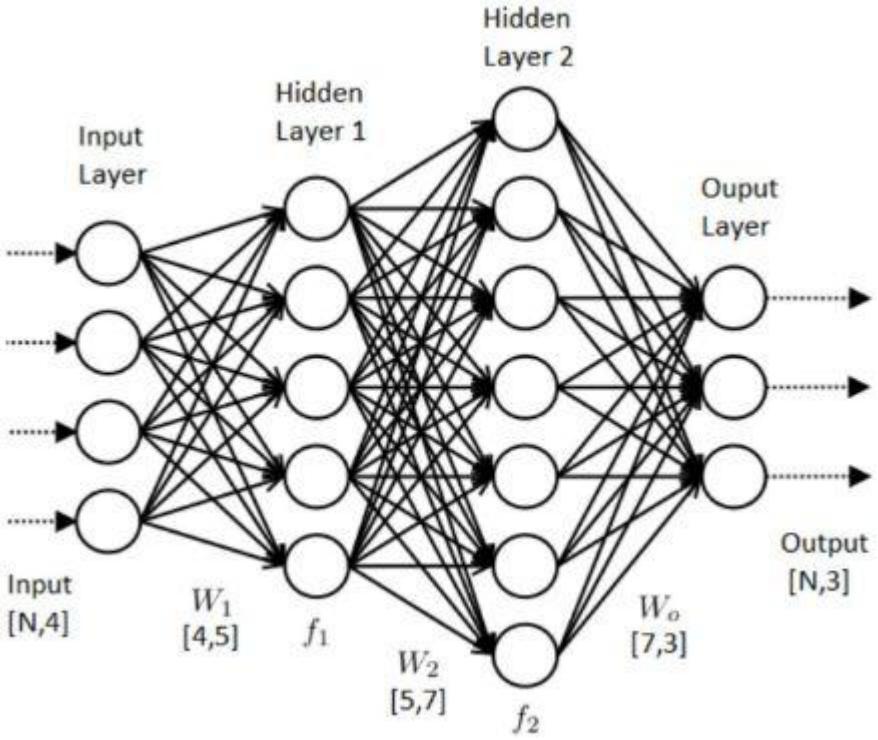
Neuron network

minhtt@vietinbank.vn

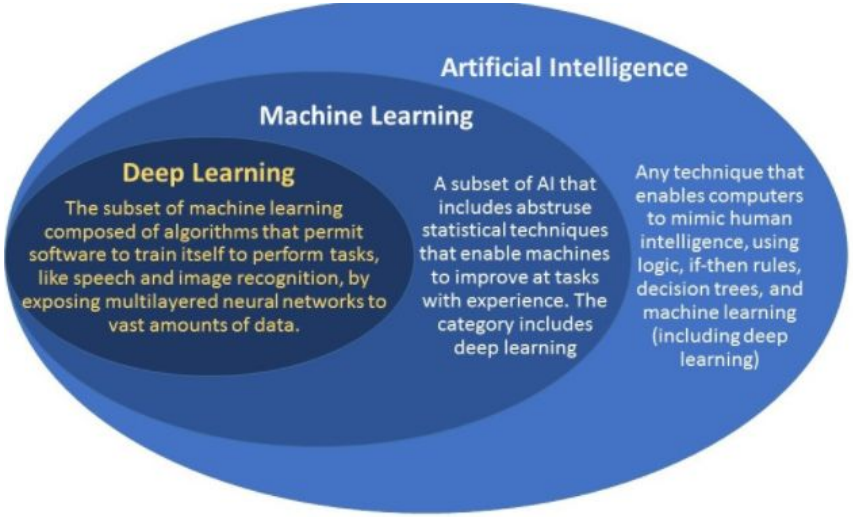


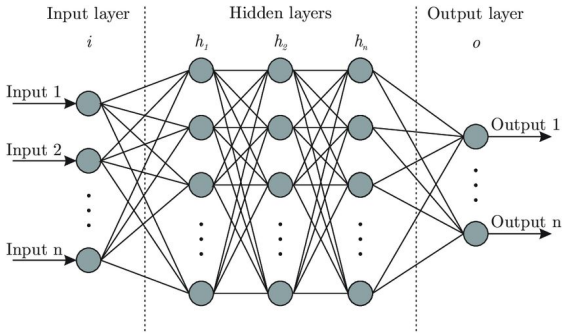
Index

1. What is neuron network
2. Application
3. Why we need?
4. Structure
5. Algorithm
    a. Gradient descent
    b. Loss funtion
    c. Activation function
    d. Feed forward
    e. Back propagation
6. Simple neuron network from scratch (lab excercise)
7. Reference

What is neuron network

- Subset of machine learning
- Are comprised of a node layers, containing an input layer, one or more hidden layers, and an output layer





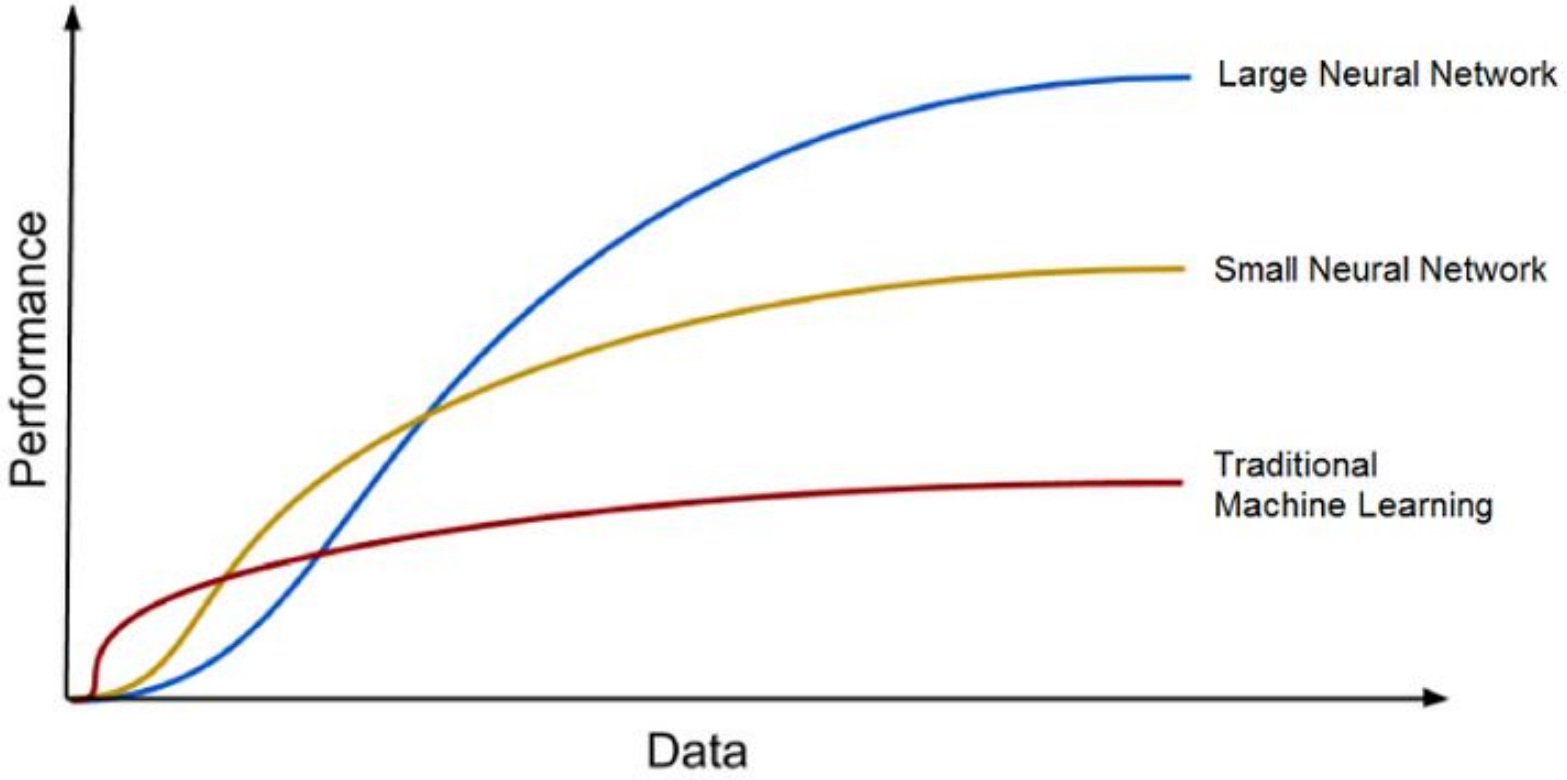
Application

- Facial Recognization
- Stock maket prediction
- Machine translation
- OCR
- Fraud detection
- …

Playground:

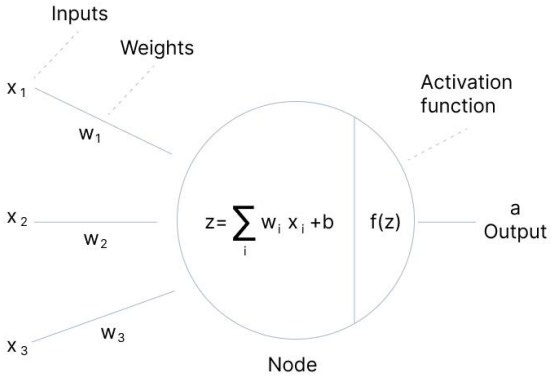
- https://teachablemachine.withgoogle.com/
- https://playground.tensorflow.org/

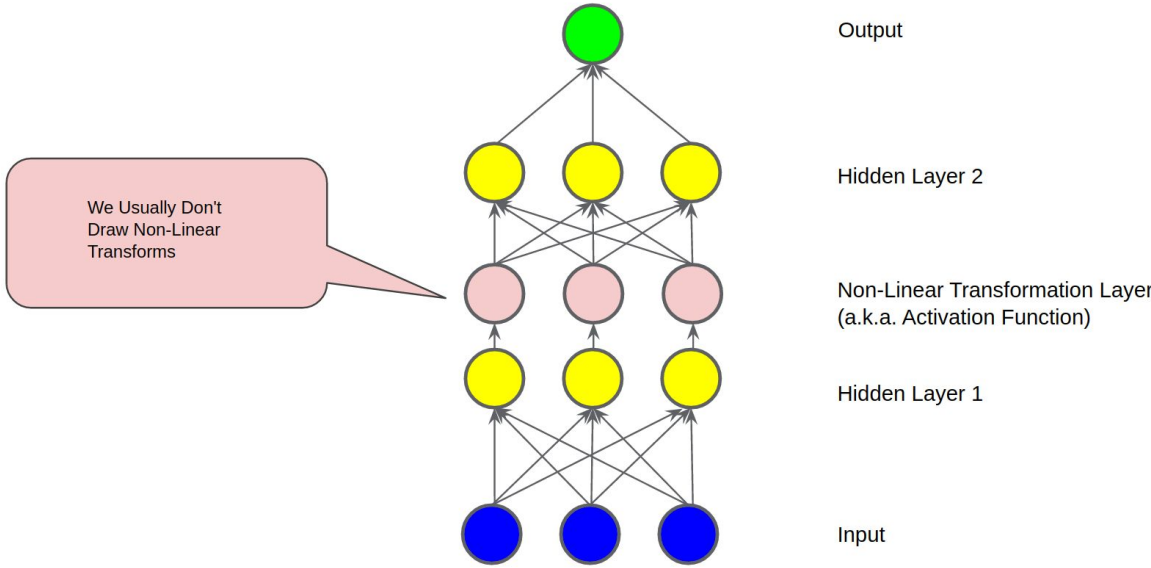
Why we need?


Large Neural Network
Small Neural Network
Traditional
Machine Learning
Performance
Data

Structure

- Input
- Hidden layer
- Activation function
- Output





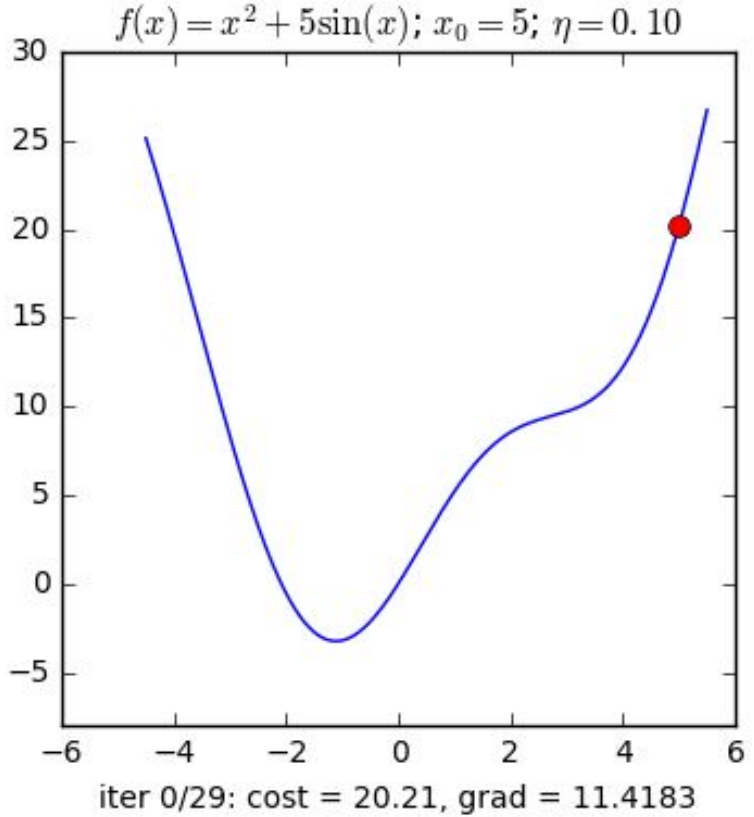
Algorithm

- Gradient descent

$$x_1 = x_0 - \alpha \nabla f(x_0)$$

More generally, we can write a formula for turning x_n into x_{n+1}:

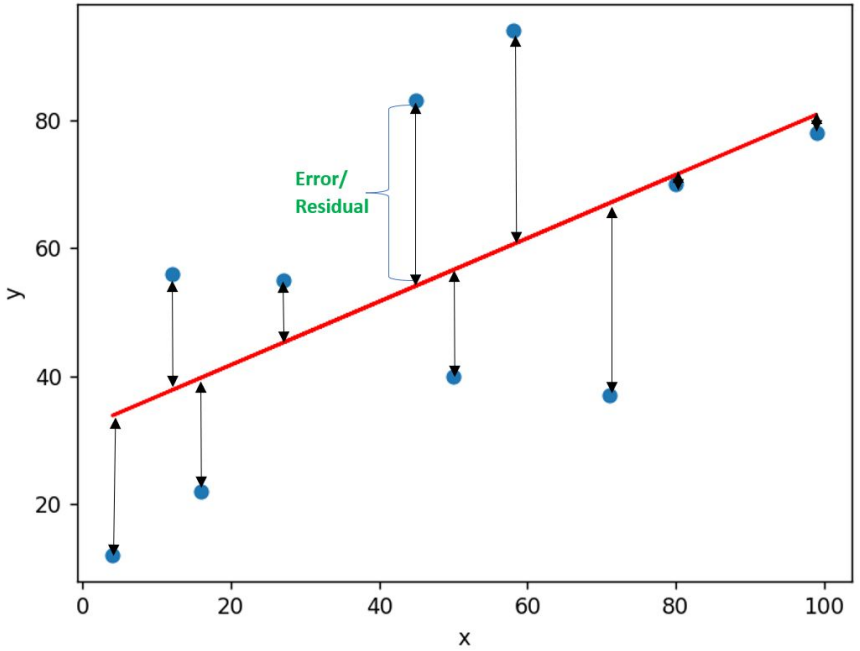
$$x_{n+1} = x_n - \alpha \nabla f(x_n)$$



Algorithm

- Loss function

The loss function is the function that determines how far the algorithm's current output is from what is desired. This is a technique for assessing how well our algorithm models the input. It can be divided into two categories. Both for regression and for classification



Algorithm

- Loss function

MAE (L1)

Mean Absolute Error, or L1 loss. Excellent overview below [6] and [10].

$$MAE = \frac{1}{m}\sum_{i=1}^{m}|h(x^{(i)}) - y^{(i)}|$$

MAE - mean absolute error

m - number of samples

$x^{(i)}$ - i-th sample from dataset

$h(x^{(i)})$ - prediction for i-th sample (thesis)

$y^{(i)}$ - ground truth label for i-th sample

Algorithm

- Loss function

MSE (L2)

Mean Squared Error, or L2 loss. Excellent overview below [6] and [10].

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y^{(i)} - \hat{y}^{(i)})^2$$

MSE - mean square error

m - number of samples

$y^{(i)}$ - ground truth label for i-th sample

$\hat{y}^{(i)}$ - predicted label for i-th sample

Algorithm

- Loss function

BINARY CROSS-ENTROPY LOSS / LOG LOSS

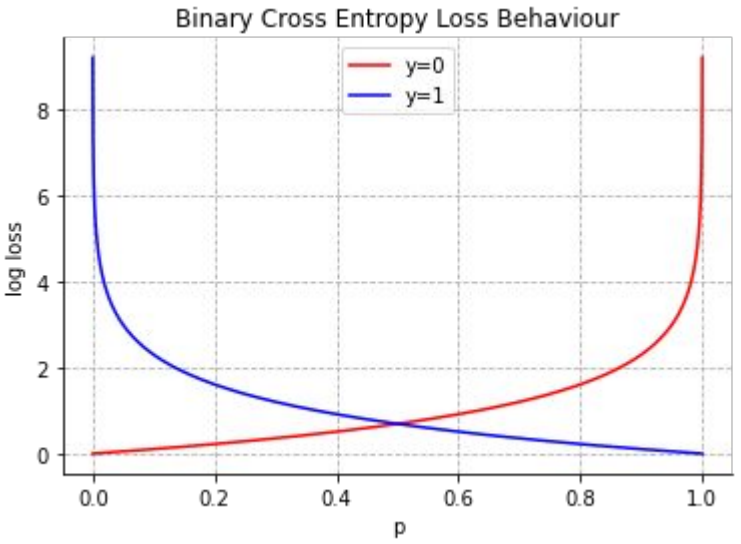
This is the most common loss function used in classification problems. The cross-entropy loss decreases as the predicted probability converges to the actual label. It measures the performance of a classification model whose predicted output is a probability value between `0` and `1`.

When the number of classes is `2`, it's *binary classification*.

$$L = -\frac{1}{m}\sum_{i=1}^{m}(y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i))$$

When the number of classes is more than `2`, it's *multi-class classification*.

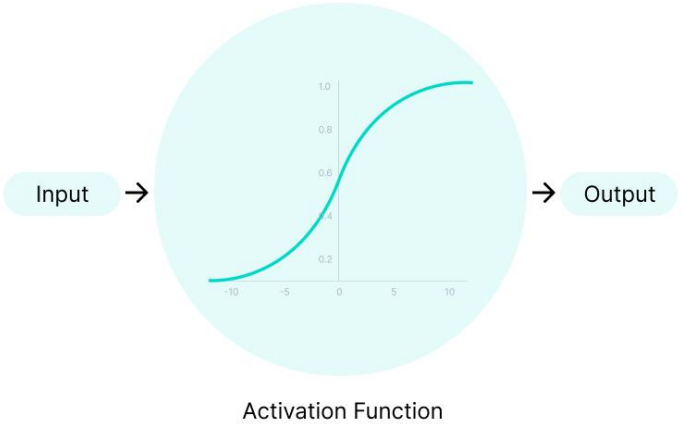
$$L = -\frac{1}{m}\sum_{i=1}^{m} y_i \cdot \log(\hat{y}_i)$$

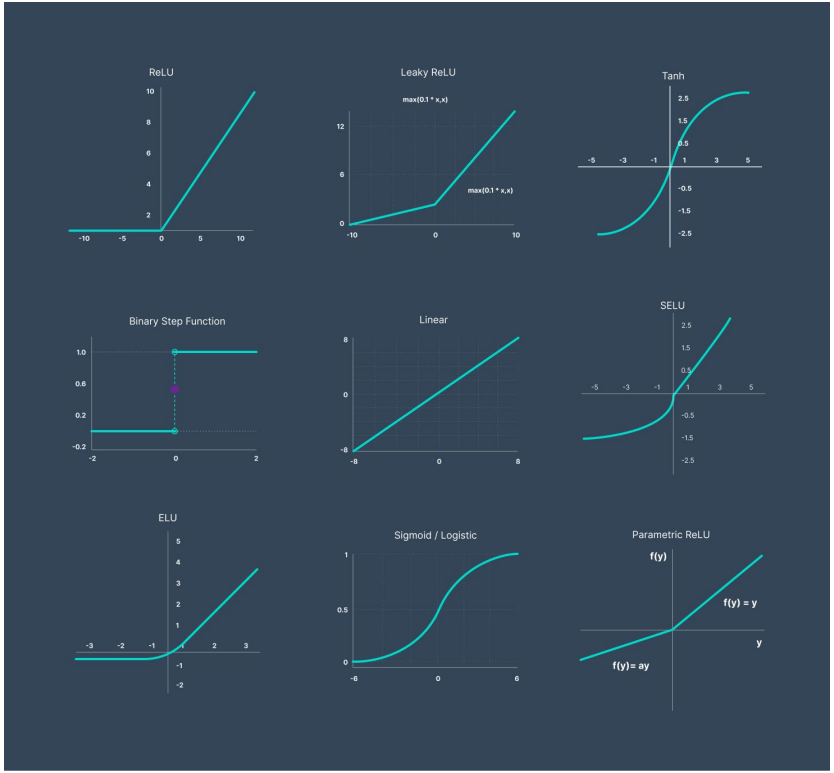


Algorithm

- Activation function

Why do Neural Networks Need an Activation Function





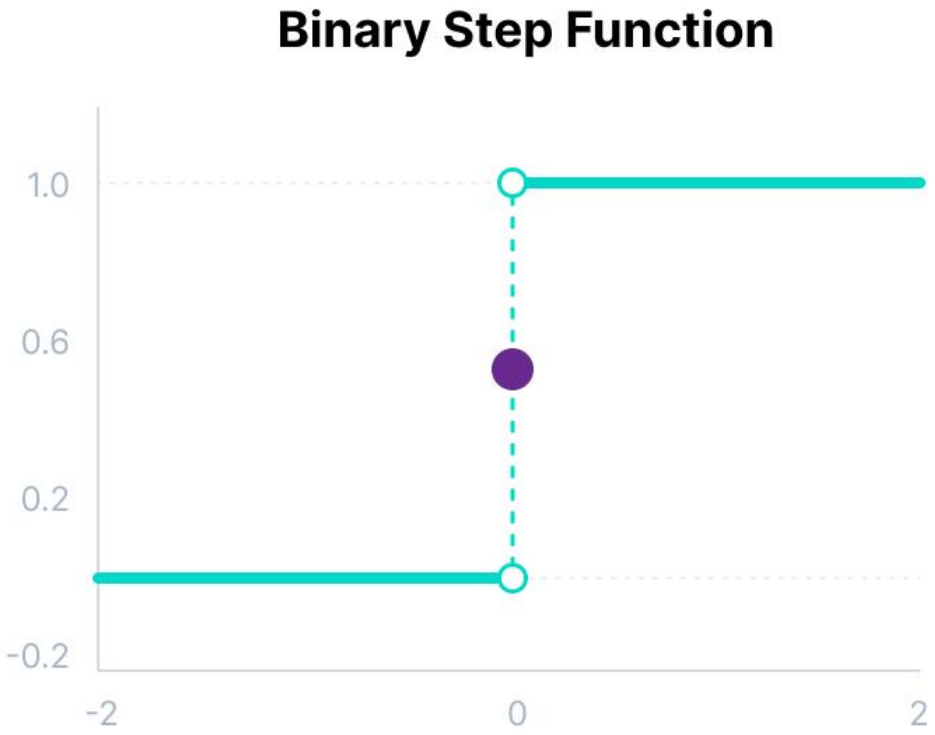
👉The purpose of an activation function is to add non-linearity to the neural network

Algorithm

- Activation function

Binary step

$$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geqslant 0 \end{cases}$$

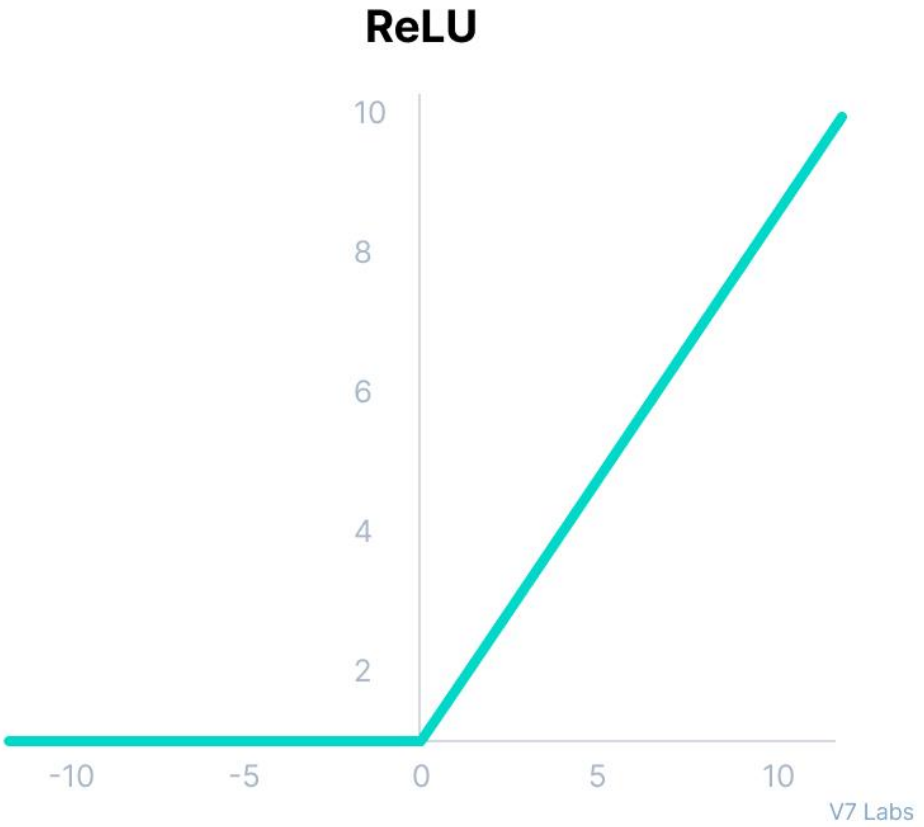


Algorithm

- Activation function

$$ReLU$$

$$f(x) = max\ (0, x)$$

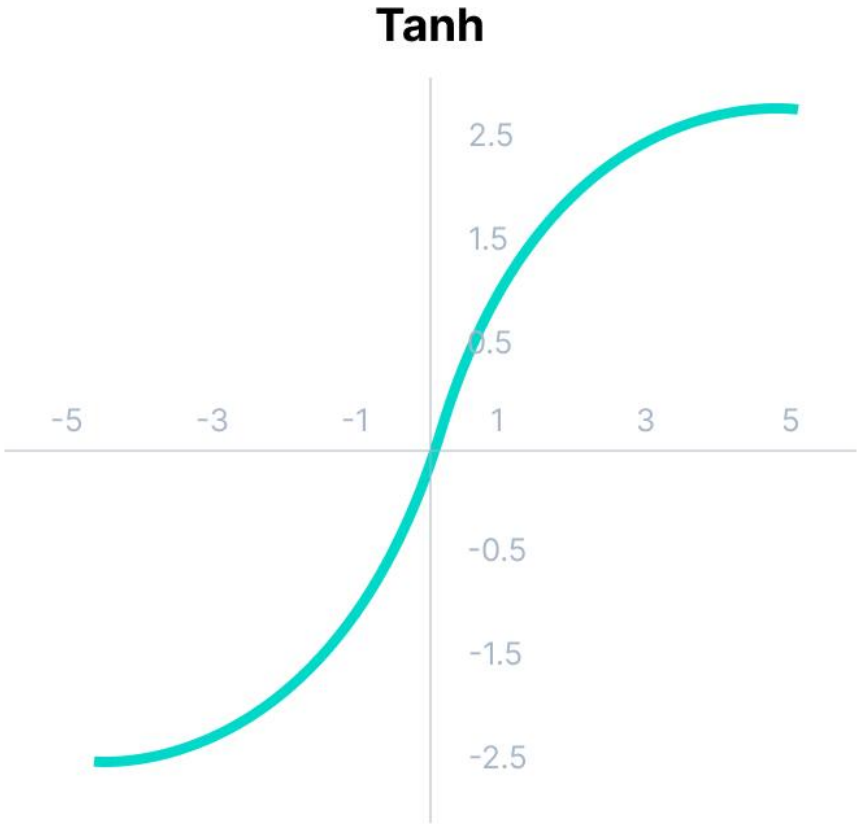


Algorithm

- Activation function

$$Tanh$$

$$f(x) = \frac{\left(e^{x} - e^{-x}\right)}{\left(e^{x} + e^{-x}\right)}$$

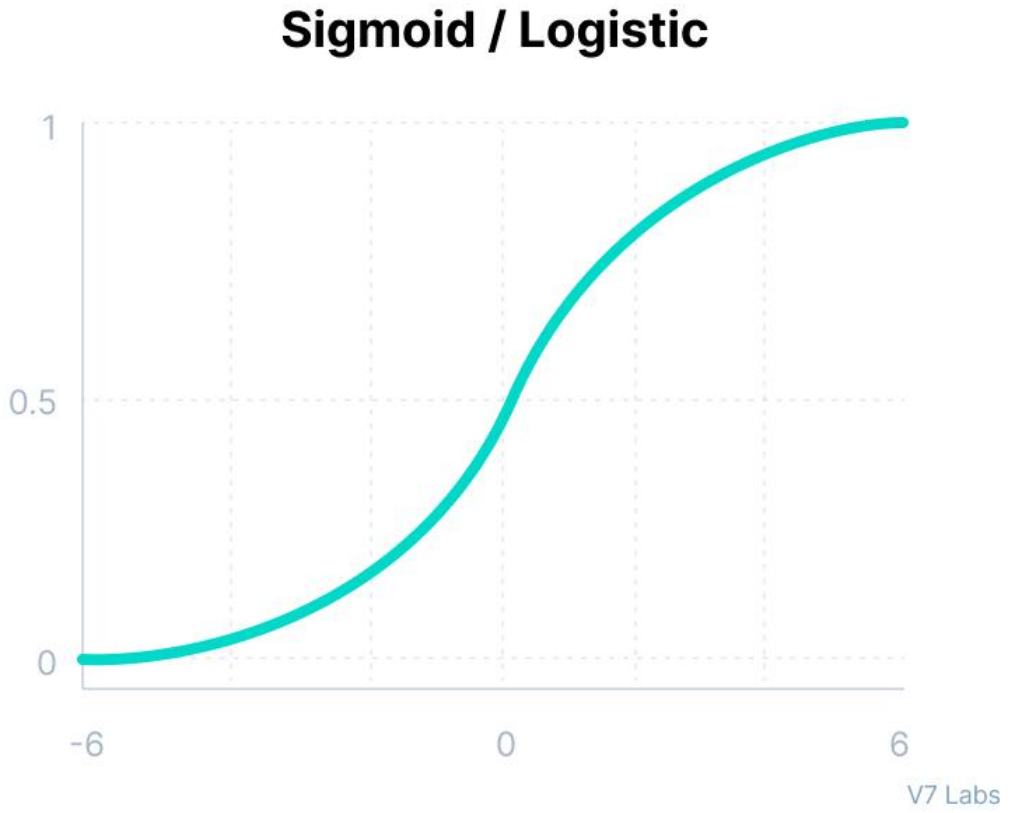


Algorithm

- Activation function

Sigmoid / Logistic

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

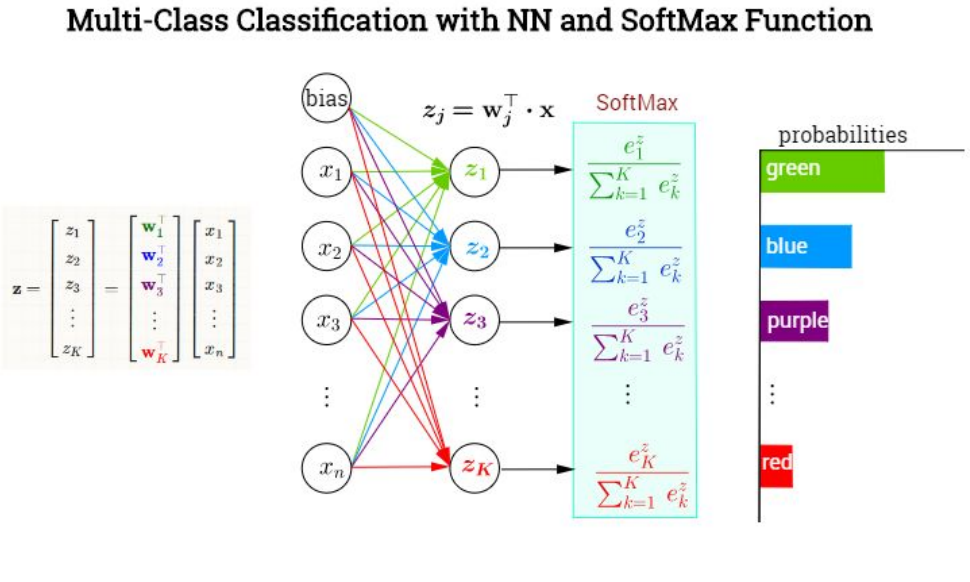


Algorithm

- Activation function

Softmax

$$\text{softmax}(z_i) = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$$

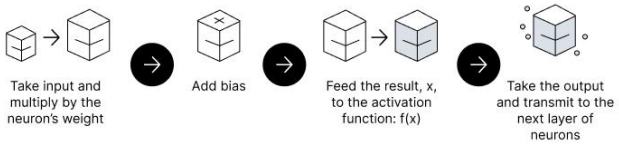


V7 Labs

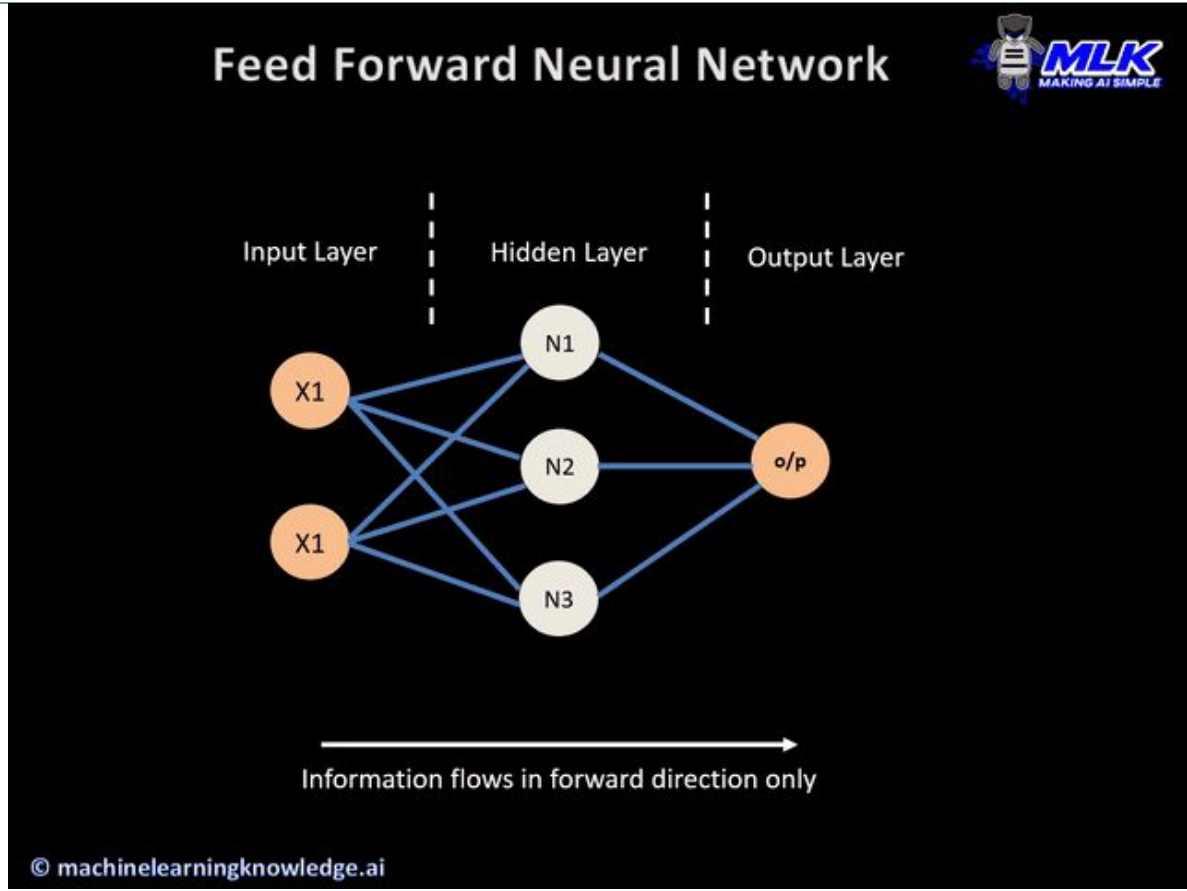
Algorithm

- Feed forward

The flow of information occurs in the forward direction. The input is used to calculate some intermediate function in the hidden layer, which is then used to calculate an output.

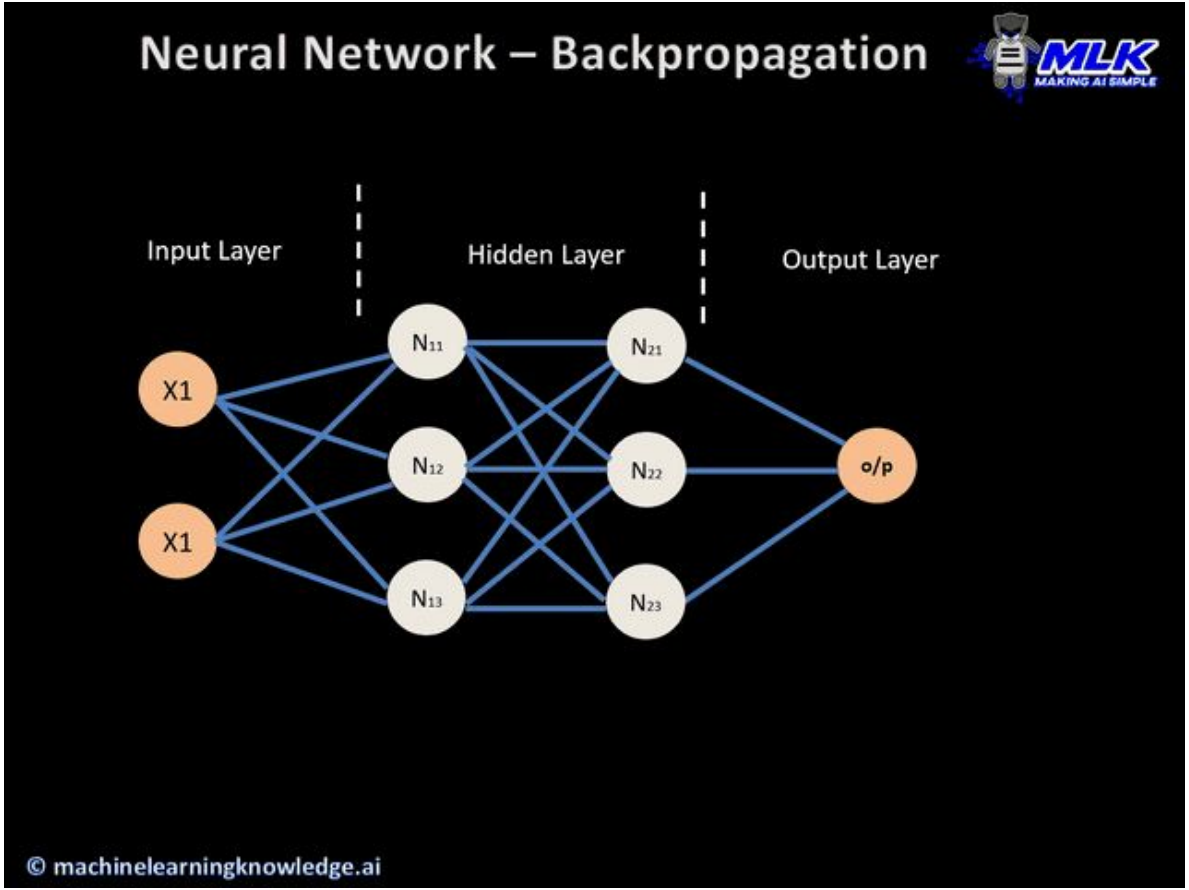


V7 Labs



Algorithm

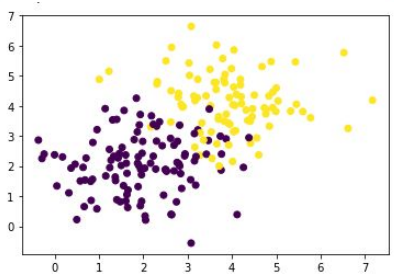
- Back propagation

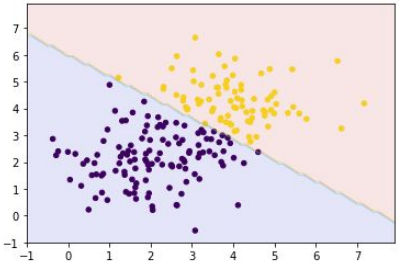


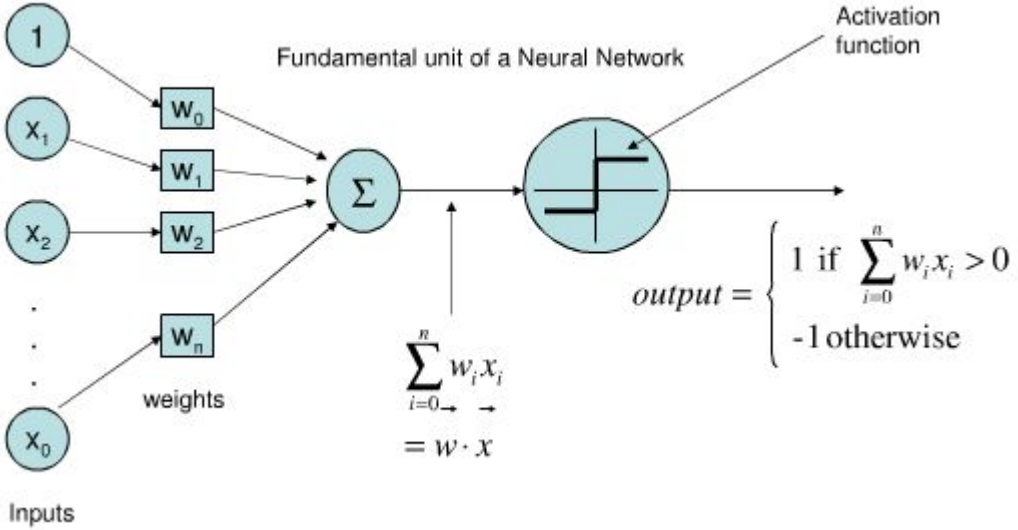
Simple neuron network from scratch

	x1	x2	y
0	0.575939	1.506680	-1
1	4.687881	4.049857	1
2	3.817095	4.896248	1
3	4.239867	4.158959	1
4	1.352819	2.472247	-1
...	...	...	...
995	3.230291	3.202380	-1
996	0.981958	1.922145	-1
997	4.844996	3.324406	1
998	3.831786	4.377910	1
999	4.623454	4.890672	1

1000 rows × 3 columns







- Predict: $\hat{\mathbf{y}}_i = \mathbf{sign}(\dot{\mathbf{w}}^\top \dot{\mathbf{x}}_i)$
- Update rule: $\dot{\mathbf{w}} = \dot{\mathbf{w}} - \eta \dot{\mathbf{x}}_i \mathbf{y}_i$ for all missclassified $\dot{\mathbf{x}}_i$

References

- https://playground.tensorflow.org
- https://teachablemachine.withgoogle.com
- https://www.v7labs.com/blog/neural-networks-activation-functions
- https://builtin.com/machine-learning/common-loss-functions
- https://rpubs.com/harshaash/ANN_1
- https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html
- https://machinelearningknowledge.ai/animated-explanation-of-feed-forward-neural-network-architecture/


Thank You!