20MCA283 - DEEP LEARNING

Module I

Review of neural networks: Model of a biological neuron, McCulloch Pitts neuron, Activation functions, perceptron, perceptron learning algorithm & convergence, multilayer perceptron, back propogation, learning XOR, Sigmoid neurons, Gradient descent, feed forward neural network.

Module II

Training neural networks: Initialisation, dropout, batch normalisation & dropout, overfitting, underfitting, training & validation curves.

Data visualisation: feature & weight visualisation, tSNE.

Introduction to Tensorflow: graph, nodes, Tensor data structures - rank, shape, type, building neural networks with tensorflow, Introduction to Keras.

Module III

Convolutional neural networks: convolution operation, convolutional layers in neural network, pooling, fully connected layers.

Case study: Architecture of Lenet, Alexnet & VGG 16

Module IV

Recurrent neural networks: Back propogation, vanishing gradients, exploding gradiants, truncated backpropagation through time, Gated Recurrent units, Long short-term memory (LSTM) cells, Solving the vanishing gradient problems with LSTMs.

Module V

Autoencoders, variational autoencoders.

Generative adversarial networks (GAN): discriminative & generative models, GAN discriminator, GAN generator, upsampling, GAN training, GAN challenges, loss functions, cross entropy minimax loss, wasserstein loss.

16/11/21

→ AI is the way of making computer/robot/ software to think like human being.

→ John Maccarthy is the father of AI.

→ AI's goal is to implement human intelligence in computer.

Machine learning

- ML is the subset of AI
- father of ML is Arthur Samuel
- It provides the s/m to automatically learn & improve from experiences & predict - output without being explicitly programmed.

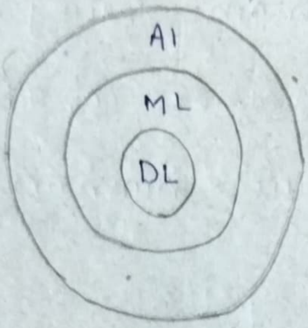
Eg: Face recognition, cyber fraud detection, self driving cars, friend suggestion in Facebook.

- It is a structured data representation.
- In ML, thousands of data points are used
- o/p is in the form of numerical data.

Deep learning

* DL is the subset of ML
* Jeffery Hinston is the father of DL
* DL uses structured model & is completely based on neurons.
* It can be called artificial Neural Network (ANN).
* ANN is used for data representation.
* Millions of data points are used.

* DL is used to Solve Complex programs.
* DL o/p can be numerical, audio o/p, text o/p etc.



Learning

SUPERVISED LEARNING | UNSUPERVISED LEARNING | REINFORCEMENT LEARNING

In Supervised learning, machine uses labelled data using labelled data, we are representing o/p in classification or regression

In Unsupervised learning, machine uses unlabelled data. Here grouping or clustering based on some similar patterns. It is called clustering or association.

Reinforcement learning is a feedback type, i.e, machine learn by doing action. Good actions get reward + bad actions get Penalty.

Good Action ⟶ Reward

Bad Action ⟶ Penalty

Raw i/p ⟶ ML algorithm ⟶ Mathematical model ⟶ (test the data) ⟶ output

new data ⟶ (test the data)

Raw i/p: (historical data) (Past data) (Training data)

ML algorithm: (using this learn training data)

Mathematical model: (Based on training data, mathematical model is created)

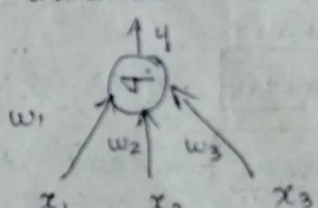
3/11/21

Neural Network

- are set of algorithm
- designed to recognise a pattern
- modelled as human brain
- It can take raw input
  ↓
  - can be labelled or unlabelled
  - clustering or grouping based on similar patterns.
- O/p is in the form of numerical data in vector format in which we can translate all the real world problems.
- It is used for feature extraction to fed into other algorithm.

Biological neuron

* fundamental unit of deep neural network is called artificial neurons.



* Artificial neurons are designed on ~~bass~~ of basis of biological neuron →
* Biological neurons are cells.

How biological neuron look like :-

1) DENDRILE → tree structure
→ receive signals from other neurons

2) SYNOPSE → point of connection to other neurons
→ connection neuron & gland
→ connection b/w neurons of muscles.

3) SOMA → part where processing is done.

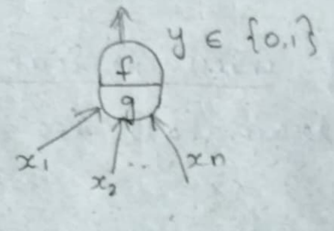
4) AXON → It transmits o/p to other parts of body

Eg: When a person hears a joke

Eg: Visualisation of an image

McCulloch Pitts Model

- High computational model of neurons



- g is the aggregate function
- f is the function in which decision based on g are made.
- Inputs → Excitatory / → Inhibitary
- INHIBITARY ~~Excitatory~~ means preventing action to get exact o/p ($y=0$)
- EXITATORY means more involved in producing o/p

$$g(x) = x_1 + x_2 + x_3 + \quad + x_n = \sum_{i=1}^{n} x_i \quad (y=1)$$

→ $y = f(g(x)) = 1$

$g(x) \longrightarrow$ Sum of Input

when $\boxed{g(x) \geq \theta}$

where θ = threshold Parameter | threshold logic

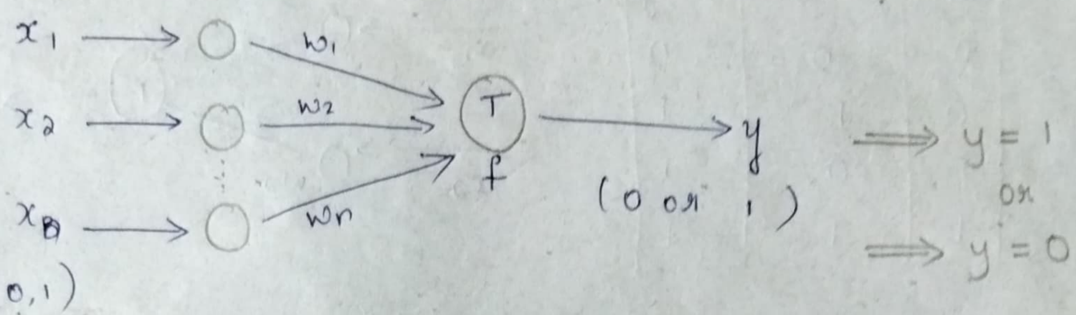
→ $y = f(g(x)) = 0$

when $\boxed{g(x) < \theta}$

23/11/2021

→ 2 layers

- I/p layer (any no. of I/p)
- O/p layer (only one O/p)



$g(x)$ = sum of weighted I/p

Output $y=1$, $\underline{\Sigma x_i w_i} \geq \theta$

$y=0$, $\Sigma x_i w_i < \theta$ → threshold value or threshold Parameter or BIAS → minimum value of weighted I/p from neuron to fire.

- Weight (w) can take +1, −1

+1 → Excitatory (Support the action)

−1 → Inhibitatory (Prevent the action)

Q. $x_1 = 1$ $\quad x_2 = 1$ $\quad x_3 = 0$

$w_1 = +1$ $\quad w_2 = +1$ $\quad w_3 = -1$

* → $x_1 w_1 = 1 \times +1 = +1$ $\quad x_2 w_2 = 1 \times +1 = +1$ $\quad x_3 w_3 = 0 \times -1 = 0$//

→ $\Sigma x_i w_i = x_1 w_1 + x_2 w_2 + x_3 w_3$

→ $\Sigma x_i w_i = 1 + 1 + 0 = 2$//

* Set threshold to set $y = 1$

Suppose $T = 1$, $\Sigma x_i w_i \geq T$

$2 \geq T$

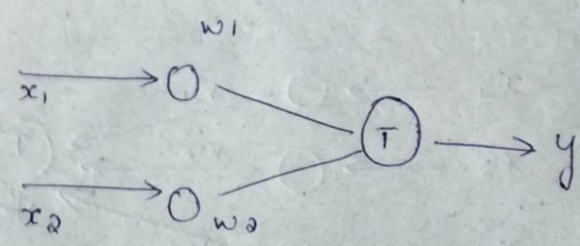
$2 \geq 1$

Condition Satisfied $\Rightarrow \boxed{y = 1}$

Implementation of $\boxed{\text{OR}}$ Gate

T.T

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1



① Suppose $w_1 = +1$ $w_2 = +1$ $\underline{T = +1}$

$(0,0) \rightarrow \Sigma x_i w_i = 0 \times +1 + 0 \times +1 = 0$ // $0 \geq 1$ ✗ $y = 0$

$(0,1) \rightarrow \Sigma x_i w_i = 0 \times +1 + 1 \times +1 = +1$ // $1 \geq 1$ ✓ $y = 1$

$(1,0) \rightarrow \Sigma x_i w_i = 1 \times +1 + 0 \times +1 = +1$ // $1 \geq 1$ ✓ $y = 1$

$(1,1) \rightarrow \Sigma x_i w_i = 1 \times +1 + 1 \times +1 = +2$ // $2 \geq 1$ ✓ $y = 1$

For OR Gate : $w_1 = +1$ $w_2 = +1$

$T = +1$

Implementation of $\boxed{\text{XOR}}$ Gate

T.T

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

① $w_1 = +1$ $w_2 = +1$

② $w_1 = +1$ $w_2 = -1$

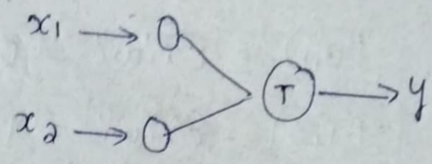
③ $w_1 = -1$ $w_2 = +1$

④ $w_1 = -1$ $w_2 = -1$

XOR can be written as $x_1 \cdot \bar{x_2} + \bar{x_1} \cdot x_2$

① $x_1 \cdot \bar{x_2}$ =

x_1	$\bar{x_2}$ x_2	y
0	0	0
0	1	0
1	0	1
1	1	0



① Suppose $w_1 = +1$ $w_2 = +1$ & $T = 1$

$(0,0) = \Sigma x_i w_i = 0 \times 1 + 0 \times 1 = 0$ $0 \geq 1$ X $y = 0$

$(0,1) = \Sigma x_i w_i = 0 \times 0 + 1 \times 1 = 1$ $1 \geq 1$ ✓ $y = 1$

$(1,0) = \Sigma x_i w_i = 1 \times 1 + 0 \times 1 = 1$ $1 \geq 1$ ✓ $y = 1$

$(1,1) = \Sigma x_i w_i = 1 \times 1 + 1 \times 1 = 2$ $2 \geq 1$ ✓ $y = 1$

Not satisfied with the original T.T.

② Suppose $w_1 = +1$ $w_2 = -1$ & $T = 1$

$(0,0) = \Sigma x_i w_i = 0 \times 1 + 0 \times -1 = 0$ $0 \geq 1$ X $y = 0$

$(0,1) = \Sigma x_i w_i = 0 \times 1 + 1 \times -1 = -1$ $-1 \geq 1$ X $y = 0$

$(1,0) = \Sigma x_i w_i = 1 \times 1 + 0 \times -1 = 1$ $1 \geq 1$ ✓ $y = 1$

$(1,1) = \Sigma x_i w_i = 1 \times 1 + 1 \times -1 = 0$ $0 \geq 1$ X $y = 0$

So condition satisfied

for XOR Gate : $w_1 = +1$ $w_2 = -1$
$(x_1 \cdot \bar{x_2})$ $T = 1$

② $\bar{x_1} \cdot x_2$ =

x_1	x_2	y
0	0	0
0	1	1
1	0	0
1	1	0

(3) Suppose $w_1 = -1$ & $w_2 = +1$ & $T = 1$

$(0,0) = \Sigma x_i w_i = 0 \times -1 + 0 \times +1 = 0$ $\quad 0 \geq 1$ ✗ $\quad y = 0$

$(0,1) = \Sigma x_i w_i = 0 \times -1 + 1 \times 1 = 1$ $\quad 1 \geq 1$ ✓ $\quad y = 1$

$(1,0) = \Sigma x_i w_i = 1 \times -1 + 0 \times 1 = -1$ $\quad -1 \geq 1$ ✗ $\quad y = 0$

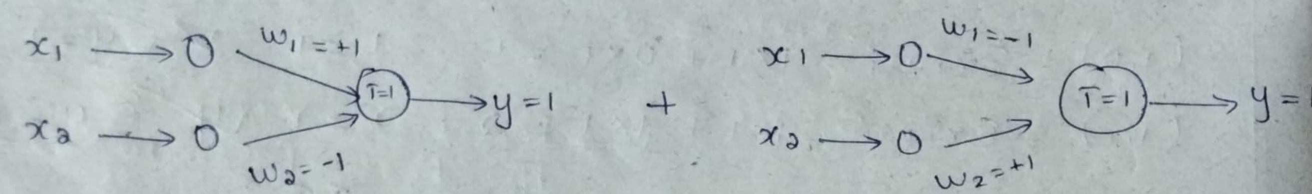
$(1,1) = \Sigma x_i w_i = 1 \times -1 + 1 \times 1 = 0$ $\quad 0 \geq 1$ ✗ $\quad y = 0$

Condition Satisfied

For XOR gate $= w_1 = -1 \quad w_2 = +1$

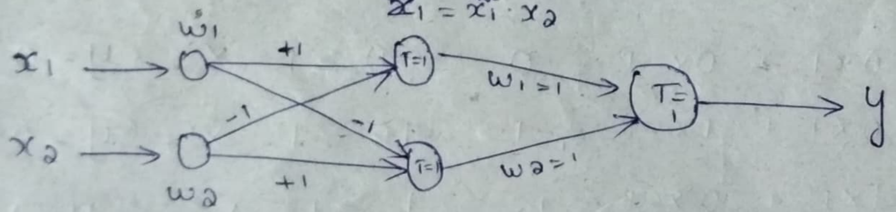
$\bar{x}_1 \cdot x_2 \qquad T = 1$

XOR =



We use extra one layer called hidden layer because we cannot implement XOR only using the above layer

$Z_1 = x_1 \cdot \bar{x}_2$



$Z_2 = \bar{x}_1 \cdot x_2$

$$XOR = \underbrace{x_1 \cdot \bar{x}_2}_{Z_1} + \underbrace{\bar{x}_1 \cdot x_2}_{Z_2} \qquad (Z_1 \text{ OR } Z_2)$$

$\hookrightarrow w_1 = 1$

$w_2 = 1$

$T = 1$

HW Implementation of $\boxed{\text{AND}}$ gate

T.T	x_1	x_2	y
	0	0	0
	0	1	0
	1	0	0
	1	1	1

① Suppose $w_1 = +1$ $w_2 = +1$ $T = +1$

$(0,0) = \Sigma x_i w_i = 0\times1 + 0\times1 = 0$ $0 \geq 1$ × $y=0$

$(0,1) = \Sigma x_i w_i = 0\times1 + 1\times1 = 1$ $1 \geq 1$ ✓ $y=1$

$(1,0) = \Sigma x_i w_i = 1\times1 + 0\times1 = 1$ $1 \geq 1$ ✓ $y=1$

$(1,1) = \Sigma x_i w_i = 1\times1 + 1\times1 = 2$ $2 \geq 1$ ✓ $y=1$

Condition not satisfied.

② Suppose $w_1 = +1$ $w_2 = -1$ $T = +1$

$(0,0) = \Sigma x_i w_i = 0\times1 + 0\times-1 = 0$ $0 \geq 1$ × $y=0$

$(0,1) = \Sigma x_i w_i = 0\times1 + 1\times-1 = -1$ $-1 \geq 1$ × $y=0$

$(1,0) = \Sigma x_i w_i = 1\times1 + 0\times-1 = 1$ $1 \geq 1$ ✓ $y=1$

$(1,1) = \Sigma x_i w_i = 1\times1 + 1\times-1 = 0$ $0 \geq 1$ × $y=0$

Condition not satisfied.

③ Suppose $w_1 = -1$ $w_2 = +1$ $T = +1$

$(0,0) = \Sigma x_i w_i = 0\times-1 + 0\times1 = 0$ $0 \geq 1$ × $y=0$

$(0,1) = \Sigma x_i w_i = 0\times-1 + 1\times1 = 1$ $1 \geq 1$ ✓ $y=1$

$(1,0) = \Sigma x_i w_i = 1\times-1 + 0\times1 = -1$ $-1 \geq 1$ × $y=0$

$(1,1) = \Sigma x_i w_i = 1\times-1 + 1\times1 = 0$ $0 \geq 1$ × $y=0$

Condition not satisfied.

4. Suppose $w_1 = -1$ $w_2 = -1$ $T = 1$

$(0,0) = \Sigma x_i w_i = 0 \times -1 + 0 \times -1 = 0$ $0 \geq 1$ × $y = 0$

$(0,1) = \Sigma x_i w_i = 0 \times -1 + 1 \times -1 = -1$ $-1 \geq 1$ × $y = 0$

$(1,0) = \Sigma x_i w_i = 1 \times -1 + 0 \times -1 = -1$ $-1 \geq 1$ × $y = 0$

$(1,1) = \Sigma x_i w_i = 1 \times -1 + 1 \times -1 = -2$ $-2 \geq 1$ × $y = 0$

Condition not satisfied

5. Suppose $w_1 = +1$ $w_2 = +1$ $T = 2$

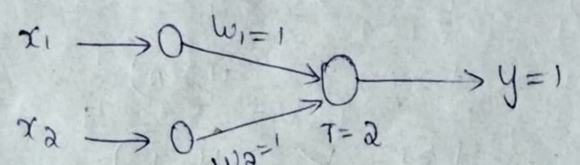
$(0,0) = \Sigma x_i w_i = 0 \times 1 + 0 \times 1 = 0$ $0 \geq 2$ × $y = 0$

$(0,1) = \Sigma x_i w_i = 0 \times 1 + 1 \times 1 = 1$ $1 \geq 2$ × $y = 0$

$(1,0) = \Sigma x_i w_i = 1 \times 1 + 0 \times 1 = 1$ $1 \geq 2$ × $y = 0$

$(1,1) = \Sigma x_i w_i = 1 \times 1 + 1 \times 1 = 2$ $2 \geq 2$ ✓ $y = 1$

Condition satisfied.



For AND condition, ~~$x_1 = 1$~~ $w_1 = 1$ $w_2 = 1$ & $T = 2$

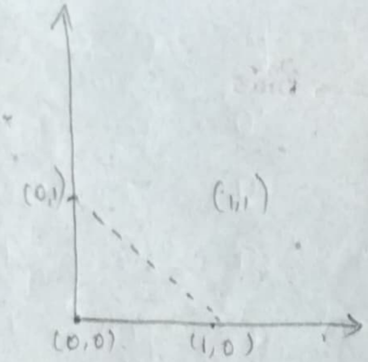
1/12/2021

Geometrical Representation of OR

$y = 1$ if $\sum_{i=1}^{n} x_i \geq \theta$

$y = 0$ if $\sum_{i=1}^{n} x_i < \theta$

0	0	0
0	1	1
1	0	1
1	1	1



Binary data are splitted into 4 portions such that inputs producing $y=1$ will be on one side & $y=0$ will be on other side.

Single neuron can be used to represent boolean function which is linearly separable (using MP)

For 2 i/p $\longrightarrow$ there exist a line

For 3 i/p $\longrightarrow$ there exist a plane

$\downarrow$

Similar to line,
$y=0$ on one side &
$y=1$ on other side

For non-boolean,

Classical Perception Model

Perceptron $\longrightarrow$ Simple model based on biological model used in ANN.

- This model was refined & analysed by Minsky (1969) & Papert (1969). Their model was referred as PERCEPTRON MODEL.

- Using this model, we can represent
  - $\rightarrow$ non-boolean
  - $\rightarrow$ non-linearly separable
  - $\rightarrow$ real data
  - $\rightarrow$ focus on weightage

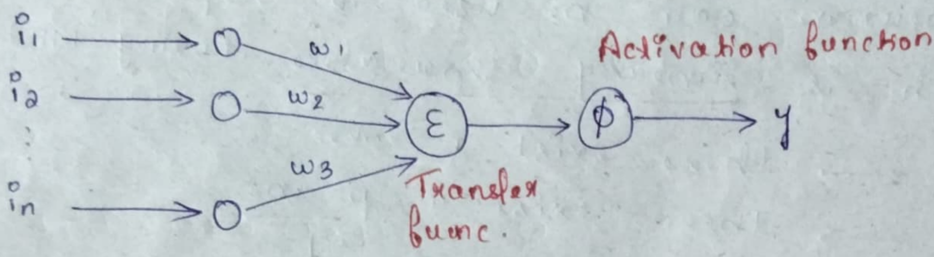
Here $y = 1$ if $\sum_{i=1}^{n} x_i w_i \geq \theta$ → Bias

$y = 0$ if $\sum_{i=1}^{n} x_i w_i < \theta$

or

$y = 1$ if $\sum_{i=1}^{n} x_i w_i - \theta \geq 0$

$y = 0$ if $\sum_{i=1}^{n} x_i w_i - \theta < 0$



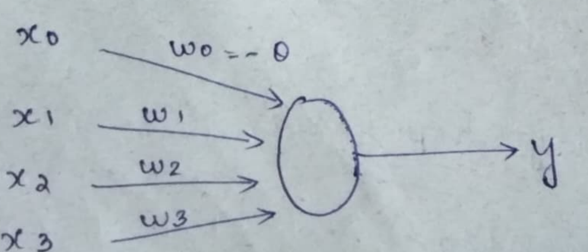
Activation function : It is a func. used in ANN which outputs a small value when i/p are small. & it produces a larger value as o/p when i/ps exceeds threshold value. (used to produce a o/p)

i/p	o/p
Small	Small
exceeds θ	large

A more accepted convention :

- $y = 1$, if $\sum_{i=0}^{n} x_i w_i \geq 0$, where $x_0 = 1$ & $w_0 = -\theta$ Bias value of threshold.

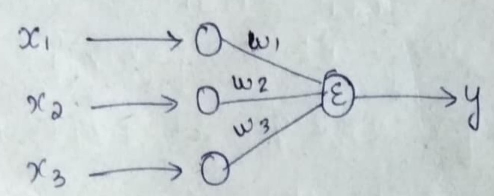
Assume, $x_0 = 1$ & $w_0 = -\theta$, then



- $y = 0$, if $\sum_{i=1} x_i w_i < 0$, where $x_0 = 1$ & $w_0 = -\theta$

Eg: PREDICTION → Based on past data

Task of predicting of liking a movie or not?



x_1 = actor A
x_2 = thriller
x_3 = director B

for example, x_1 = not A x_2 = not thriller x_3 = dir B, we can make $y=1$ by giving more weightage to x_3.

Movie buf = y = θ = 0

Selective viewer = y = (3) θ = 3

OR operation:

- Prediction based on past data.
- I/p, weigh & θ depend on these past data

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

$0\ 0\ 0 \rightarrow w_0 + \sum_{i=1}^{2} x_i w_i < 0$

$\left.\begin{matrix}0\ 1\ 1\\1\ 0\ 1\\1\ 1\ 1\end{matrix}\right\} \rightarrow w_0 + \sum_{i=1}^{2} x_i w_i \geq 0$

i) $w_0 + 0 \times w_1 + 0 \times w_2 < 0$

$\Rightarrow w_0 < 0$ ✓

ii) $w_0 + 0 \times w_1 + 1 \times w_2 \geq 0$

$w_0 + w_2 \geq 0 \Rightarrow w_2 \geq -w_0$

iii) $w_0 + 1 \times w_1 + 0 \times w_2 \geq 0$

$w_0 + w_1 \geq 0 \Rightarrow w_1 \geq -w_0$

iv) $w_0 + 1 \times w_1 + 1 \times w_2 \geq 0$

$w_0 + w_1 + w_2 \geq 0 \Rightarrow w_1 + w_2 \geq -w_0$

7/12/

Perception learning algorithm

x_1	x_2	
0	0	0
0	1	1
1	0	1
1	1	1

Q. $w_0 = -1$, $w_1 = 1.1$, $w_2 = 1.1$

- $w_0 + x_1 w_1 + x_2 w_2 < 0$

$-1 + x_1 1.1 + x_2 1.1 < 0$

$-1 + 1.1 x_1 + 1.1 x_2 < 0$

case 1 $(x_1 = 0, x_2 = 0)$

$-1 + 0 \times 1.1 + 0 \times 1.1 < 0$

$-1 < 0$ ✓ $y = 0$

case 2

$-1 + 0 \times 1.1 + 1 \times 1.1 < 0$

$0.1 > 0$ ✓ $y = 1$

$-1 + 1.1$

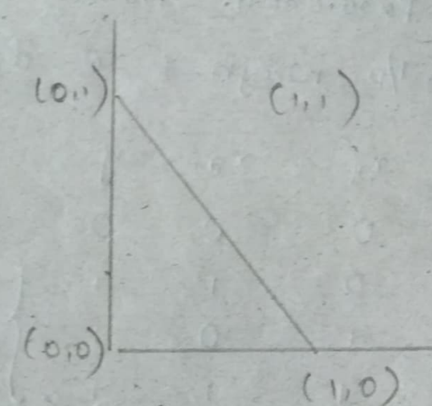
0.1

$-1 + 1.1$

case 3

$-1 + 1 \times 1.1 + 0 \times 1.1 < 0$

$0.1 > 0$ ✓ $y = 1$



case 4

$-1 + 1 \times 1.1 + 1 \times 1.1 > 0$ ✓ $y = 1$

Q. $w_0 = -1$, $w_1 = -1$, $w_2 = -1$

- $w_0 + x_1 w_1 + x_2 w_2 < 0$

case 1 $(x_0 = 0 , x_1 = 0)$

$-1 + 0 \times -1 + 0 \times -1 < 0$

$-1 < 0$ $y = 0$

case 2

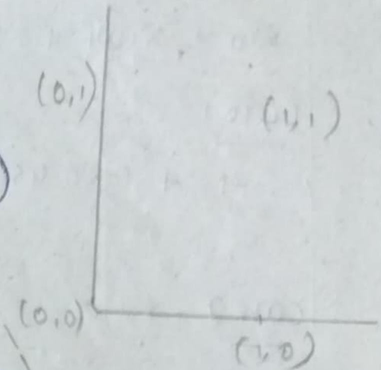
$-1 + 0 \times -1 + 1 \times -1 < 0$

$-2 < 0$ $y = 0$

<u>case 3</u>

$-1 + 1\times -1 + 0\times -1 < 0$

$-2 < 0$ $y=0$



<u>case 4</u>

$-1 + 1\times -1 + 1\times -1 < 0$

$-3 < 0$ $y = 0$

Not satisfying. (3 errors)

Q. $w_0 = -1$ $w_1 = 1.5$ $w_2 = 0$

$w_0 + w_1 x_1 + x_2 w_2 < 0$

<u>case 1</u>

$-1 + 0\times 1.5 + 0\times 0 < 0$

$-1 < 0$ $y=0$ ✓

<u>case 2</u>

$-1 + 0\times 1.5 + 1\times 0 < 0$

$-1 < 0$ $y=0$ ✗

<u>case 3</u>

$-1 + 1\times 1.5 + 0\times 0 < 0$

$0.5 \leq 0$

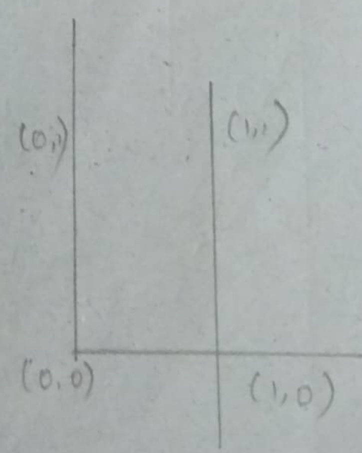
$0.5 > 0$ $y=1$ ✓

<u>case 4</u>

$-1 + 1\times 1.5 + 0\times 0 < 0$

$0.5 < 0$

$0.5 > 0$ $y=1$ ✓



1 error. Not satisfying

Q. $w_0 = -1, \quad w_1 = 0.45 \quad w_2 = 0.45$

$w_0 + x_1 w_1 + x_2 w_2 \geq 0$

<u>case 1</u>

$-1 + 0 \times 0.45 + 0 \times 0.45 \leq 0$

$-1 < 0 \quad y = 0$ ✓

<u>case 2</u>

$-1 + 0 \times 0.45 + 1 \times 0.45$

$-1 + 0.45 < 0$

$-0.55 < 0 \quad y = 0$ ×

100

<u>case 3</u>

$-1 + 1 \times 0.45 + 0 \times 0.45$

$-1 + 0.45 < 0$

$-0.55 < 0 \quad y = 0$ ×

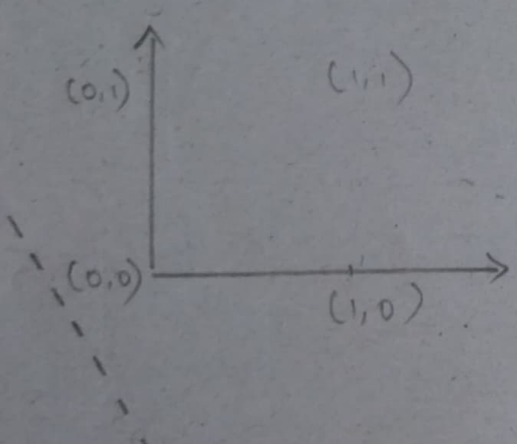
<u>case 4</u>

$$\begin{array}{r} 0.45 \\ 0.45 \\ \hline 0.90 \\ -1 \end{array}$$

$-1 + 1 \times 0.45 + 1 \times 0.45$

$-0.10 < 0 \quad y = 0$ ×

3 errors.



Activation function (transfer function)

> It decides whether a neuron should be activated or not by calculating weighted sum further adding bias with it ($w_0 + x_1w_1 + x_2w_2$)

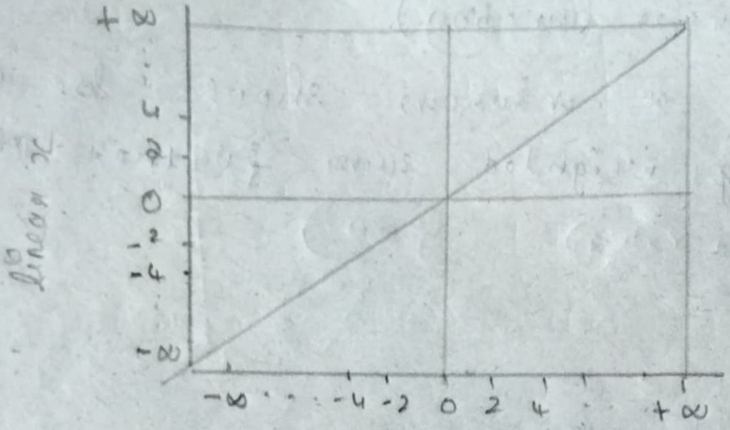
Why we use activation function?

* To determine the o/p of neural network like 1/0, YES/NO.
* It maps the resulting value in b/w 0 to 1 or from -1 to 1 depending on the activation func. (Range will be dependent)
* The purpose of activation fnc. is to introduce non-linearity into the o/p of the neuron
* Neural n/w have neurons that work in correspondance of weight, bias & respective activation point.
* update the weights & bias of the neuron on the basis of the errors at the o/p. This process is known as BACK PROPAGATION.
* Activation func make BACK PROPOGATION POSSIBLE.
* Activation function can be basically divided into 2

1) linear or identity activation func.
2) Non-linear activation function

- Linear (Identity) activation function

> This func is a line or linear ∴ o/p of the func. will not be confined or restricted b/w any range. $(-\infty, +\infty)$

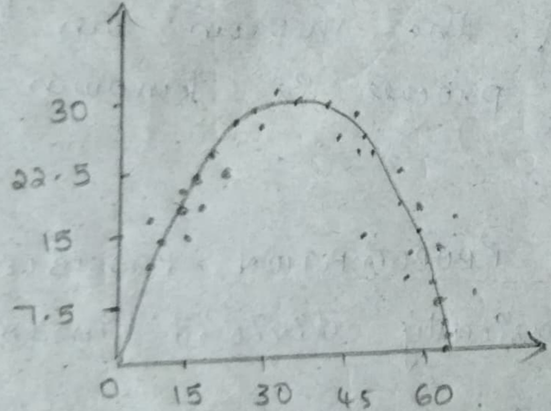


Here range is -∞ to ∞.

$f(x) = x$

- Non-linear activation functions

> These are the most used activation func.
> It makes easy for the model to generalise or adapt with varieties of data values & differentiate b/w o/p.
> focus on solving complex task.



we don't get a straight line
we get a curve

Terminologies associated with non-linear function

① Derivative or Differential

* Changing y-axis w.r.t. changing x-axis.
* also known as slop

② Monotonic function

* A func. which is ~~either~~ ~~not~~ either entirely

decreasing or non-increasing.

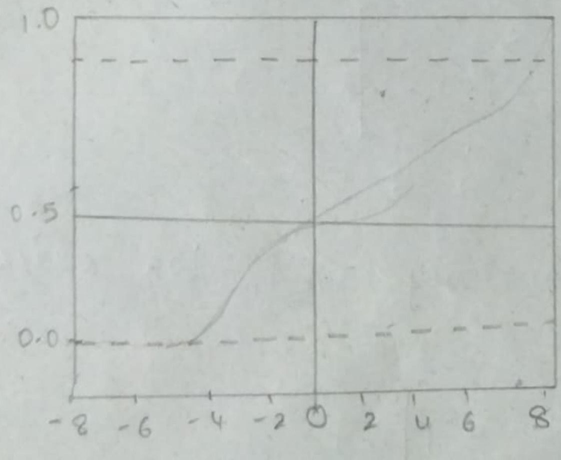
The non-linear activation function are categorised on the basis of RANGE or CURVE.

It can be categorised into

1) sigmoid (logistic) activation func.
2) TANH or Hyperbolic Tangent activation func.
3) ReLU or Rectified Linear Unit
4) Leaky ReLU

1) SIGMOID Activation func.

* looks like S shaped Curve.
* Range is from 0 to 1
* function is differential and monotonic.
* Used to predict probability
* func. has derivative which is not monotonic.



* $\boxed{\text{Softmax}}$ func. is a more generalised logistic activation func. used for multi-class classification.

Why derivative or differentiation used?

→ when updating the curve, to know in which direction & how much to change or to update the curve depending upon the slope.

→ We use differentiation in almost every part of ML & DL

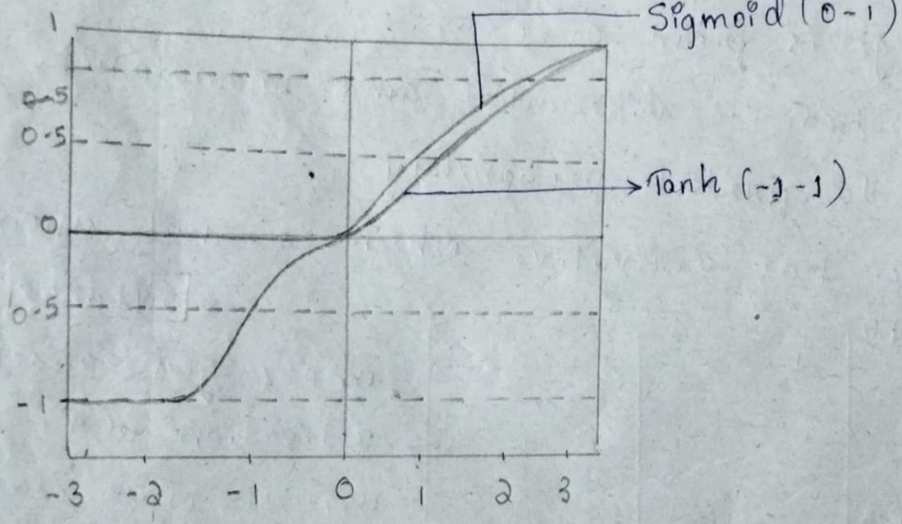
$$\boxed{\sigma(z) = \frac{1}{1+e^{-z}}}$$

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

derivative $f'(x) = f(x) \cdot (1 - f(x))$

2) TANH or hyperbolic Tangent Activation func.

- Similar to sigmoid, S shaped
- range from -1 to 1
- The function is differential - monotic while derivative is not monotonic.
-



3) • The Tanh func is mainly used for classification b/w two classes. (binary)

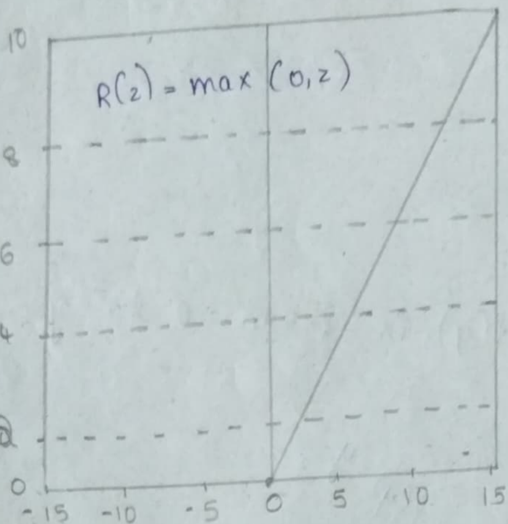
Equation = $\boxed{f(x) = \frac{2}{1+e^{-2x}} - 1}$

derivation = $f'(x) = 1 - f(x)^2$

• Both Tanh + Sigmoid Activation function are used in feed-forward neural n/w.

3) ReLu

* most frequently or commonly used activation func.
* used in CNN (convolutional neural Network)
* Range is 0 to ∞.
* Both function & derivative are monotonic



* It acquires Half - Rectified. (No -ve values)
* $f(z) = 0$ when $z<0$ (negative).
* $f(z) = z$ when $z>=0$
* Equation : $f(x) = \begin{cases} 0 & \text{for } x<0 \\ 1 & \text{for } x \geq 0 \end{cases}$

derivation : $f'(x) = \begin{cases} 0 & \text{for } x<0 \\ 1 & \text{for } x \geq 0 \end{cases}$

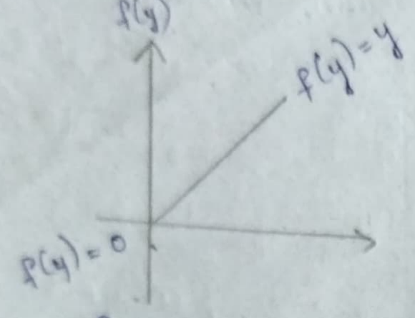
* Issue :

> All the negative values become zero. Immediately in the graph which in turn affects the resulting graph by not mapping -ve values appropriately.

> It decreases the ability of the model to fit or train from the data properly.

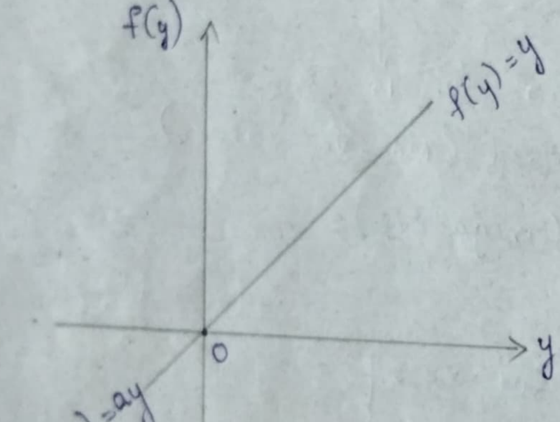
- To solve dying Relu problem, Leaky Relu was introduced
- It helps to increase the range of Relu function.
- Taking the parameter 'α' = 0.01.
- Range is $-\infty$ to ∞.

↳ increasing factor or parameter



Relu

- ~~Here~~ f in Relu, f(y) is from 0 to y.



- Monotonic (for func. & derivative) in nature.
- Equation : $f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$

derivative : $f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$

- If $a = 0.01$ → Leaky Relu

→ If parameter value, $a \neq 0.01$ → $\underbrace{\text{Randomised Relu}}$

↓

differential & derivative are monotonic

15/12/2021

Single layer Perceptron

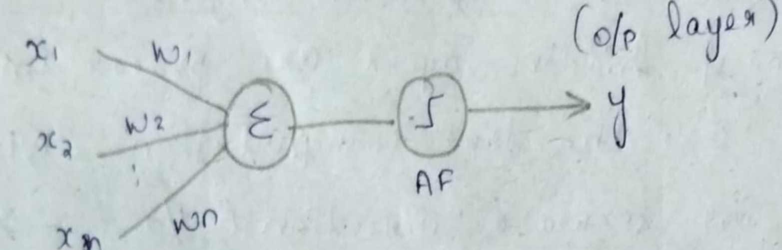
→ It is a neural network with a set of inputs.

→ It consist of two layers.

1. Input layer
2. Output layer

- Produce single binary o/p.

→ Each input have weight ↳ used to decide the o/p



$x_i w_i \geq 0$

→ Add biased value (w_0) to sum of weighted input.

Multi Layer Perceptron (MLP)

- Most complicated structure
- It consist of multiple layer & therefore known as multilayer Perceptron.
- Multiple layers of n/w of perceptron.
- Consist of 3 layers

1. input layer (single)
2. hidden layers (any no of)
3. Ouput layer (single)