ICT Academy of Kerala

Building the Nation's Future

Artificial Neural Network

A **GOVT. OF INDIA** SUPPORTED, **GOVT. OF KERALA** PARTNERED SOCIAL ENTERPRISE.

Government of India

TATA CONSULTANCY SERVICES

History Of AI

- Talos - Greek Mythology
  - Giant automation made up of Bronze to protect Europa from pirates
- 1950 - Turing Test
  - Paper by Alan Turing

History Of AI

- 1956 - Artificial Intelligence
  - Coined by John McCarthy at the Dortmund conference
- 1959 - Ist AI lab was setup
  - MIT - still running
- 1960 - Ist Robot was introduced
  - In GM assembly line

History Of AI

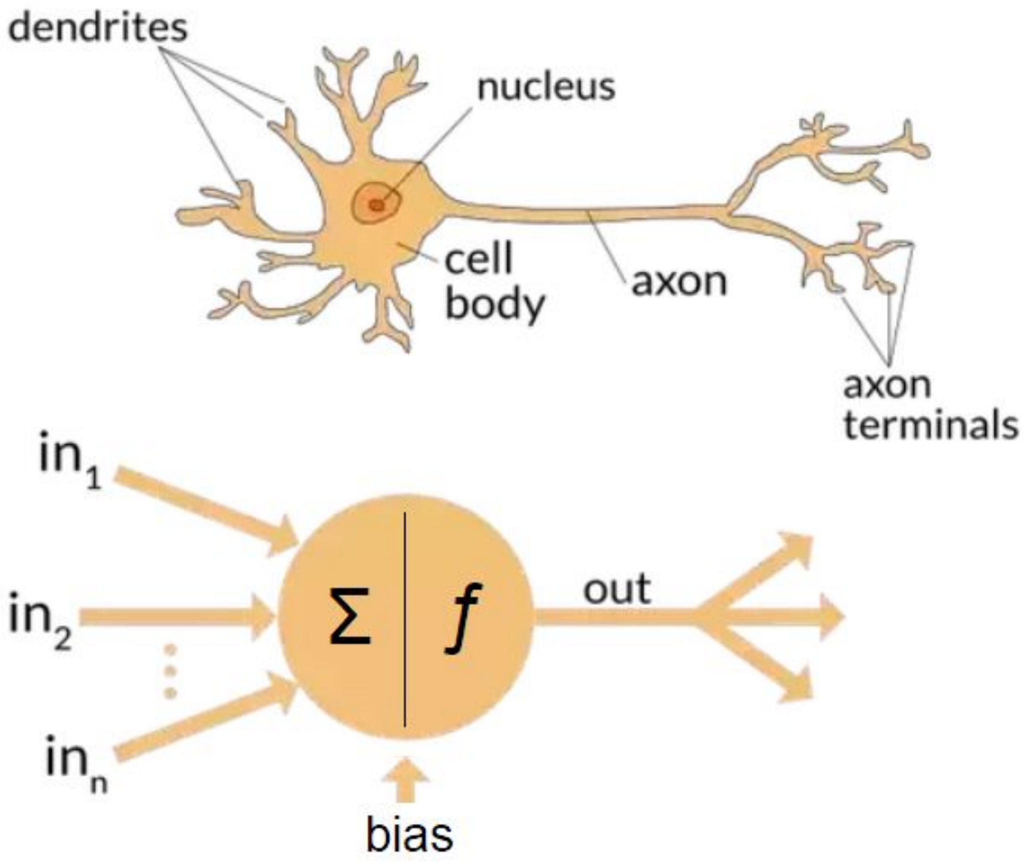
- 1961 - Ist chatbot was introduced
  - Eliza was an NLP program created at MIT lab
  - Created to demonstrate the superficiality of communication between humans and machines
- 1997 - IBM Deep Blue
  - Defeated Garry Kasparov in the game of chess
  - He was the World champion then

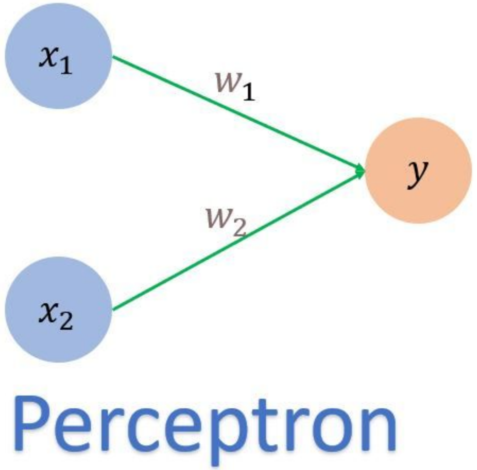
History Of AI

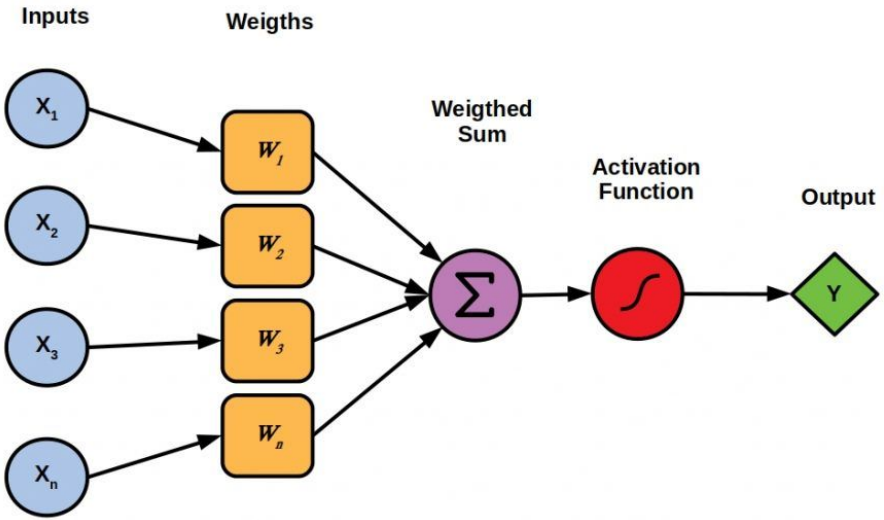
- 2005 - DARPA Grand challenge
  - Stanley wins the challenge
  - Stanford Racing team’s autonomous robotic car
- 2011 - Watson
  - IBMs Q&A system
  - Defeated Jeopardy winners Brad Rutter & Ken Jennings

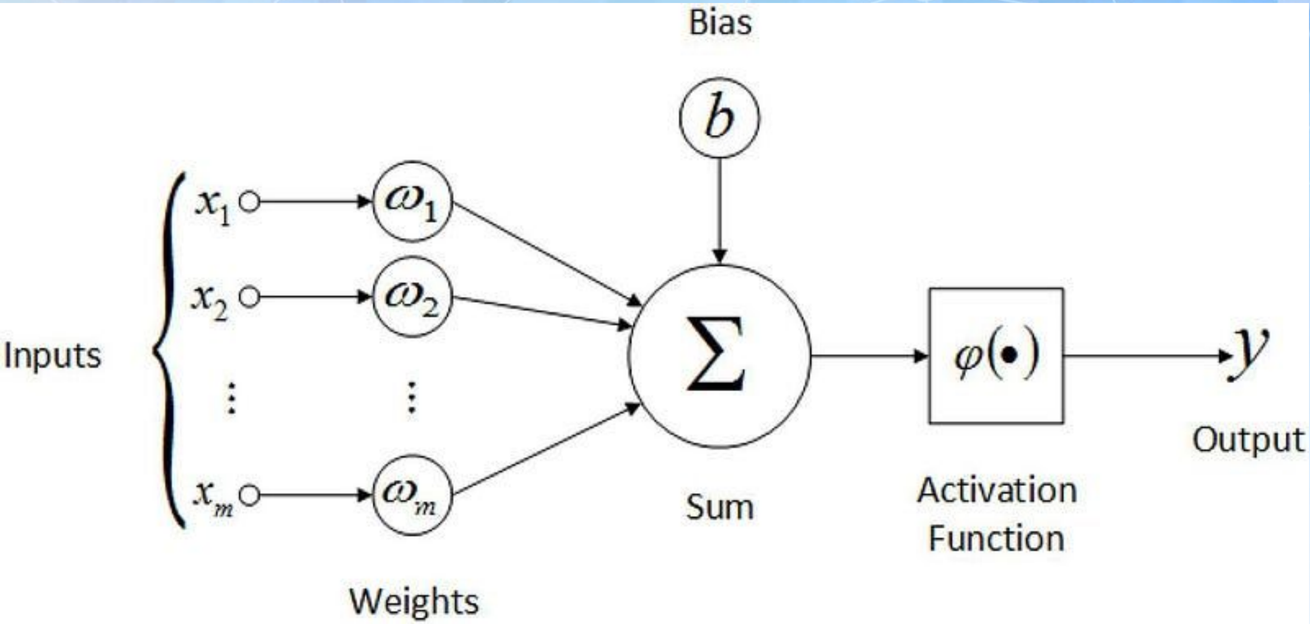
Artificial Neural Network

- Artificial Neural Network (ANN) is a deep learning method
- Based on the concept of the human brain Biological Neural Networks.


dendrites
nucleus
cell body
axon
axon terminals
in1
in2
inn
Σ
f
out
bias


x_1
w_1
y
w_2
x_2
Perceptron


Inputs
Weigths
Weigthed Sum
Activation Function
Output
X_1
X_2
X_3
X_n
W_1
W_2
W_3
W_n
Σ
Y



$$y_{in} = x_1.w_1 + x_2.w_2 + x_3.w_3 \ldots x_m.w_m$$

$$y_{in} = \sum_i^m x_i.w_i$$

$$Y = F(y_{in})$$

PERCEPTRON

- Perceptron is a building block of an Artificial Neural Network
- Also known as an Artificial Neuron
- a single-layer neural network with four main parameters,
  - input values
  - weights and Bias
  - net sum
  - activation function.

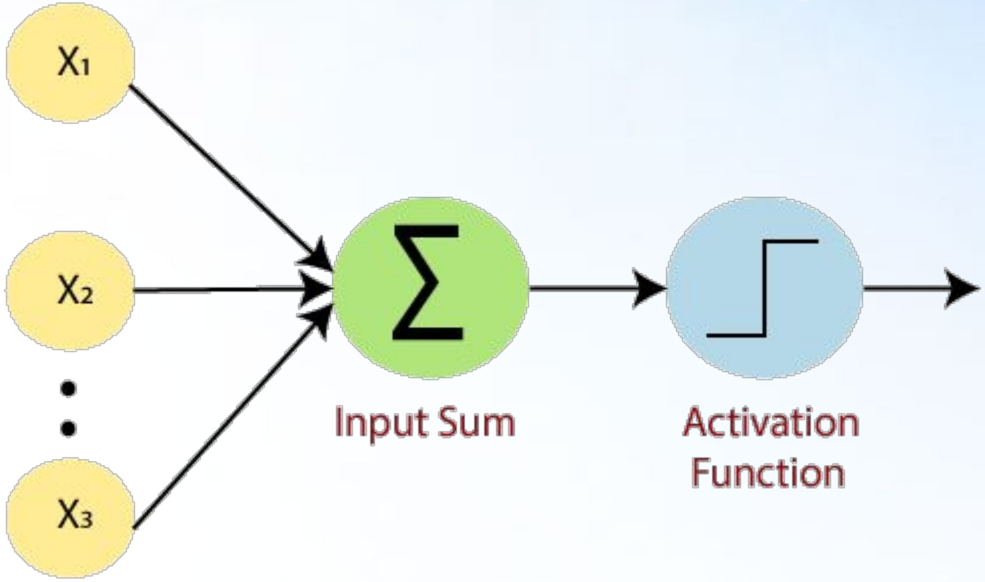
Types of Perceptron Models

Based on the layers, Perceptron models are divided into two types. These are as follows:

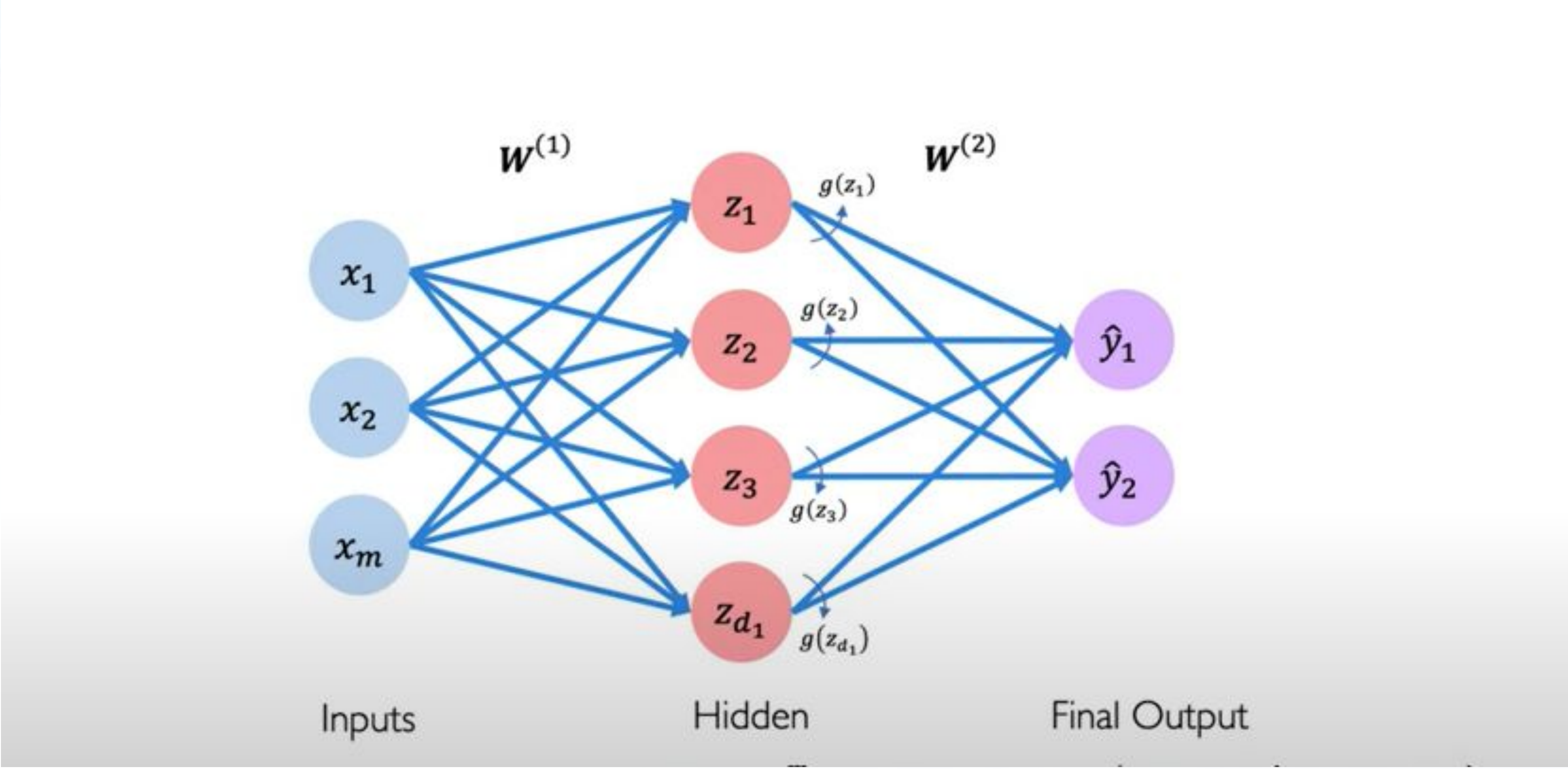
1. Single-layer Perceptron Model : to analyze the linearly separable objects with binary outcomes.

1. Multi-layer Perceptron model : Two or more layers with higher processing power.

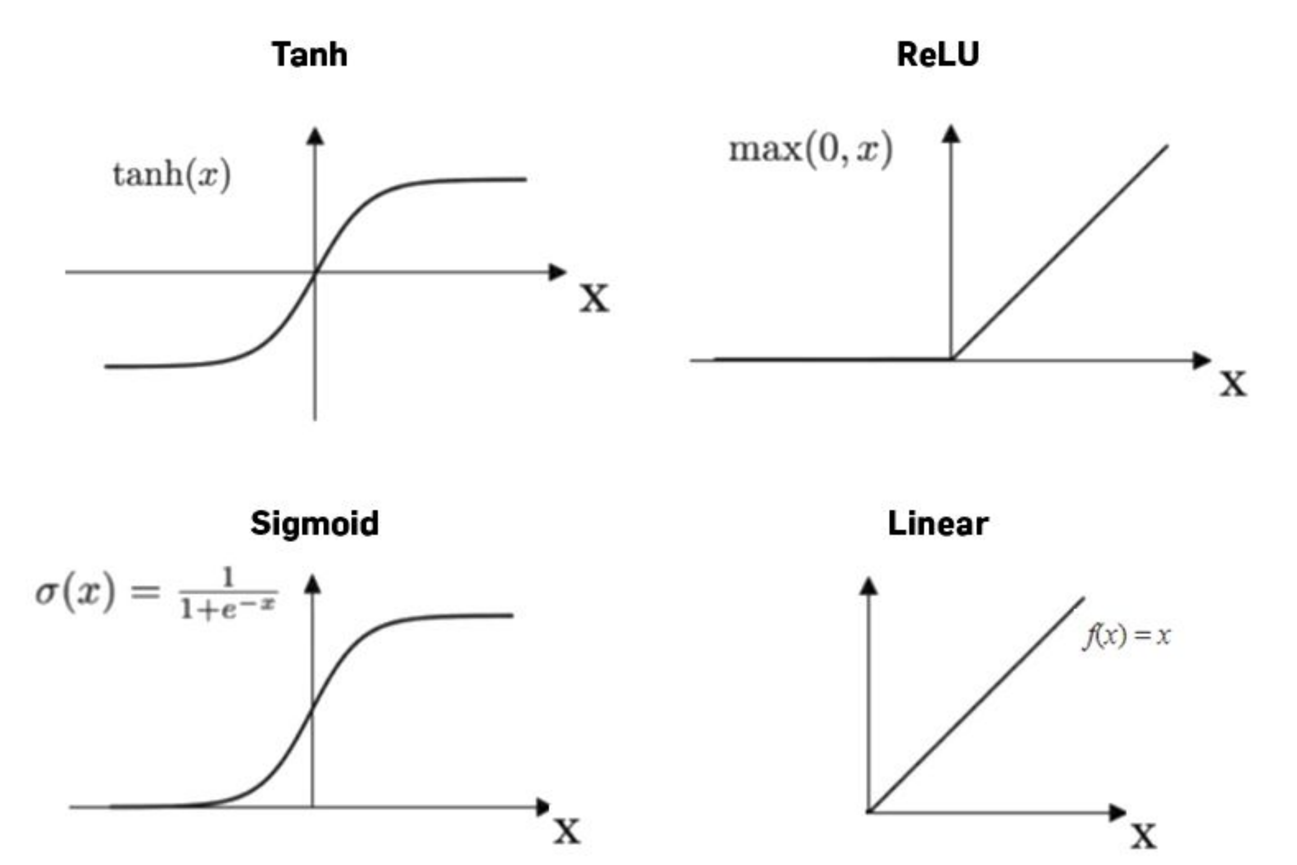
Single Layer Perceptron



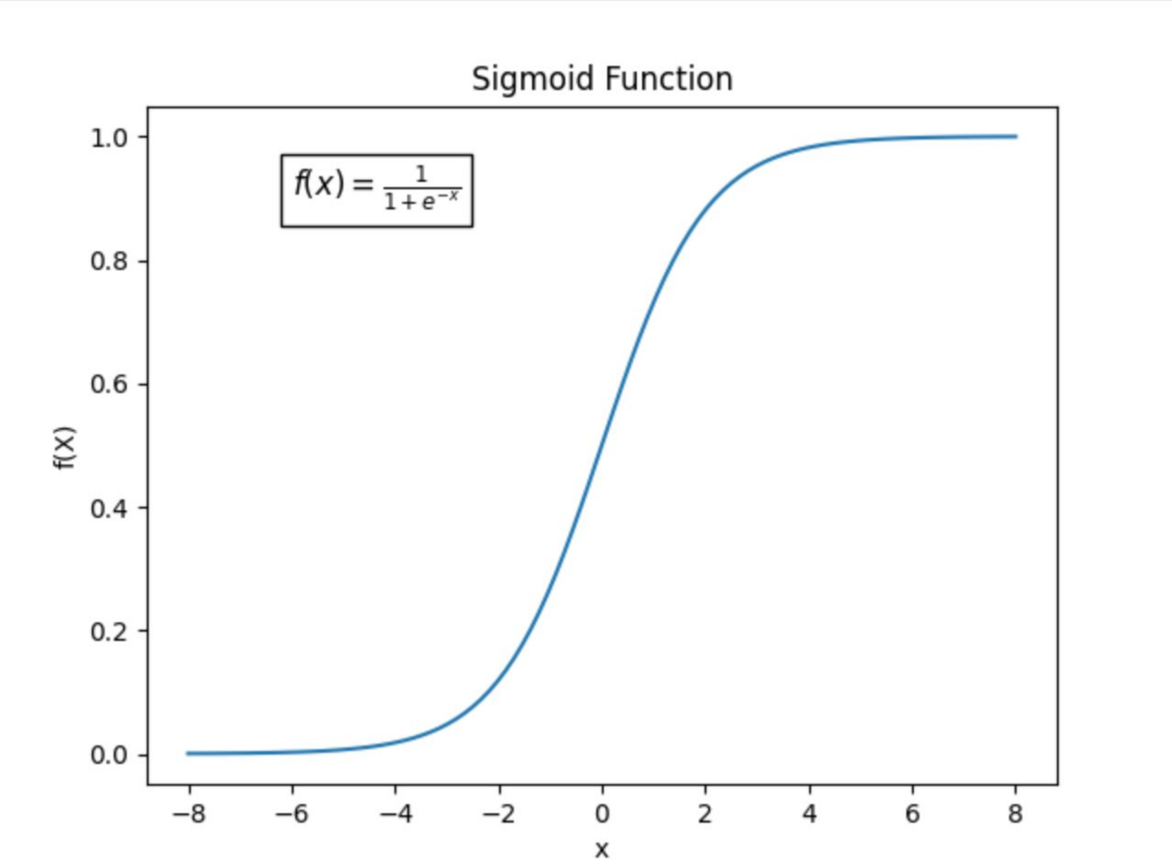
MULTI LEVEL PERCEPTRON



ACTIVATION FUNCTION

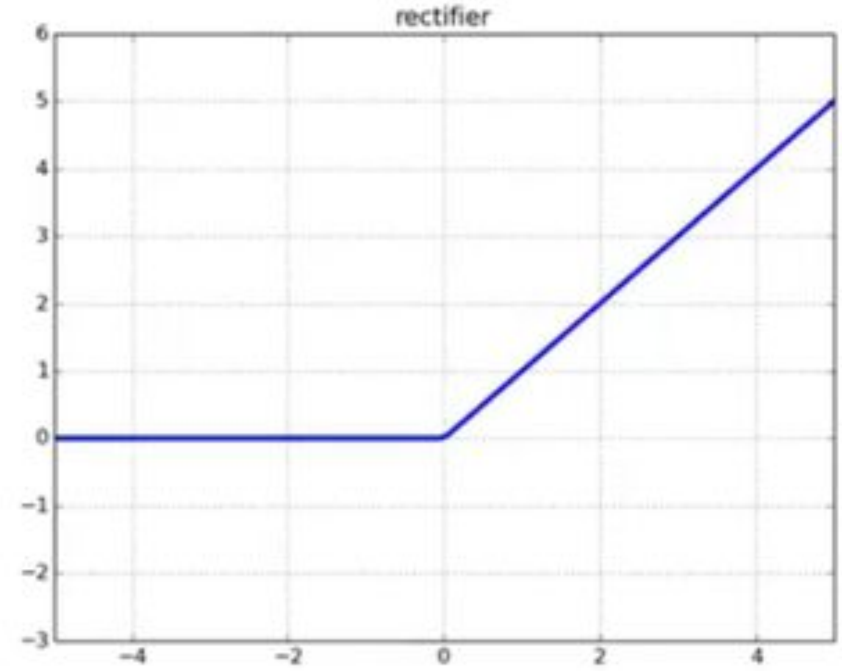


Sigmoid

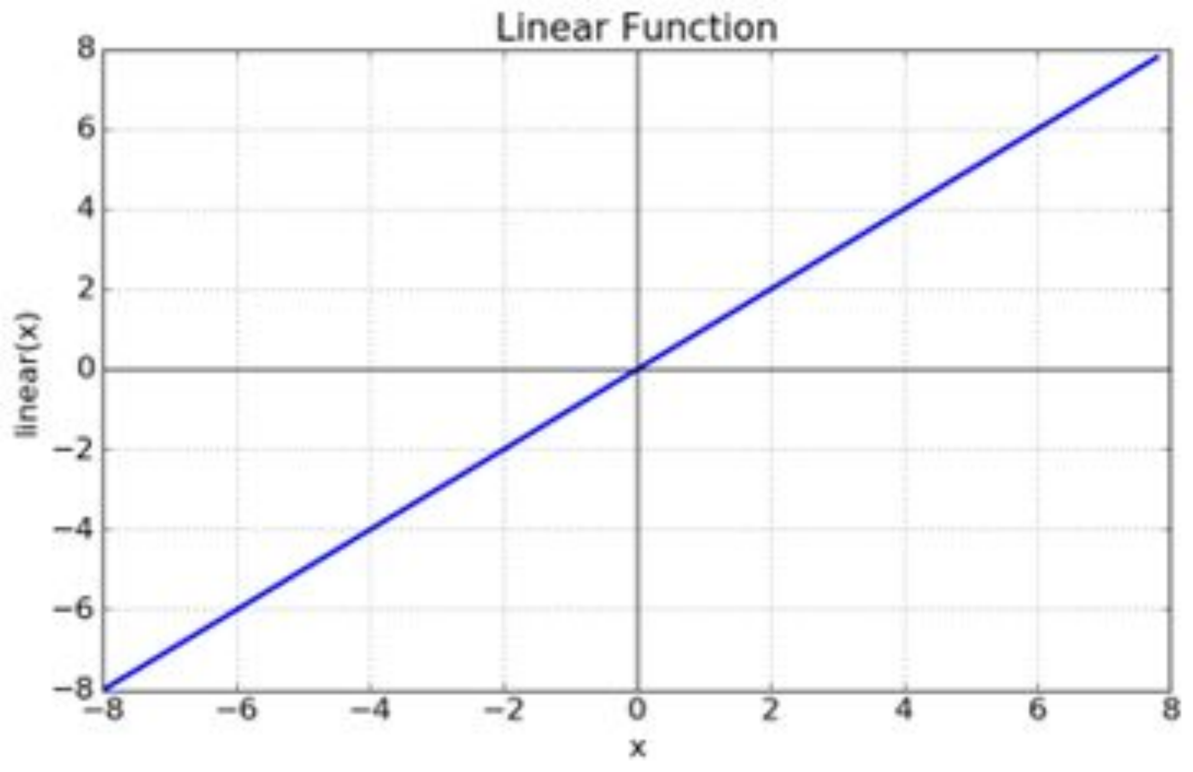


ReLu

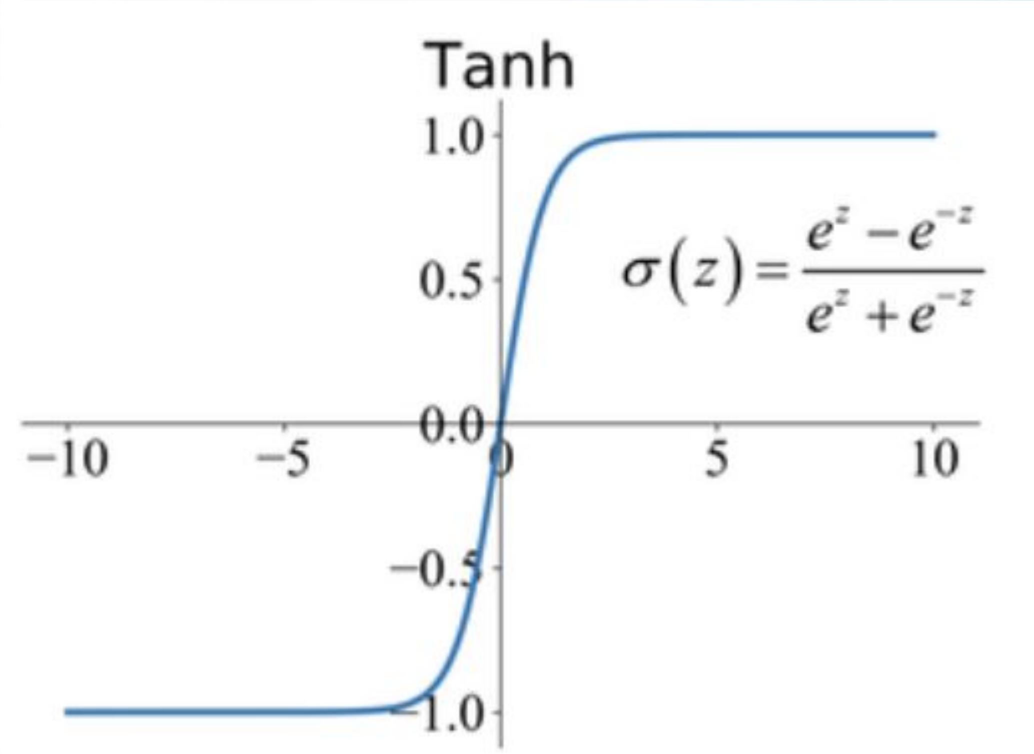
$$f(x)= \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x => 0 \end{cases}$$



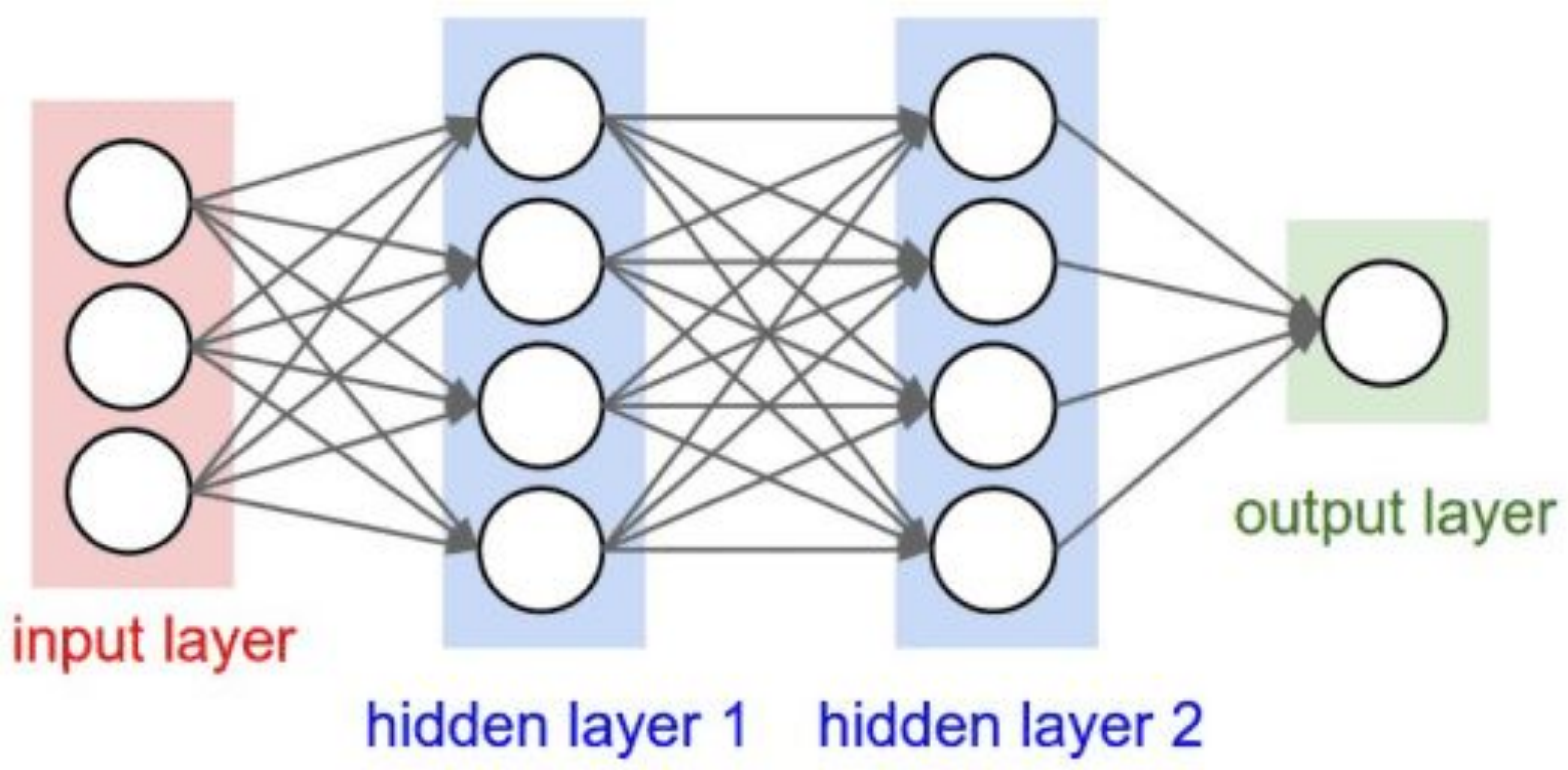
Linear Activation

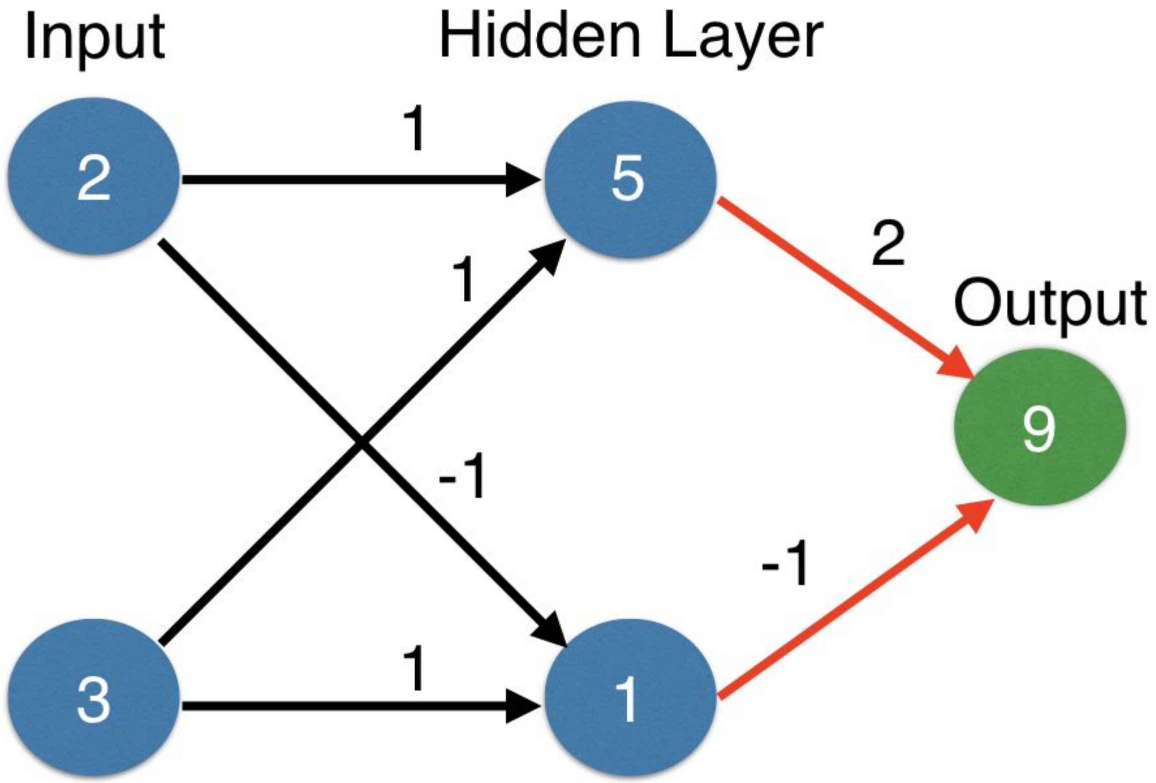


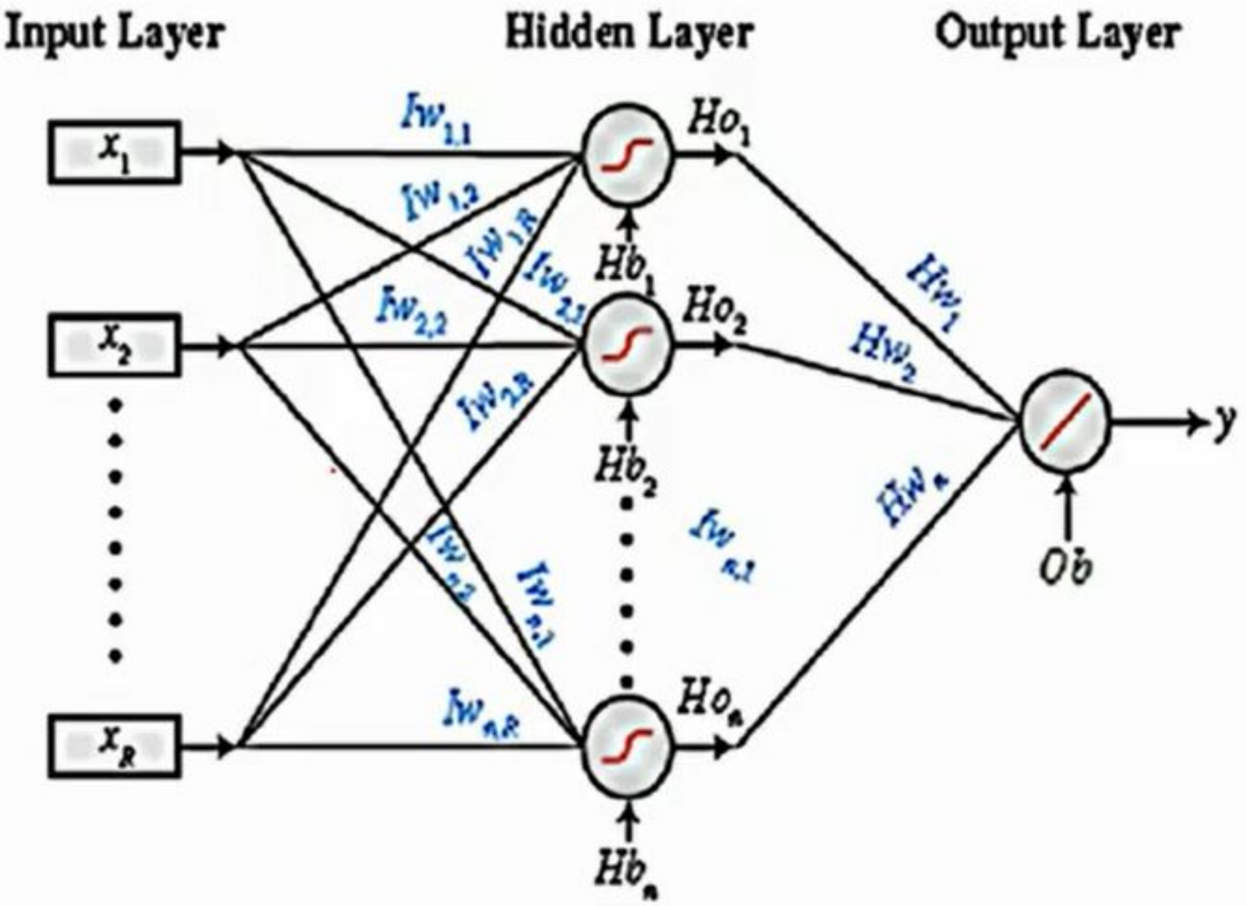
Tanh



Feedforward Propagation




Input
Hidden Layer
Output
2
3
5
1
9
1
1
-1
1
2
-1


Input Layer
Hidden Layer
Output Layer
x_1
x_2
x_R
$Iw_{1,1}$
$Iw_{1,2}$
$Iw_{1,R}$
$Iw_{2,1}$
$Iw_{2,2}$
$Iw_{2,R}$
$Iw_{n,2}$
$Iw_{n,1}$
$Iw_{n,R}$
Ho_1
Ho_2
Ho_n
Hb_1
Hb_2
Hb_n
Hw_1
Hw_2
Hw_n
Ob
y

Loss Function

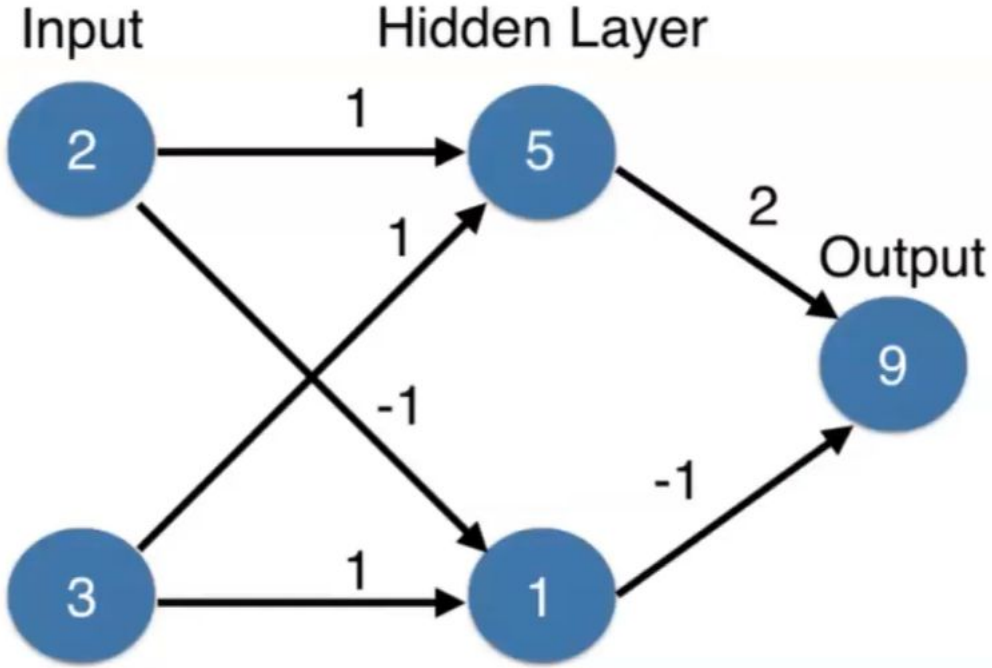
- Loss function is a method of evaluating how well your algorithm is modeling your dataset.

    - MSE(Mean Squared Error)
    - Binary cross-entropy
    - Categorical cross-entropy

Backward Propagation

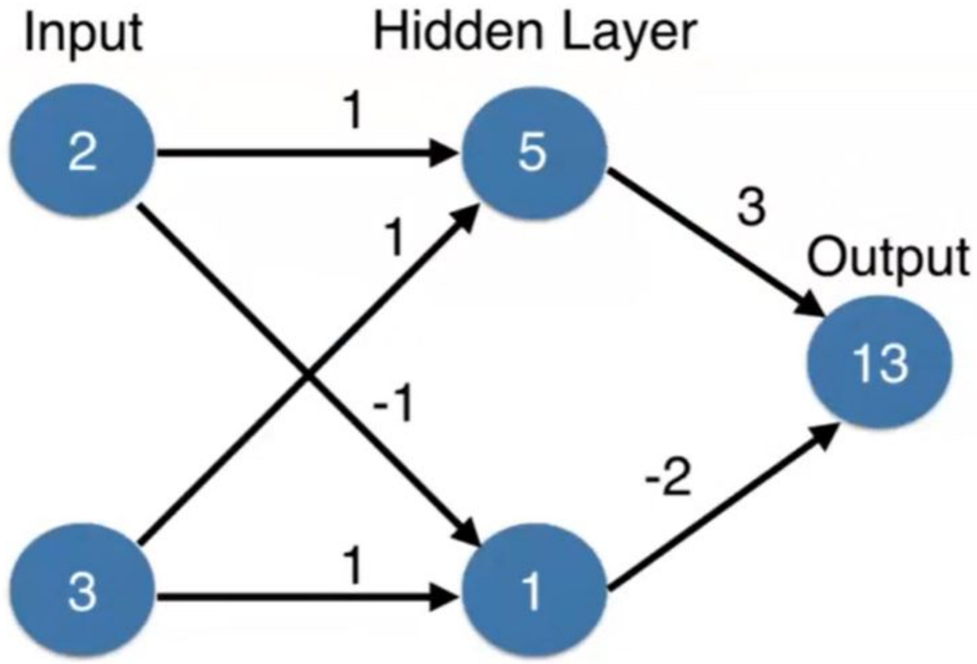
- Backward Propagation is the preferable method of adjusting or correcting the weights to reach the minimized loss function.
- Preferred method for adjusting the weights and biases since it is faster to converge as we move from output to the hidden layer.
- change the weights of the hidden layer that is closest to the output layer, re-calculate the loss and if further need to reduce the error then repeat the entire process and in that order move towards the input layer.

Backward Propagation



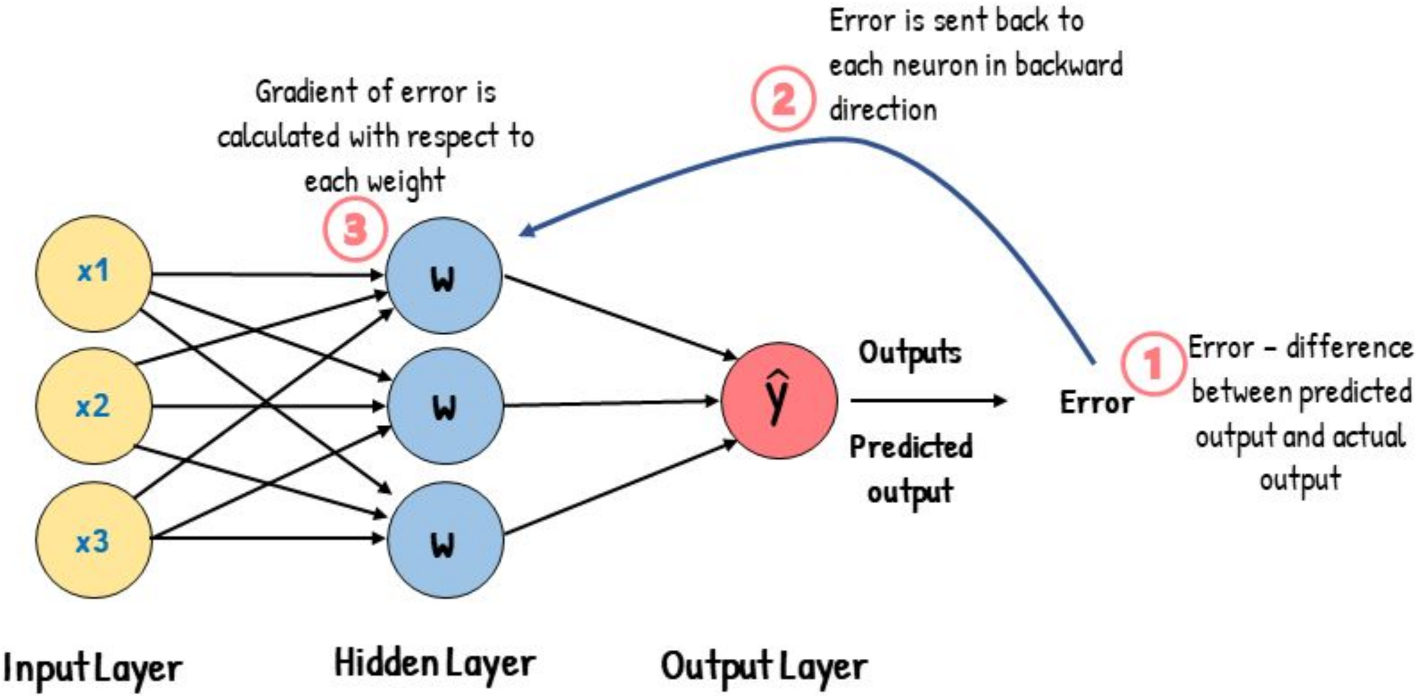
- Actual Value of Target: 13
- Error: Predicted - Actual = -4

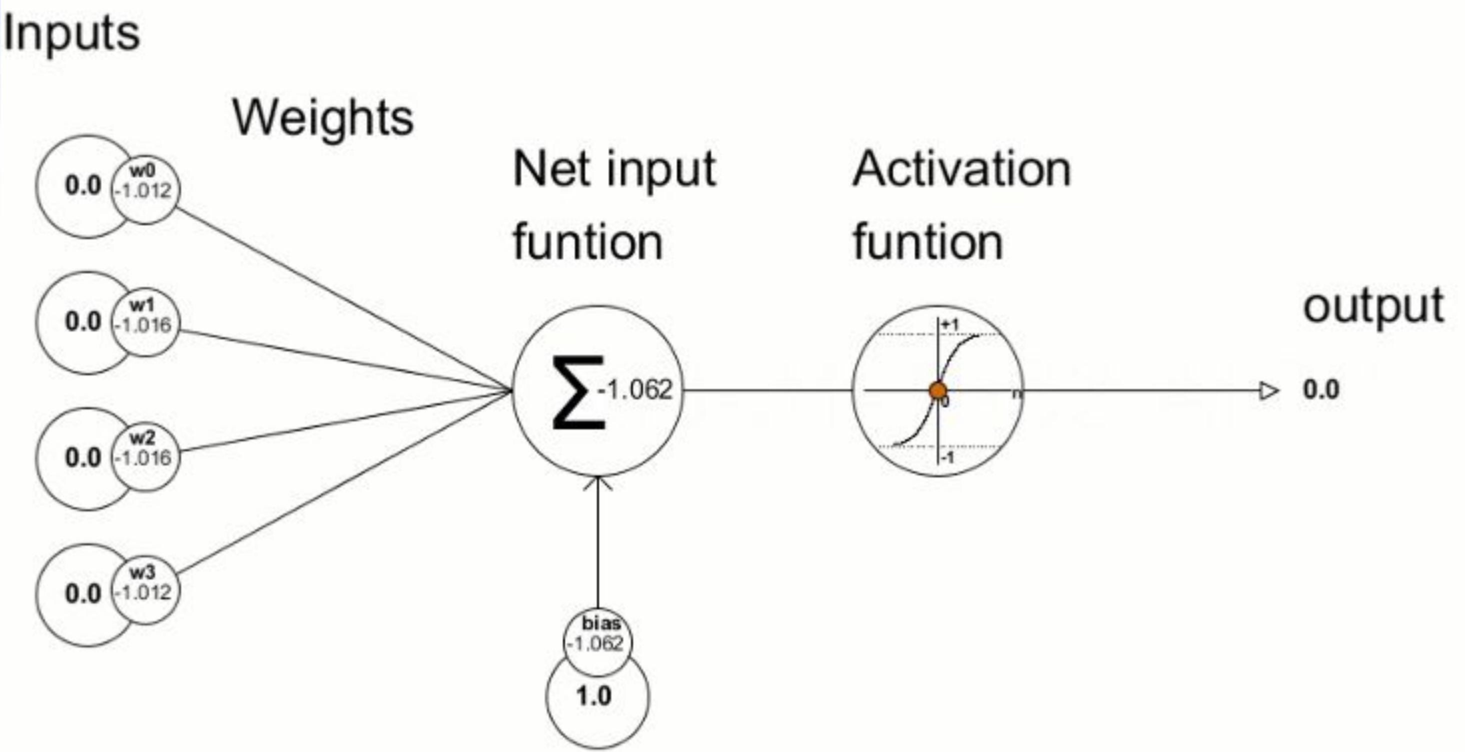
Backward Propagation



- Actual Value of Target: 13
- Error: Predicted - Actual = 0

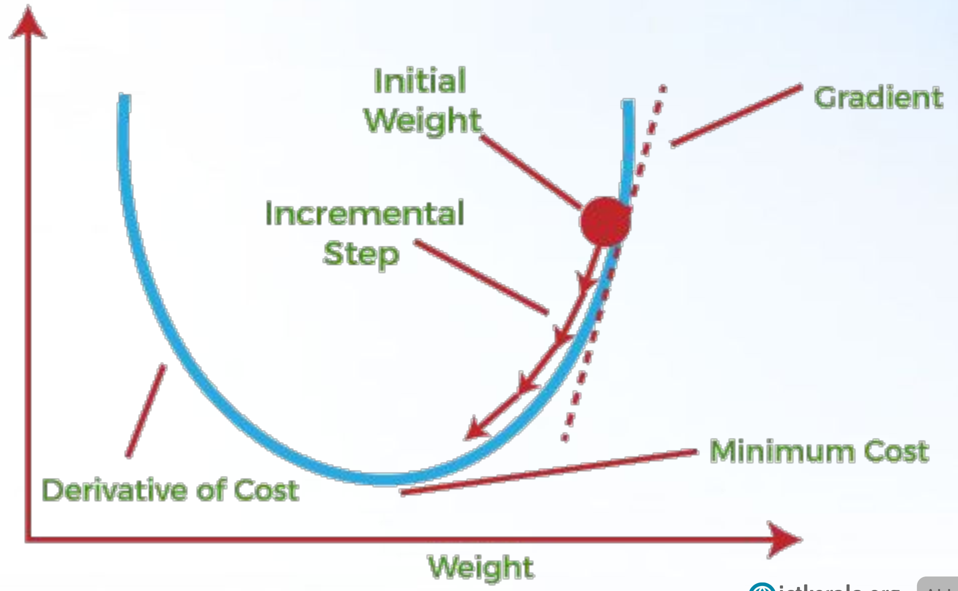
Backpropagation




Inputs
Weights
Net input
funtion
Activation
funtion
output
0.0
w0
-1.012
0.0
w1
-1.016
0.0
w2
-1.016
0.0
w3
-1.012
-1.062
+1
0
-1
0.0
bias
-1.062
1.0

Gradient Descent

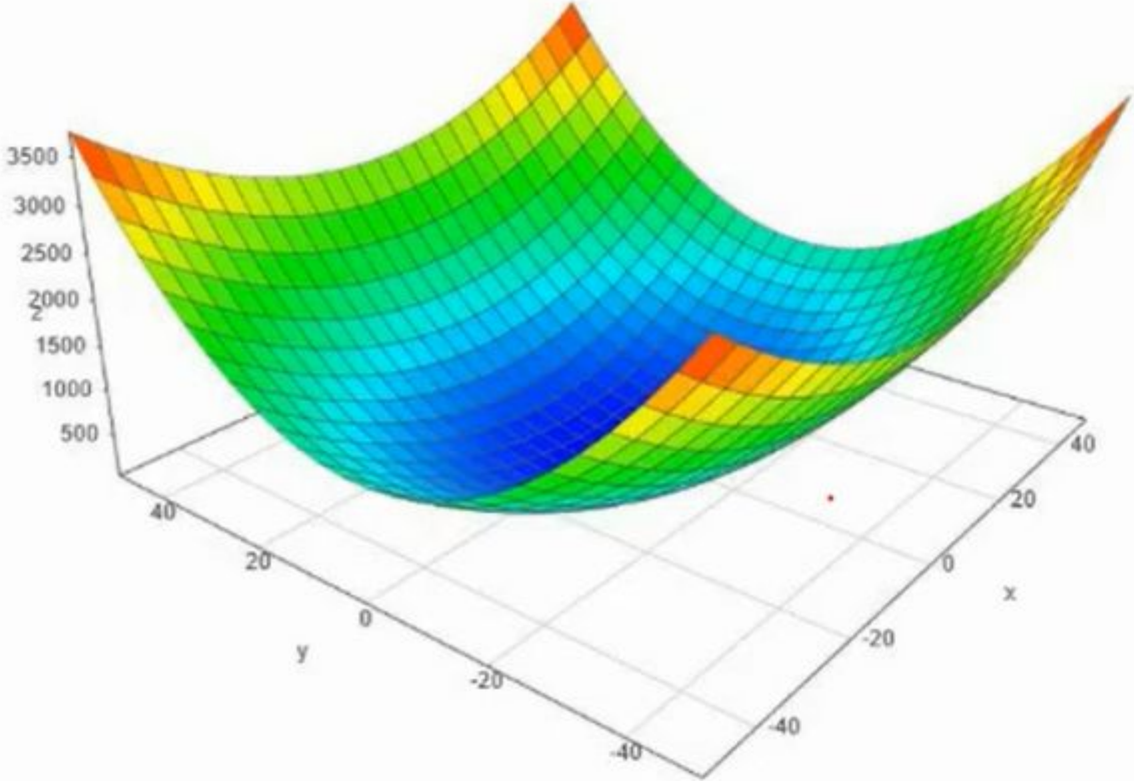
- Gradient Descent is known as one of the most commonly used optimization algorithms to train machine learning models by means of minimizing errors between actual and expected results.

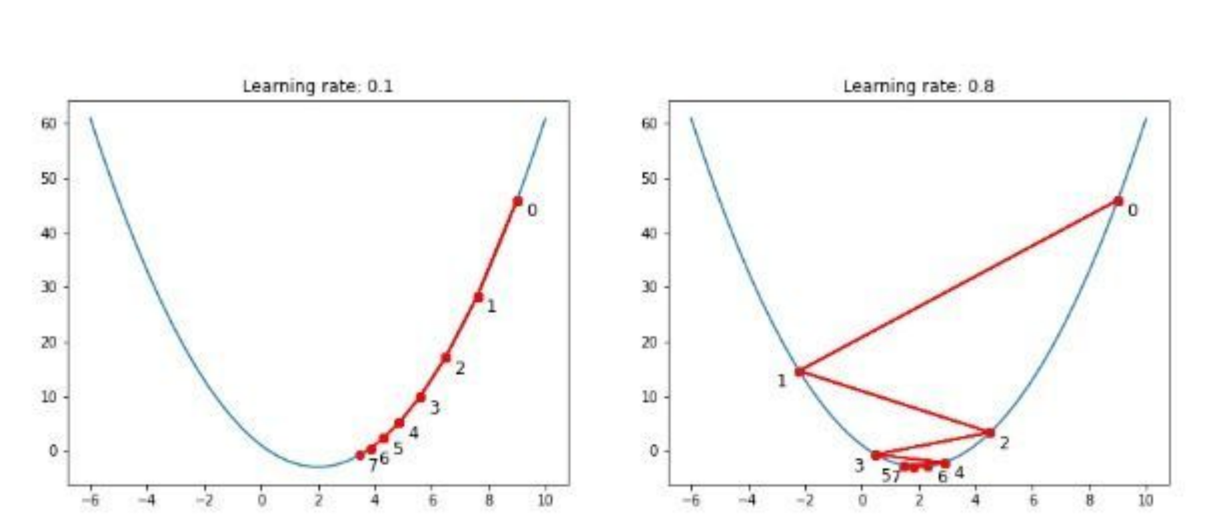


- The main objective of using a gradient descent algorithm is to minimize the cost function using iteration.
- It helps models find the optimal set of parameters by iteratively adjusting them in the opposite direction of the gradient.

    - Compute the gradient (slope), the first order derivative of the function at that point
    - Make a step (move) in the direction opposite to the gradient, opposite direction of slope increase from the current point by alpha times the gradient at that point.

$$f(x, y) = 0.5x^2 + y^2$$





Old weight

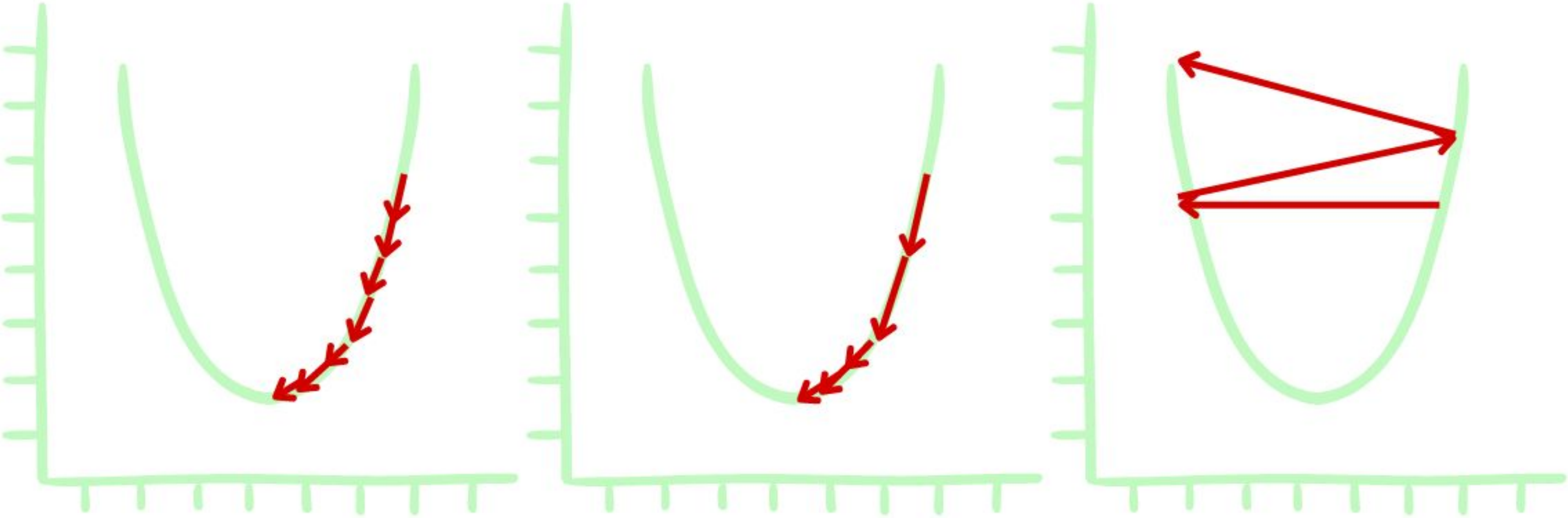
Derivative of Error with respect to weight

$$^*W_x = W_x - a\left(\frac{\partial Error}{\partial W_x}\right)$$

New weight

Learning rate

Too low | Just Right | Too Hight



Smaller learning rate requires many updates, hence learning would be very slow

Optimal learning rate smoothly reaches to the minima

Larger learning rate may lead to drastic updates and GD may diverge from the minima.

enjoyalgorithms.com