Machine Learning Principles

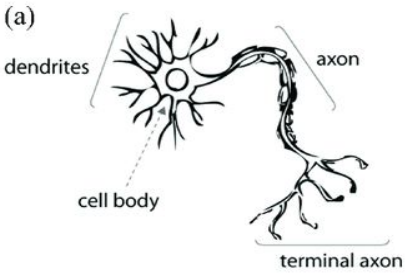
Multi Layer Perceptrons

Instructor: Tugce Gurbuz

Nov 4th, 2022

Perceptron (1958)

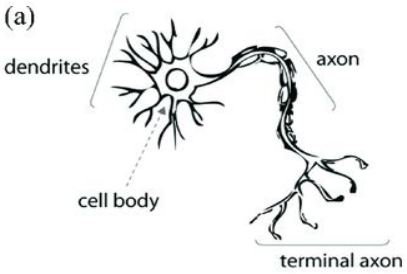
Source



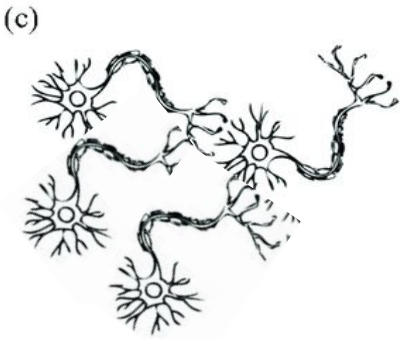
1: fire
0: not fire

Perceptron (1958)

Source



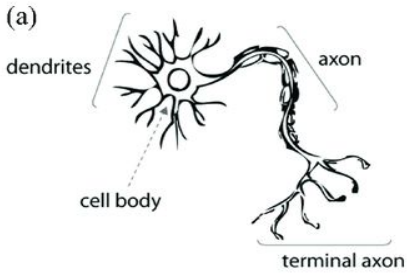
1: fire
0: not fire



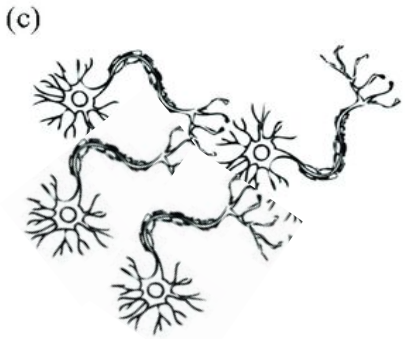
Excitatory and inhibitory connections

Perceptron (1958)

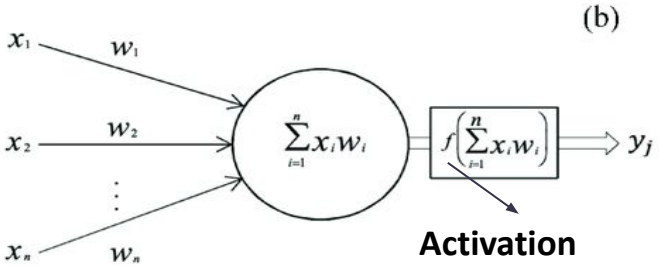
Source



1: fire
0: not fire

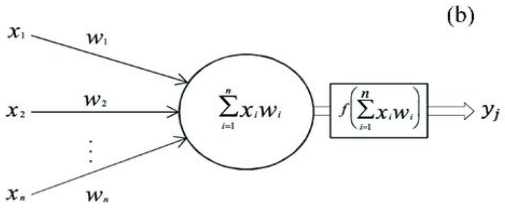


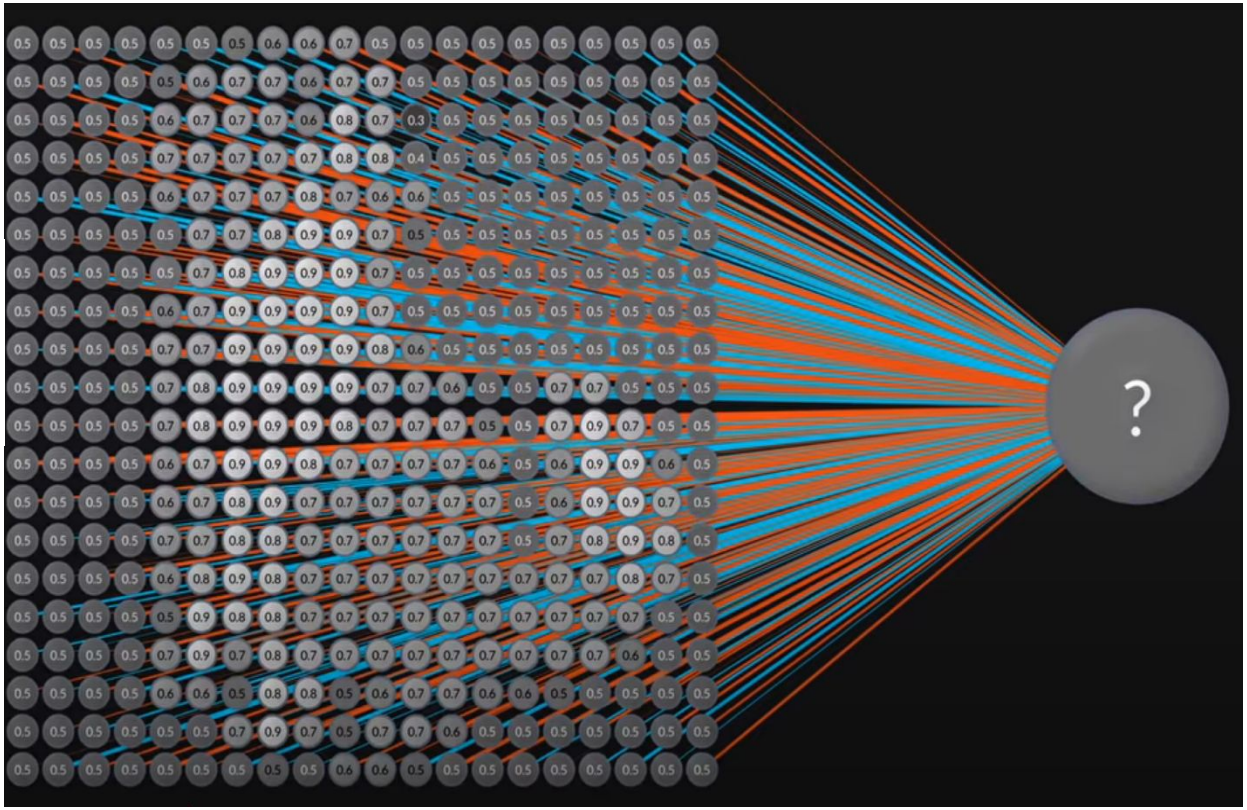
Excitatory and inhibitory connections



Activation function

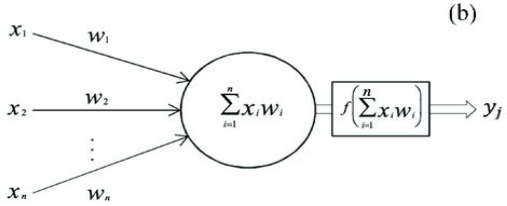
How does perceptron distinguish between 2 images?

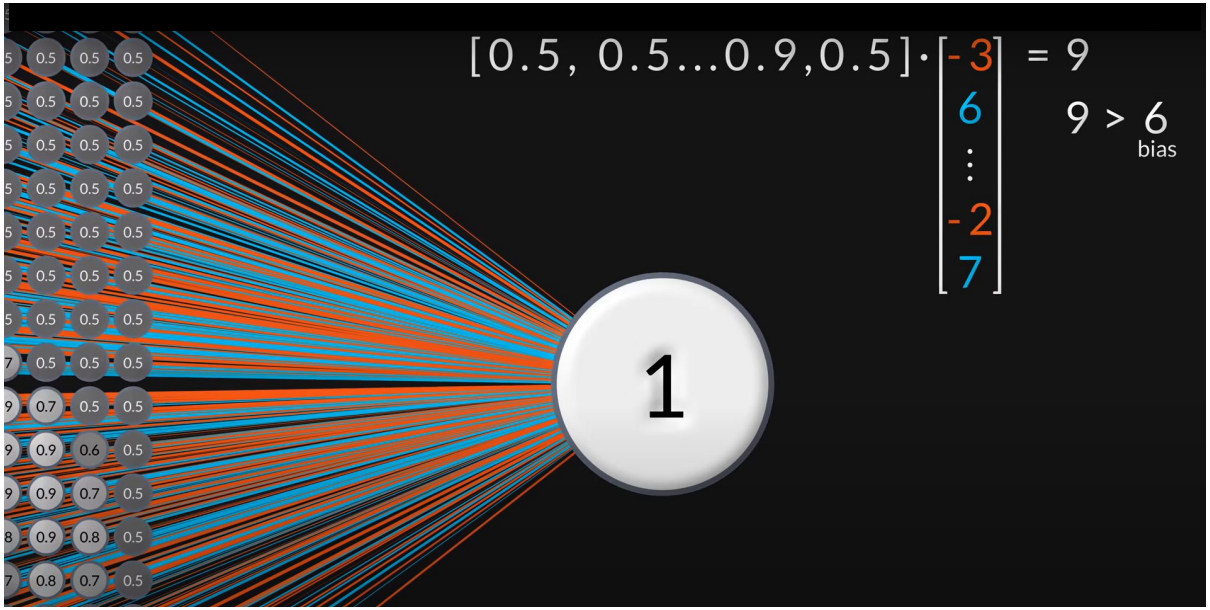




Source

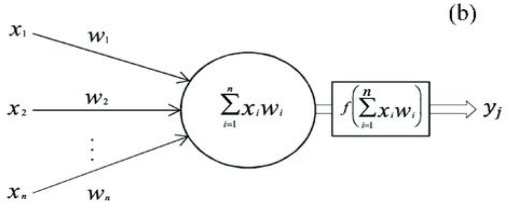
How does perceptron distinguish between 2 images?

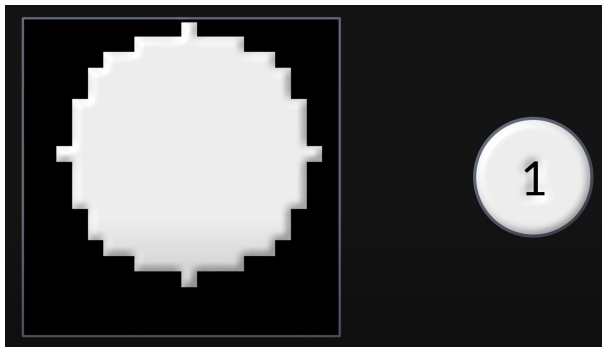




Source

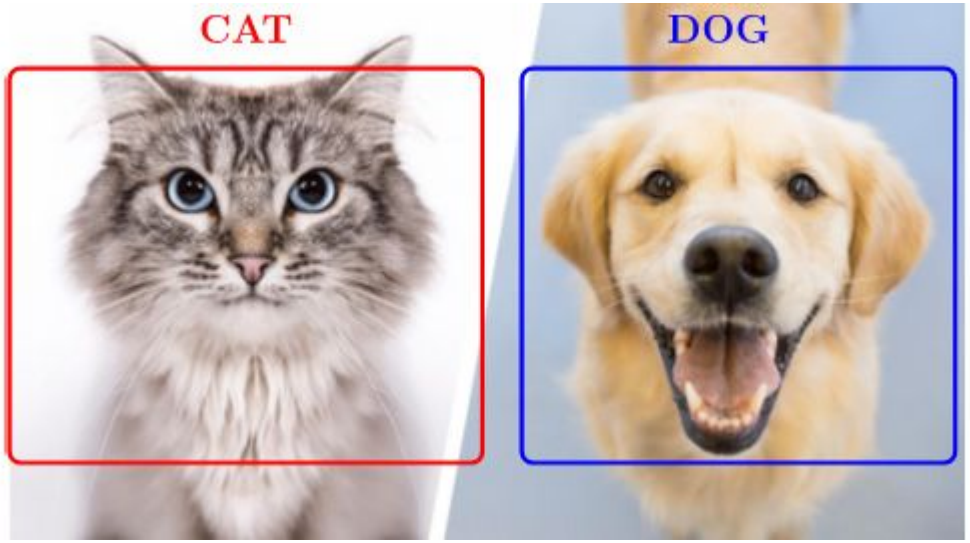
How does perceptron distinguish between 2 images?



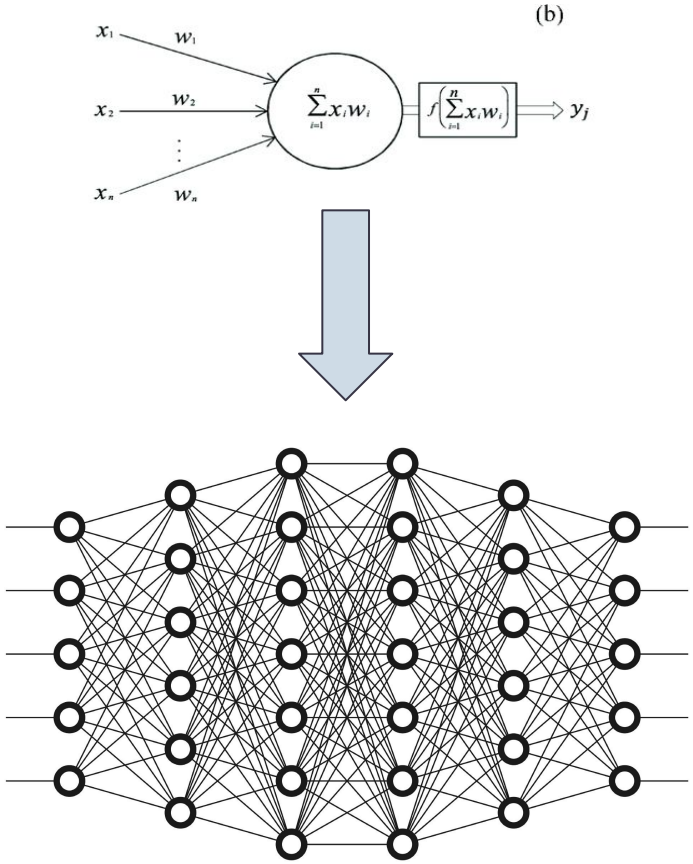


Source

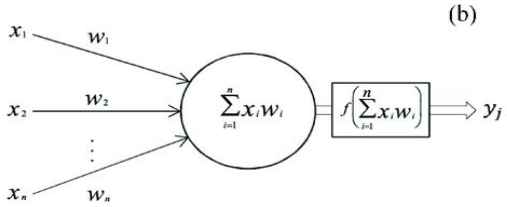
How does perceptron distinguish between 2 images?



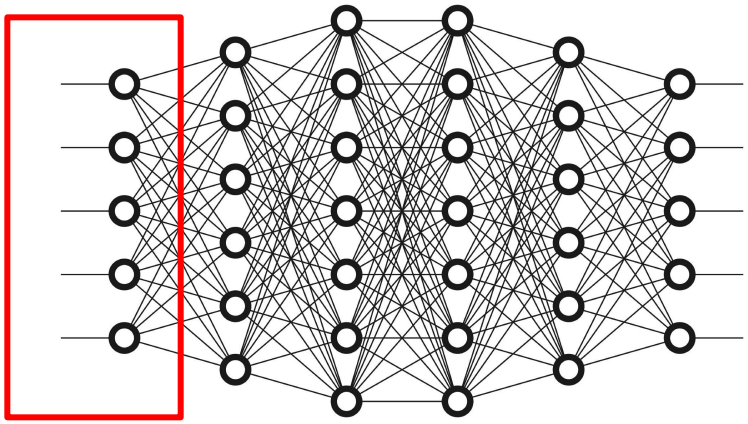
Multi Layer Perceptron



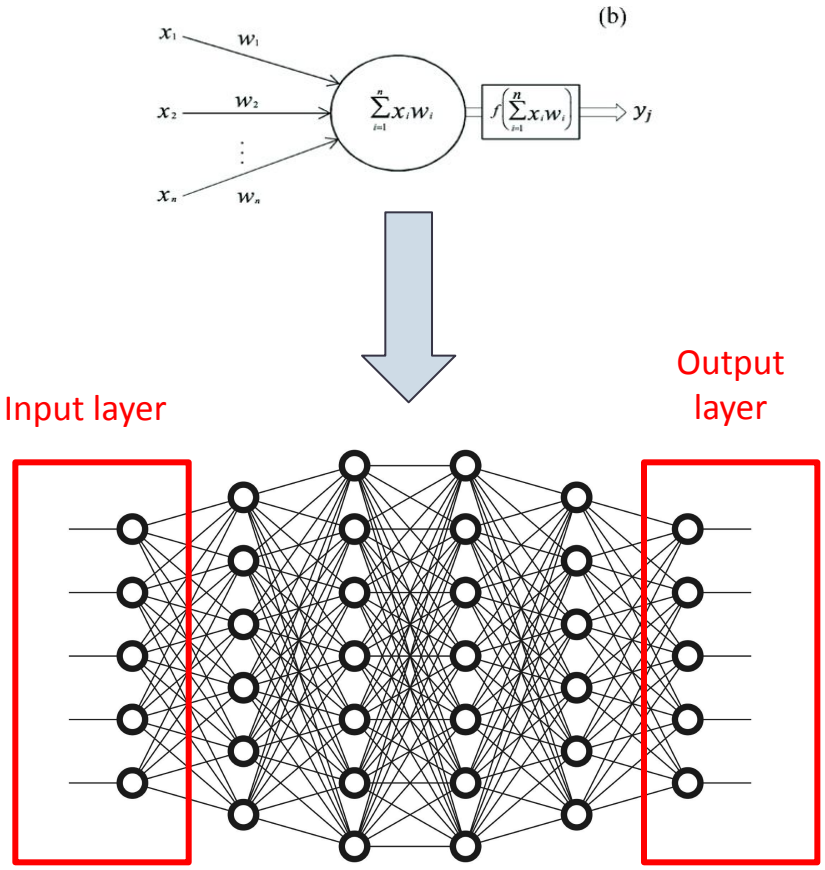
Multi Layer Perceptron



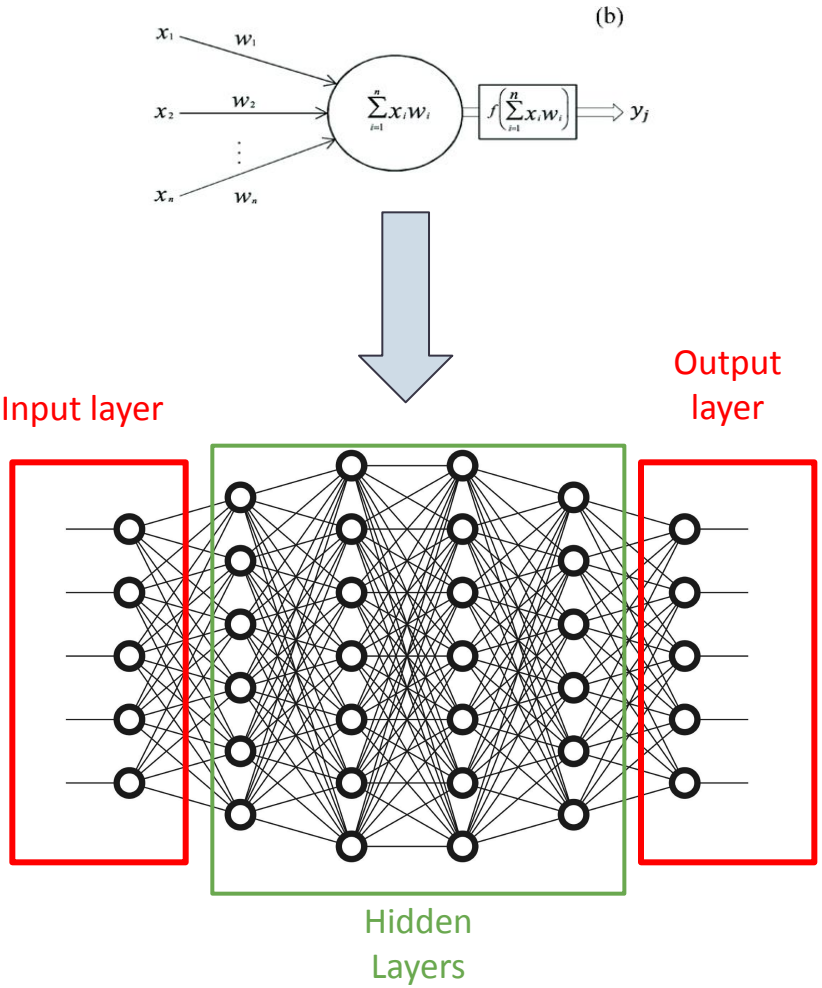
Input layer



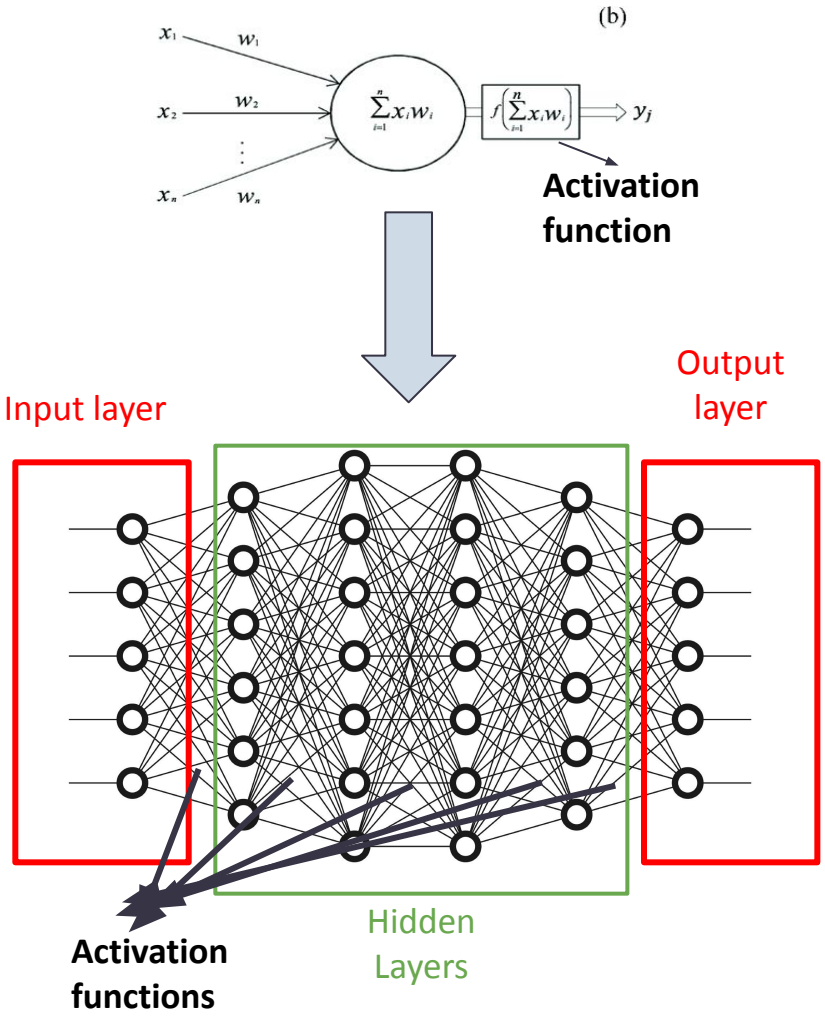
Multi Layer Perceptron



Multi Layer Perceptron



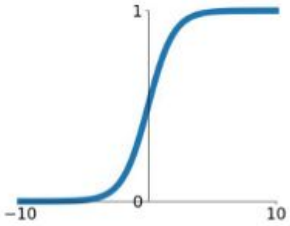
Multi Layer Perceptron



Activation Functions

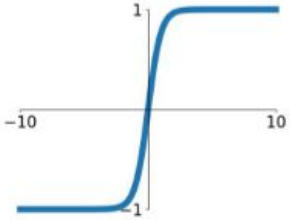
Sigmoid

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



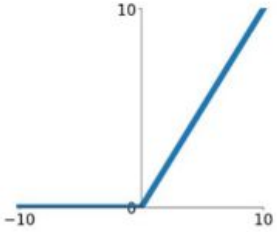
tanh

$$\tanh(x)$$



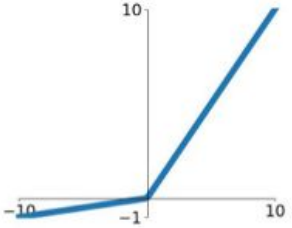
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

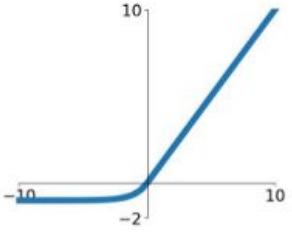


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

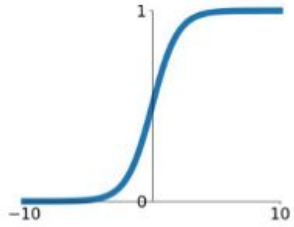
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Activation Functions

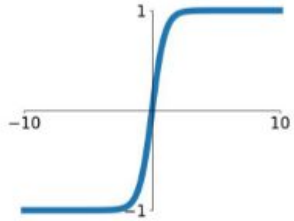
Sigmoid

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



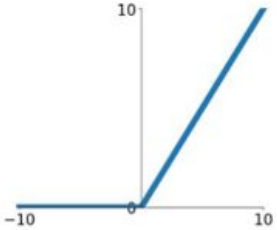
tanh

$$\tanh(x)$$



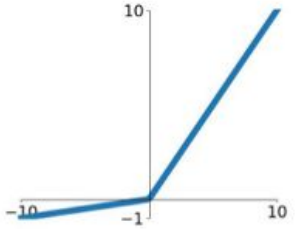
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

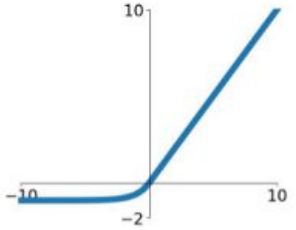


Maxout

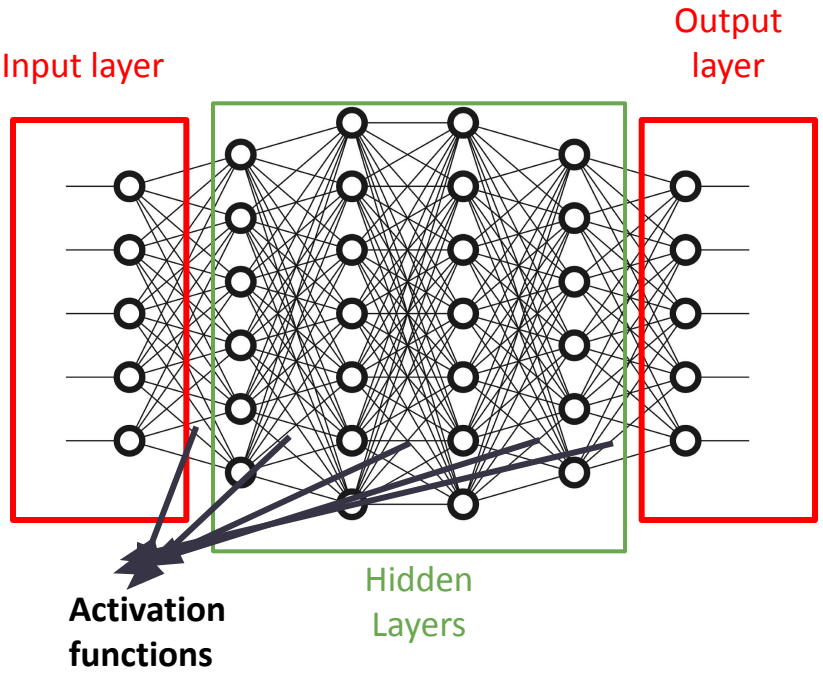
$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Let's implement our first ML model!

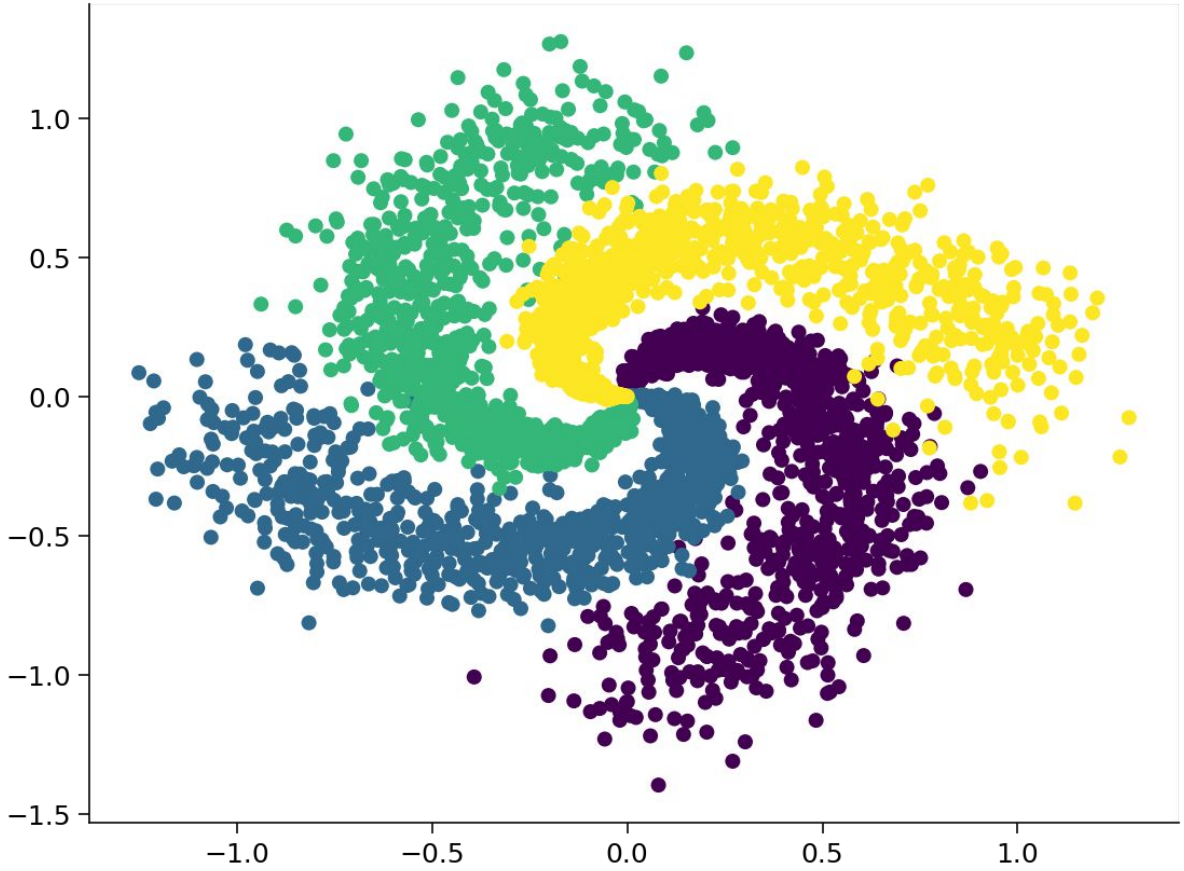


Let's go to our first tutorial and implement our first ML model!!

How do we teach MLP to classify?

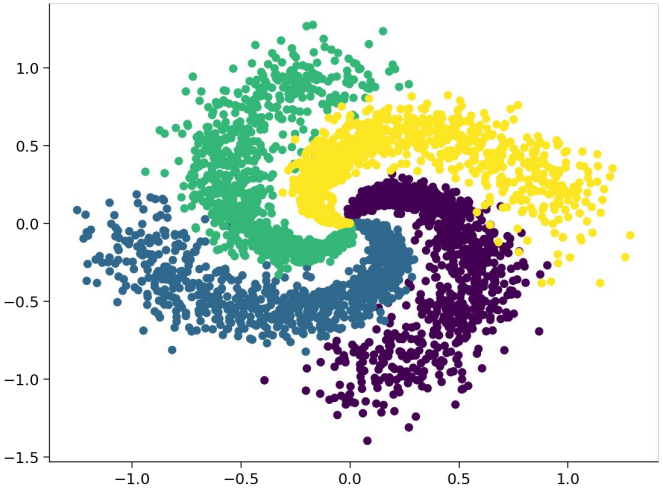
How do we teach MLP to classify?

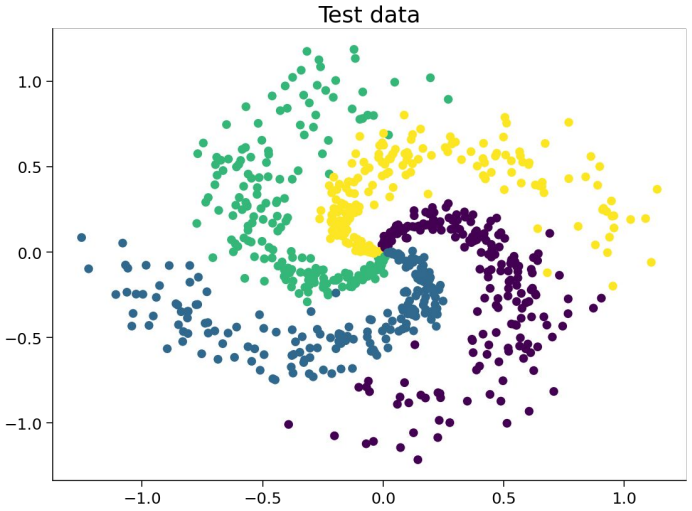
Dataset



How do we teach MLP to classify?

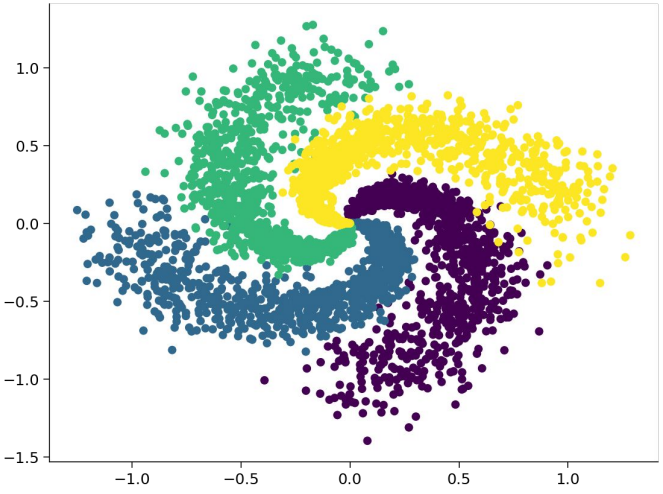
Dataset

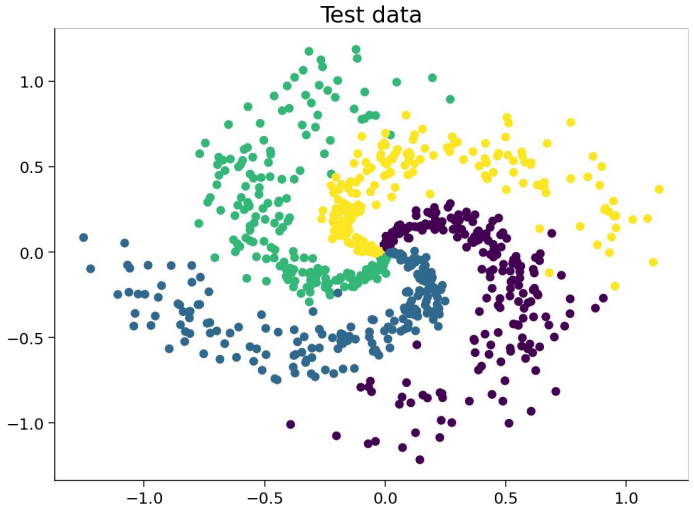




How do we teach MLP to classify?

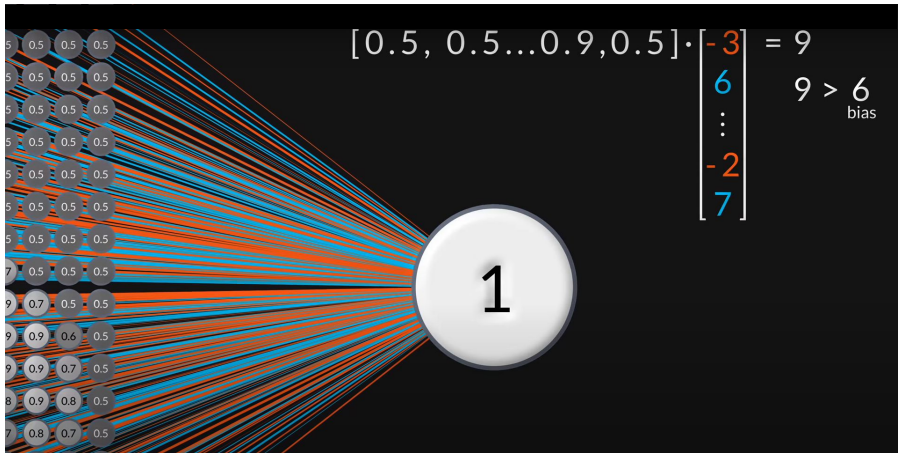
Dataset

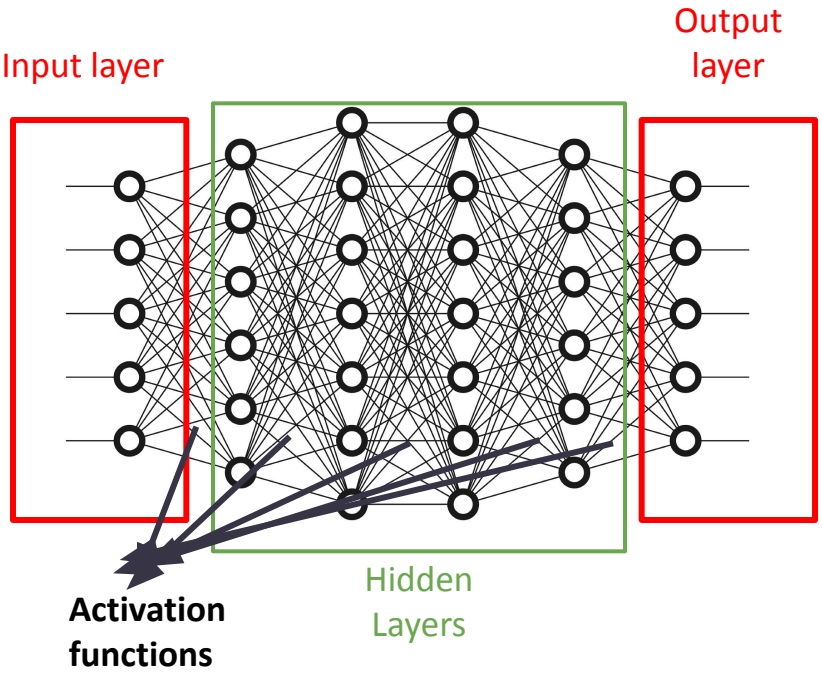




Let's prepare data for our model!

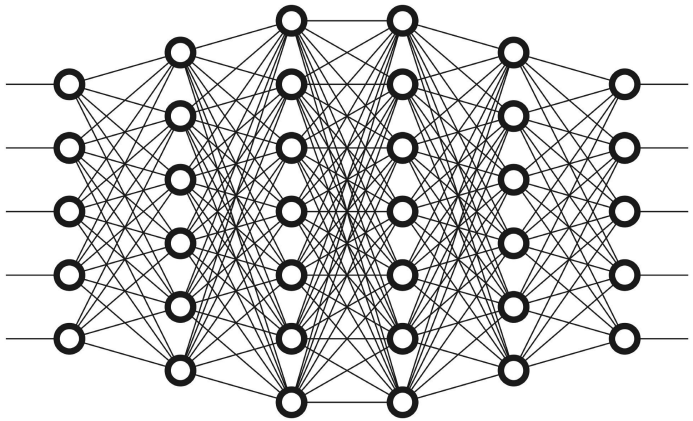
How do we teach MLP to classify?





How do we teach MLP to classify?

Learning good weights



Learning Recipe:

1-Initialize weights

2-Run a forward pass and make a prediction

3-Calculate loss to evaluate how good/bad your prediction was

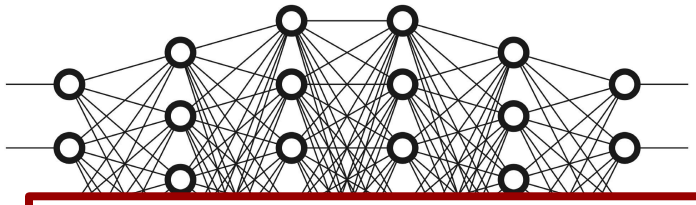
4-Adjust your weights in a way so that loss will be minimized

Popular weight initialization methods: Xavier/Glorot Initialization , Normalized Xavier/Glorot Initialization, He Weight Initialization, Random Weight Initialization

Appendix: Andrew Ng's Lecture on Weight Initialization

How do we teach MLP to classify?

Learning good weights

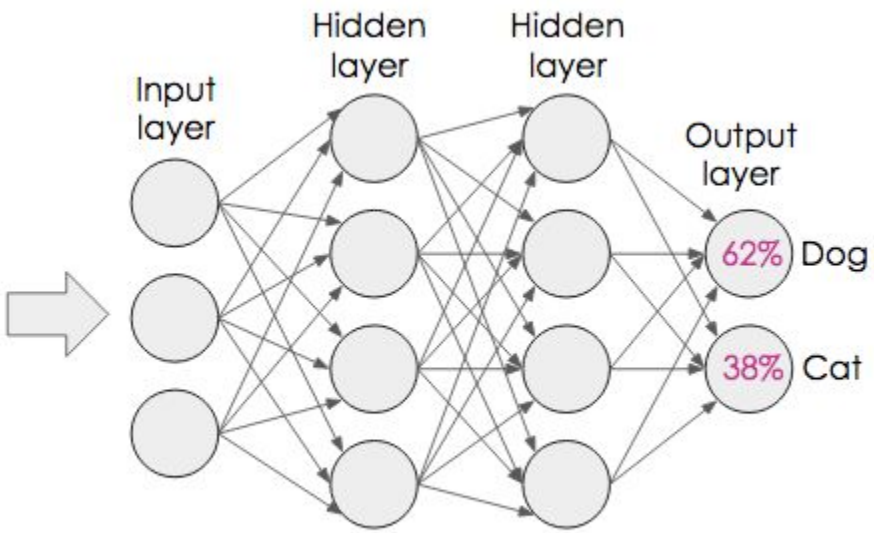


Learning Recipe:

1-Initialize weights

2-Run a forward pass and make a prediction

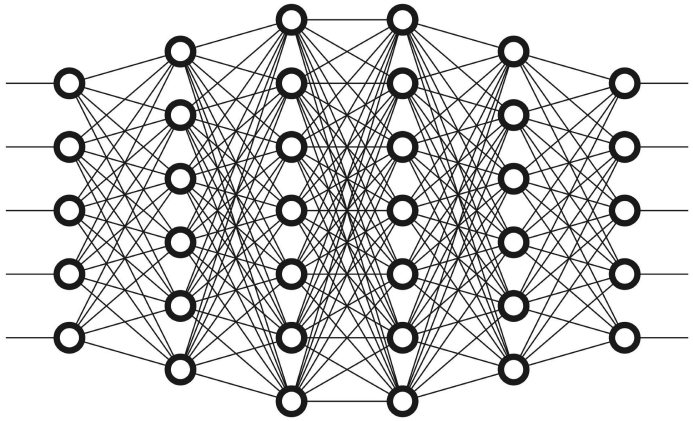
3-Calculate loss to evaluate how



It should be 100% Cat :(

How do we teach MLP to classify?

Learning good weights



Learning Recipe:

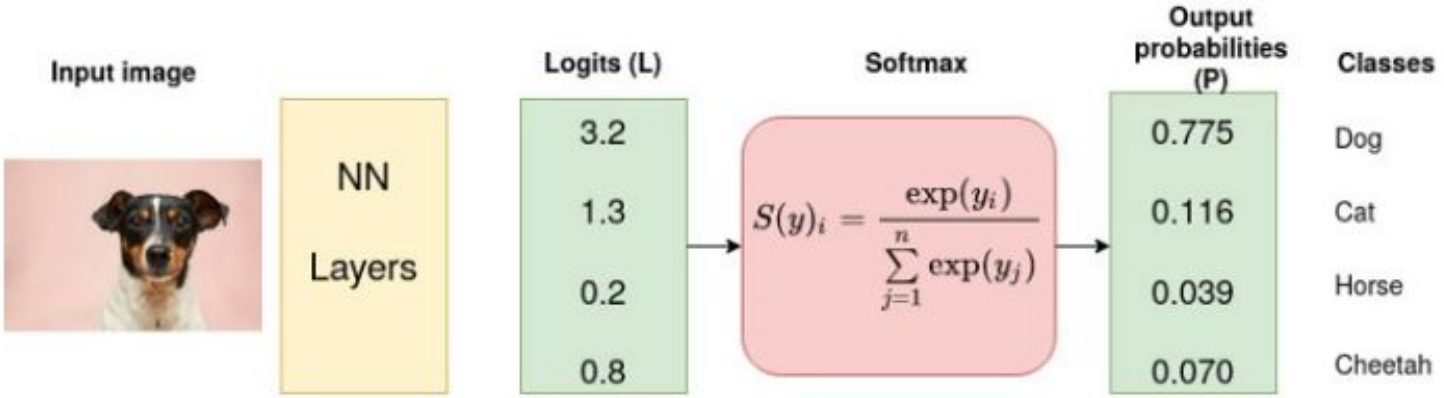
1-Initialize weights

2-Run a forward pass and make a prediction

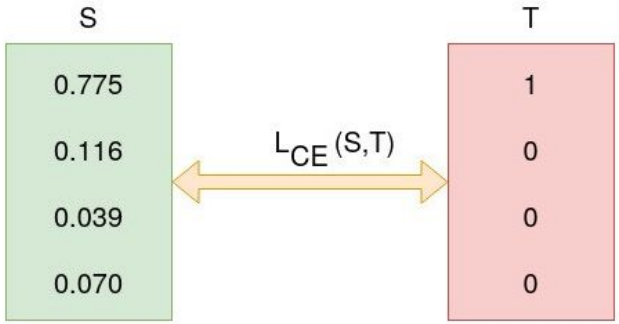
3-Calculate loss to evaluate how good/bad your prediction was

4-Adjust your weights in a way so that loss will be minimized

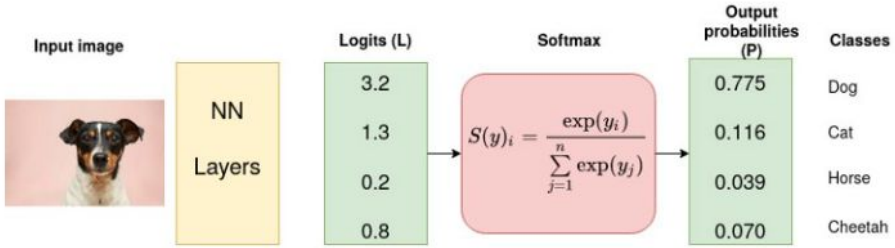
Classifying categorical variables -> Cross-entropy Loss Function



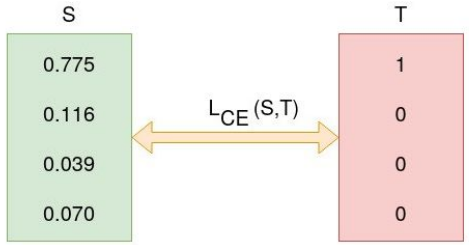
Input image source: Photo by Victor Grabarczyk on Unsplash . Diagram by author.



Cross Entropy (L) (Source: Author).



Input image source: Photo by Victor Grabarczyk on Unsplash . Diagram by author.



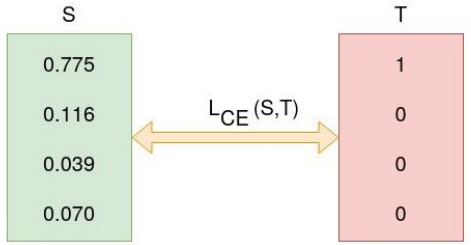
Cross Entropy (L) (Source: Author).

$$L_{\text{CE}} = -\sum_{i=1}^{n} t_i \log(p_i), \text{ for n classes,}$$

where t_i is the truth label and p_i is the Softmax probability for the i^{th} class.

$$L_{\text{CE}} = -\sum_{i=1}^{n} t_i \log(p_i), \quad \text{for n classes,}$$

where t_i is the truth label and p_i is the Softmax probability for the i^{th} class.

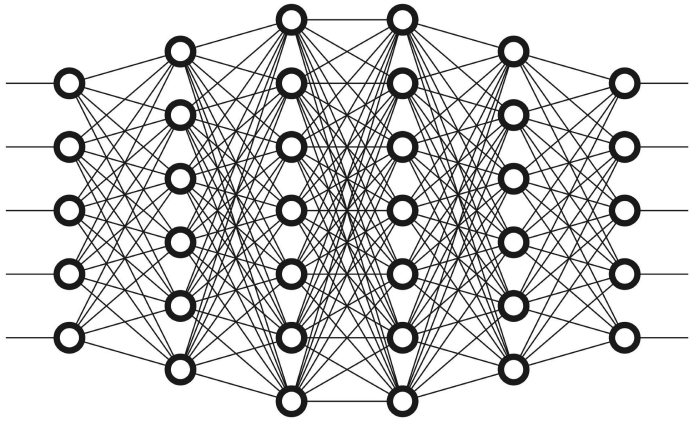


Cross Entropy (L) (Source: Author).

$$\begin{aligned} L_{CE} &= -\sum_{i=1} T_i \log(S_i) \\ &= -\left[1 \log_2(0.775) + 0 \log_2(0.126) + 0 \log_2(0.039) + 0 \log_2(0.070)\right] \\ &= -\log_2(0.775) \\ &= 0.3677 \end{aligned}$$

How do we teach MLP to classify?

Learning good weights



Learning Recipe:

1-Initialize weights

2-Run a forward pass and make a prediction

3-Calculate loss to evaluate how good/bad your prediction was

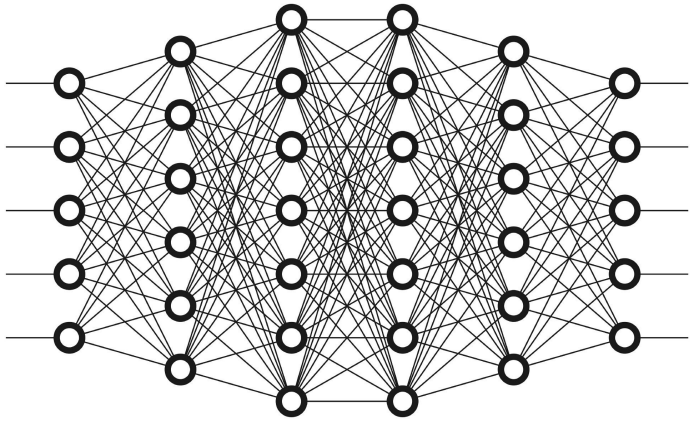
4-Adjust your weights in a way so that loss will be minimized

Optimization! -> More details in the next lecture

We will use Adam Optimizer for this tutorial

How do we teach MLP to classify?

Learning good weights



Learning Recipe:

1-Initialize weights

2-Run a forward pass and make a prediction

3-Calculate loss to evaluate how good/bad your prediction was

4-Adjust your weights in a way so that loss will be minimized

Let's train our MLP model!!