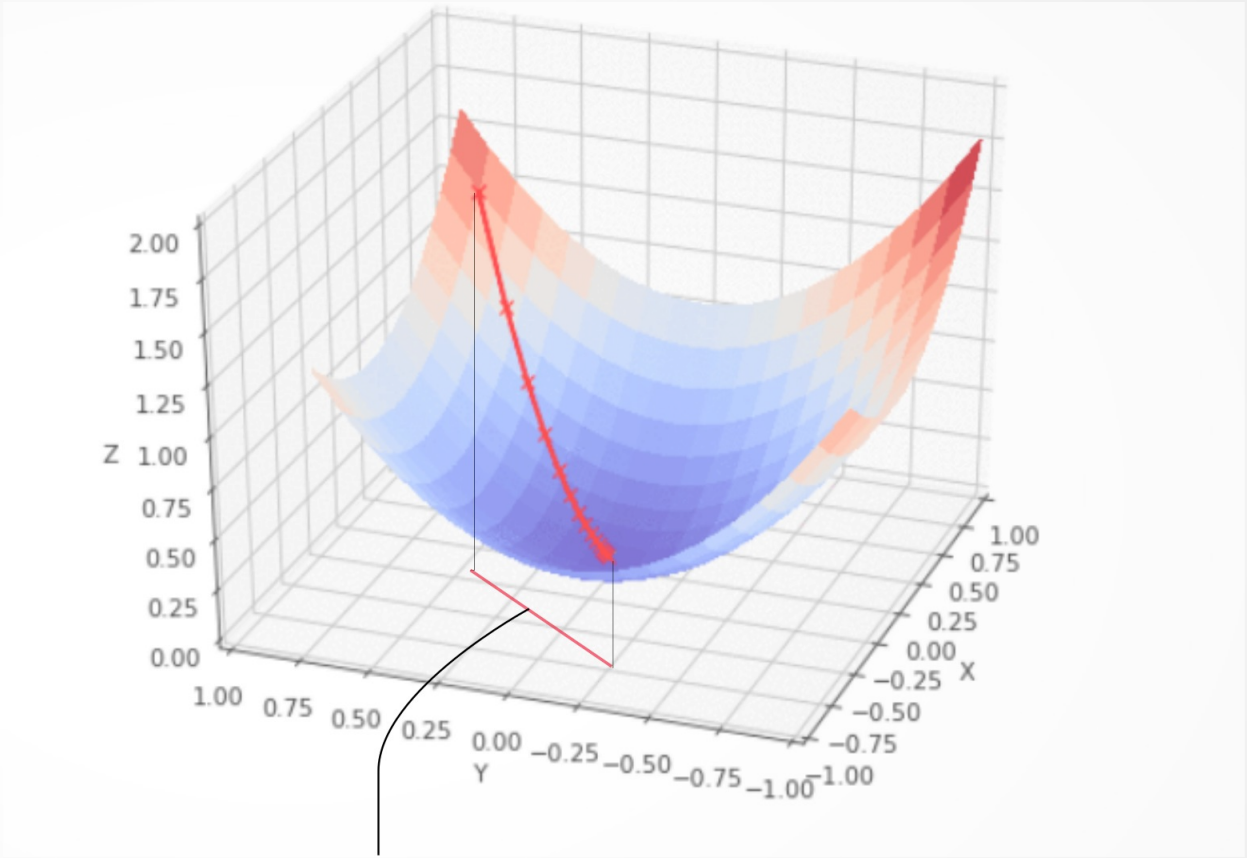
Discesa del gradiente (1)

- Supponiamo che abbiamo un solo peso nella nostra rete



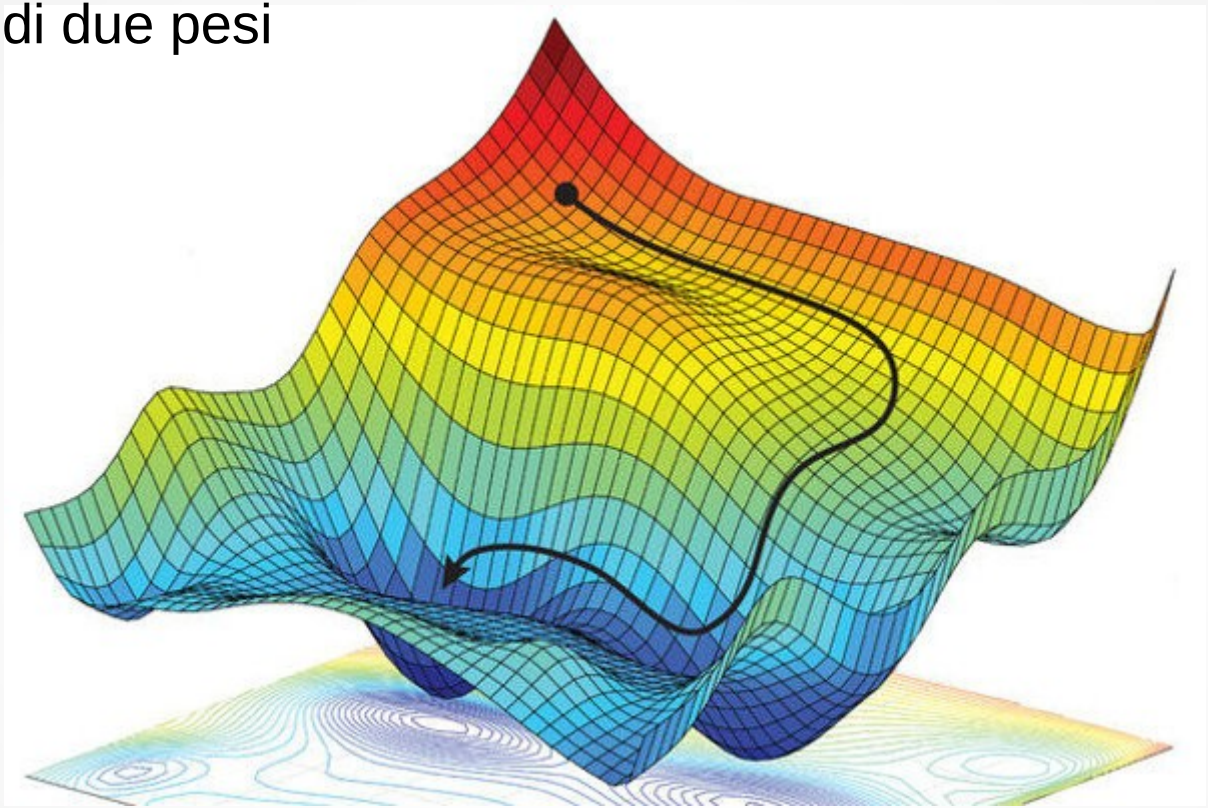
Discesa del gradiente (2)

- Nel caso di due pesi



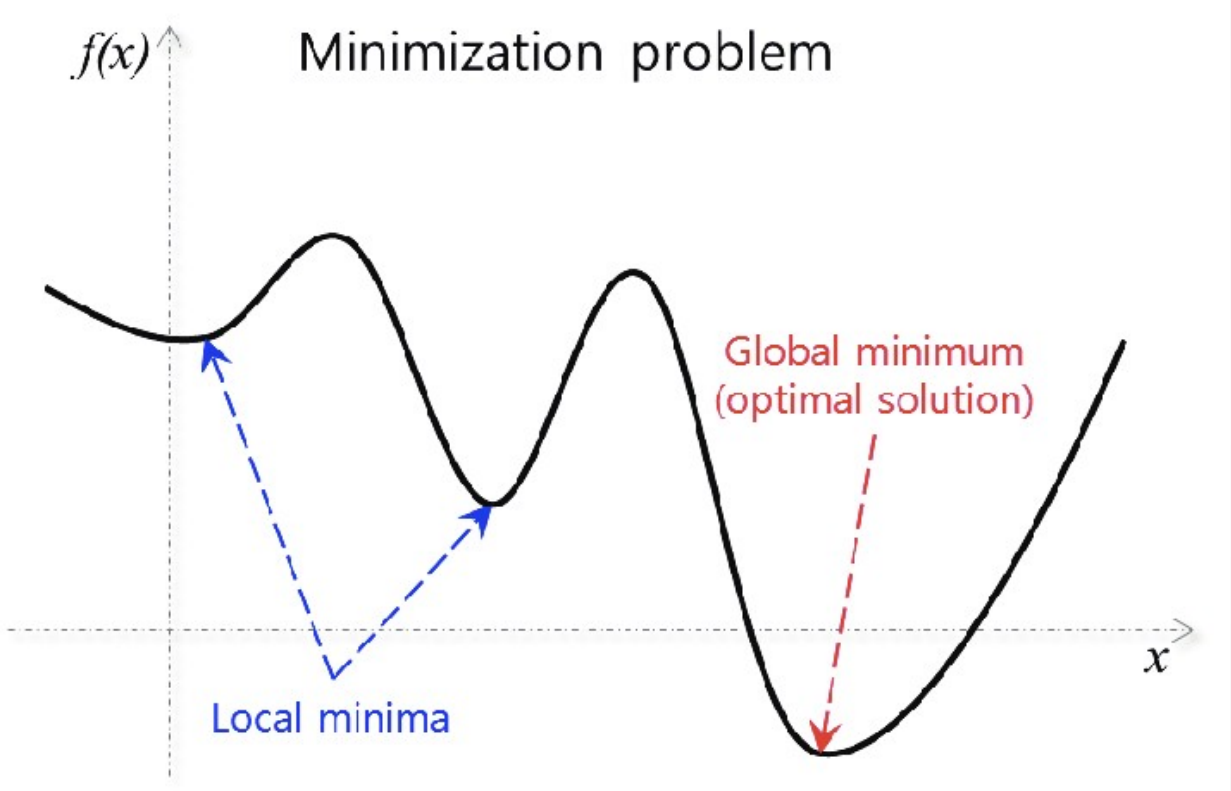
Discesa del gradiente (3)

- Nel caso di due pesi



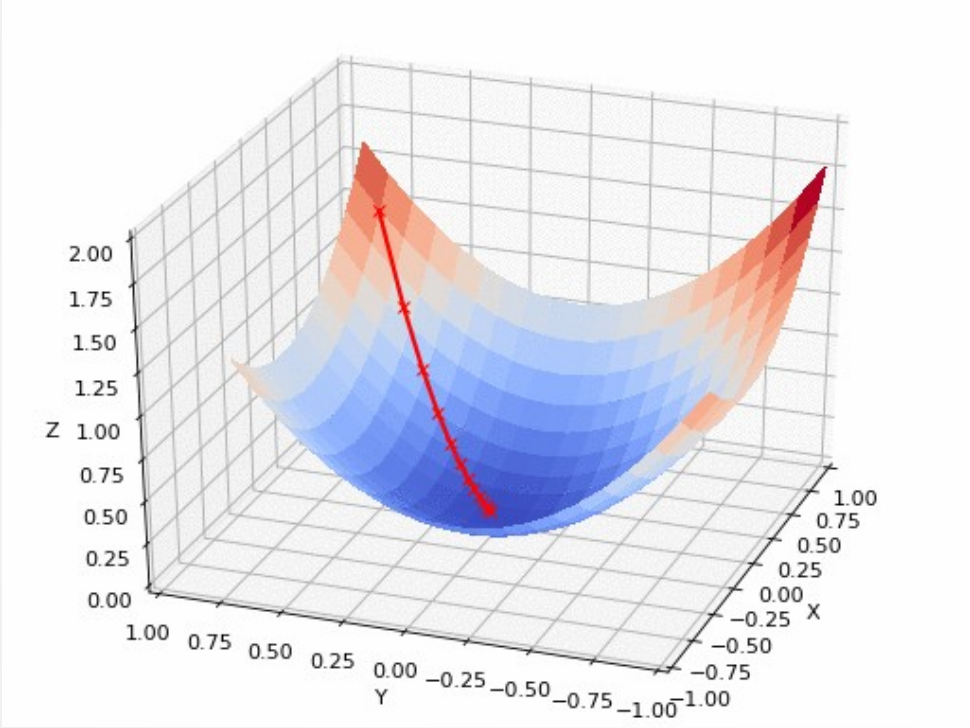
Problema: minimi locali

- Talvolta il metodo di discesa del gradiente rimane bloccato su minimi locali

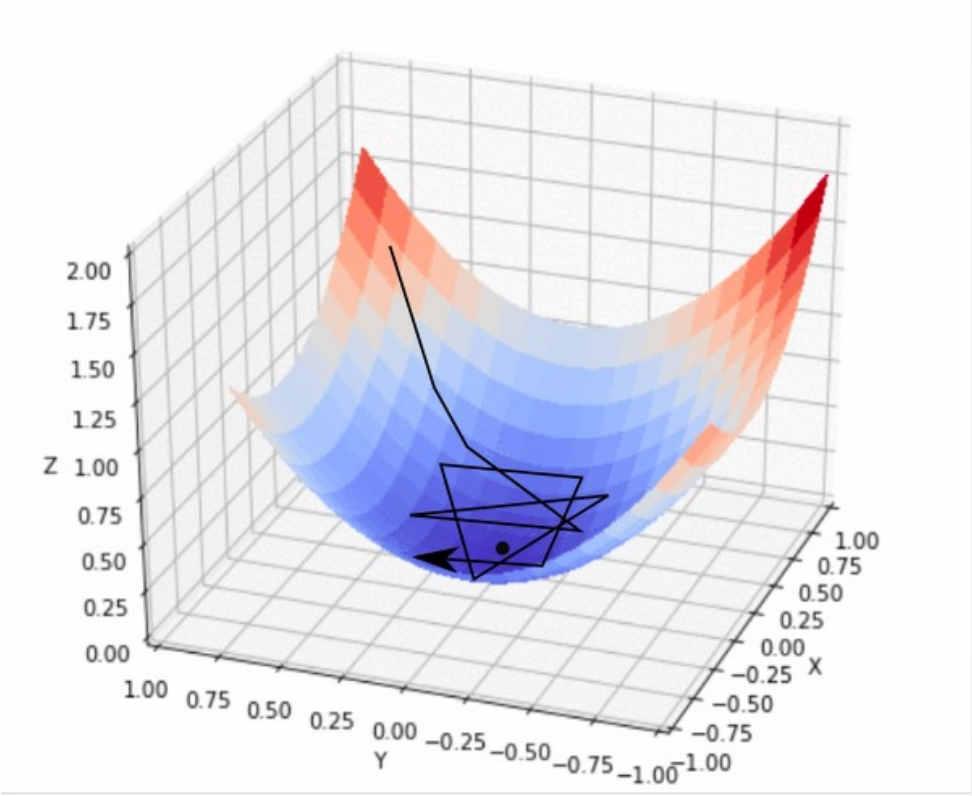


Problema: scelta del learning rate

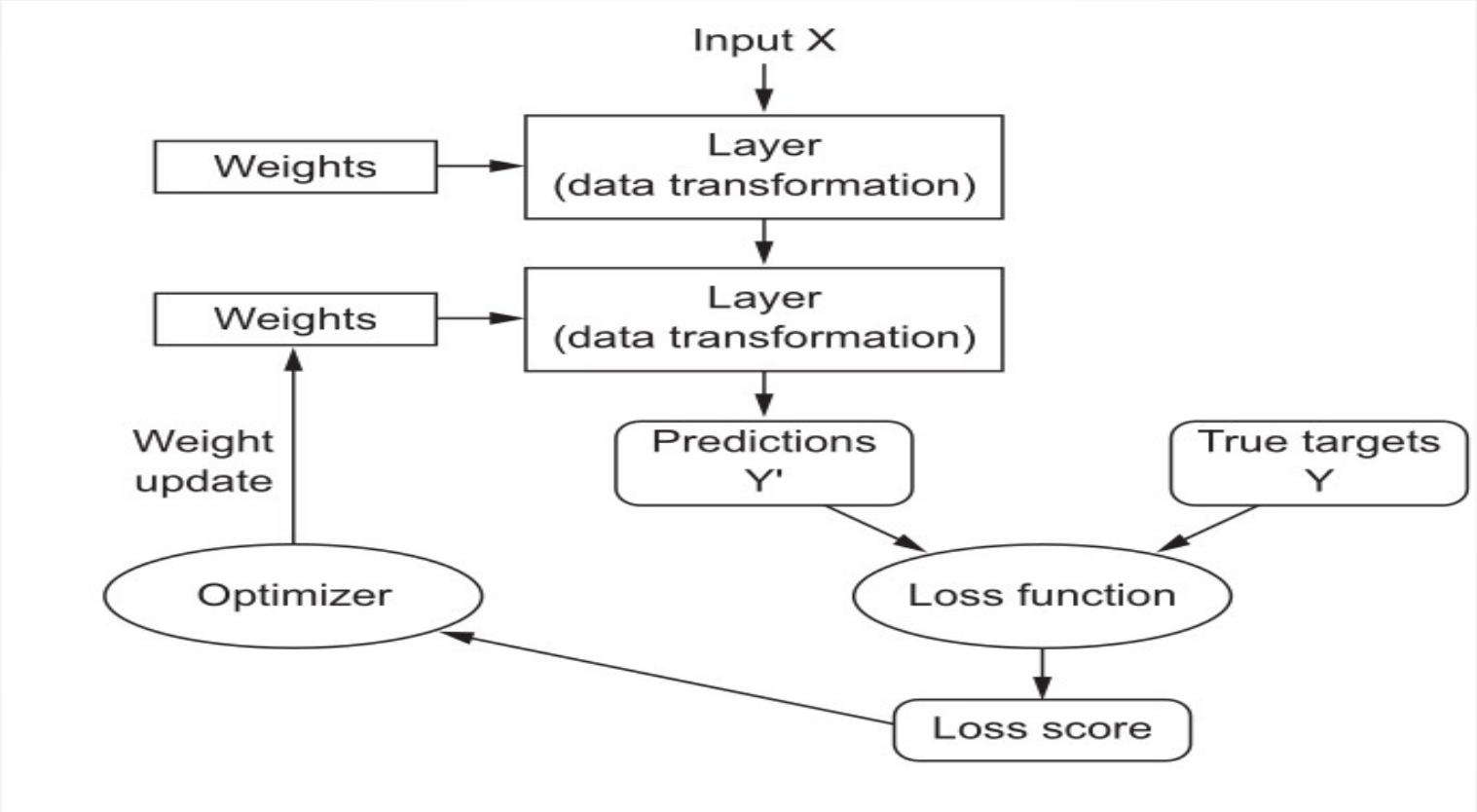
Learning rate “buono”



Learning rate troppo alto



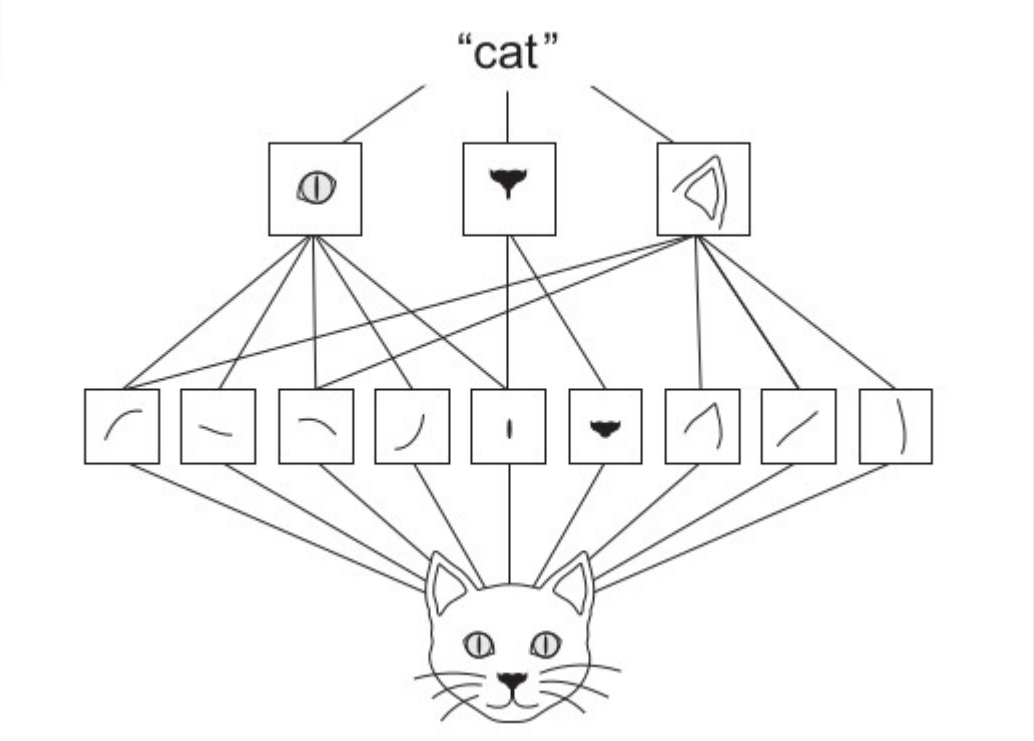
Funzionamento della fase di addestramento



Reti convoluzionali

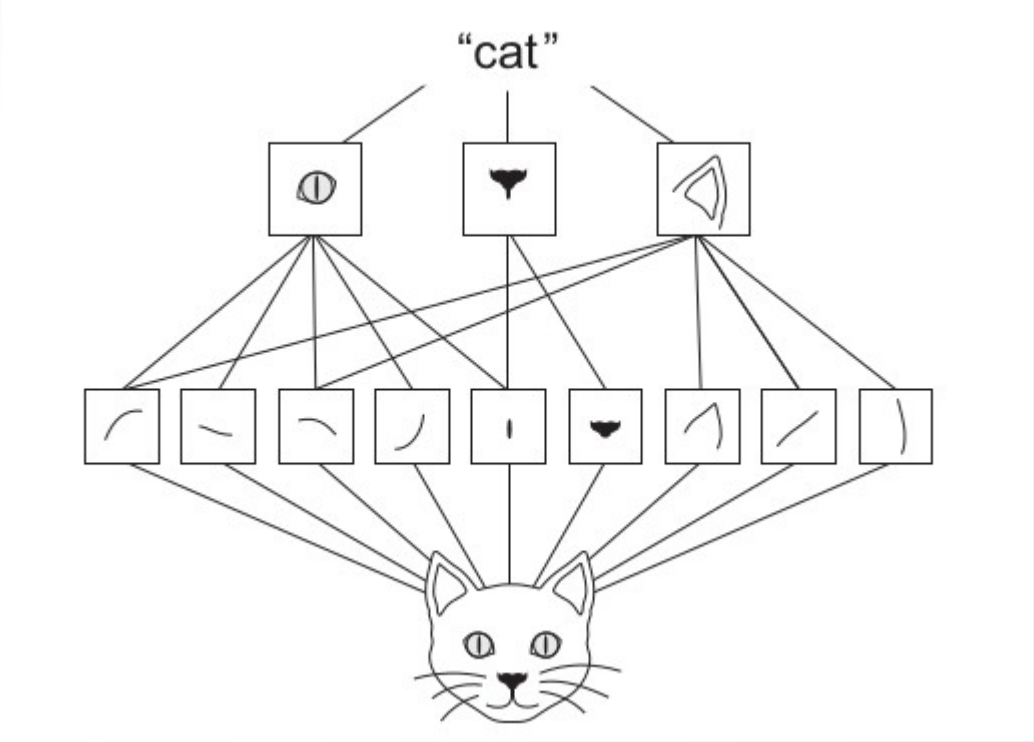
Reti convoluzionali (1)

- I neuroni di una rete neurale “standard” hanno come input tutti gli output dello strato precedente.
- Nelle reti convoluzionali, invece, ogni neurone è connesso soltanto a un piccolo sottinsieme dei neuroni dello strato precedente, secondo uno schema regolare.
- Riconoscono quindi dei *pattern* locali

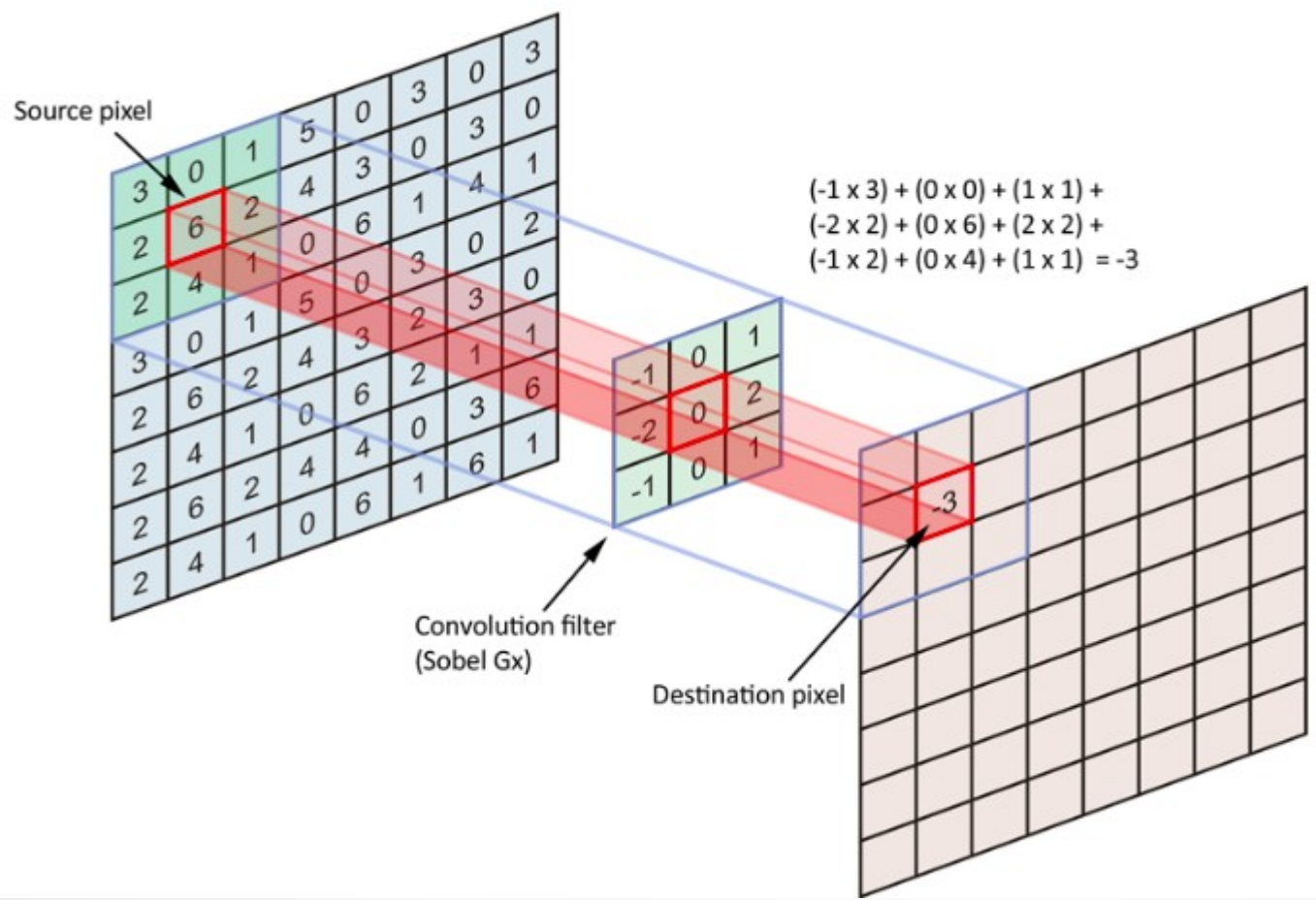


Reti convoluzionali (2)

- I pattern appresi sono invarianti per traslazioni
  - una volta imparato a riconoscere un pattern (ad esempio, una riga orizzontale) in una zona dell'immagine, sarà in grado automaticamente di riconoscere lo stesso pattern ovunque
  - questo è fondamentale nel riconoscimento delle immagini, che sono fondamentalmente invarianti per traslazioni
- Gli strati successivi costruiscono pattern più complessi partendo dai pattern degli strati precedenti



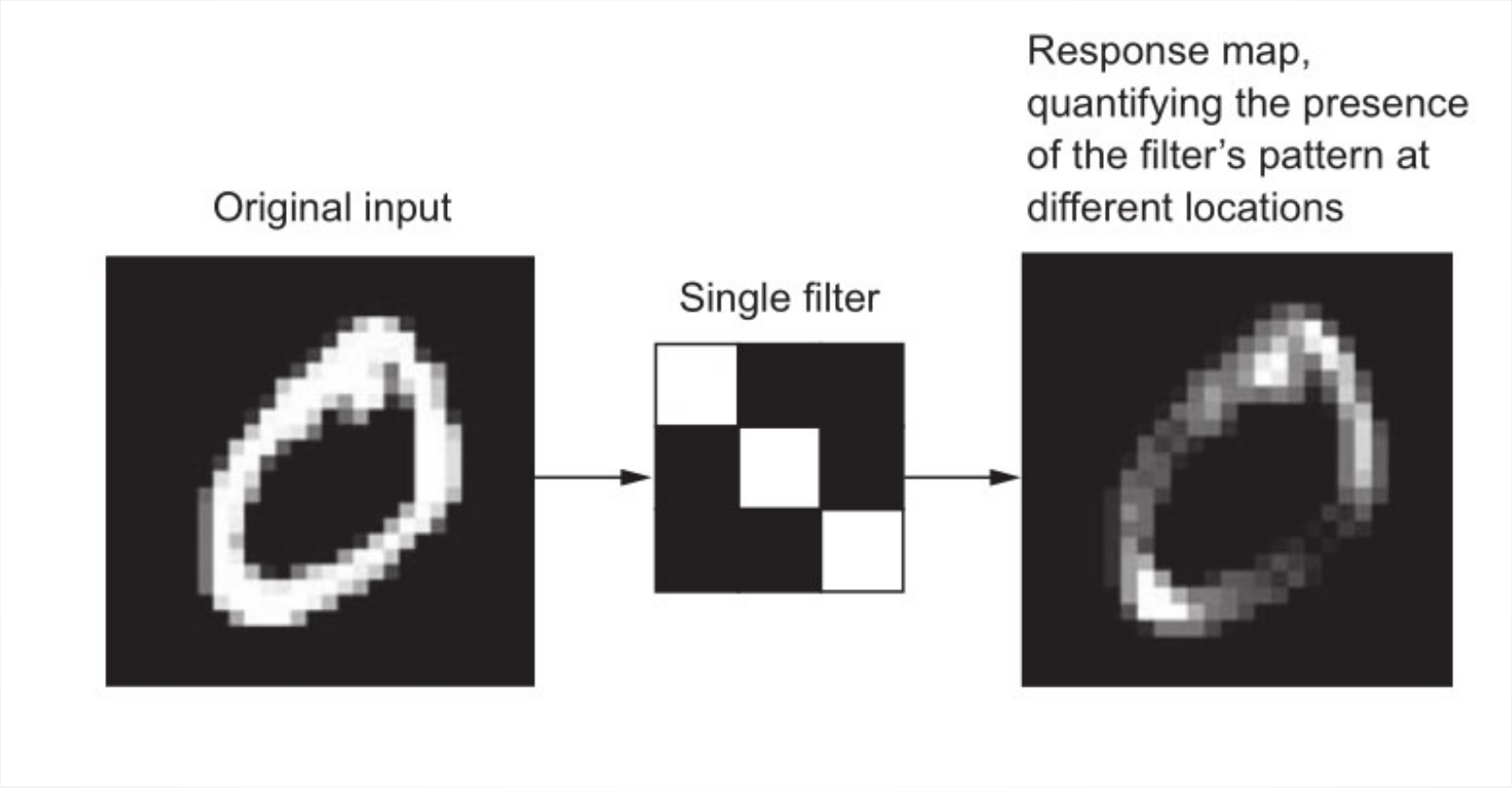
Esempio di filtro (1)



Stiamo supponendo che la funzione di attivazione sia l'identità…

Nella realtà va considerata anche quella!

Esmepio di filtro (2)



Struttura di una rete convoluzionale