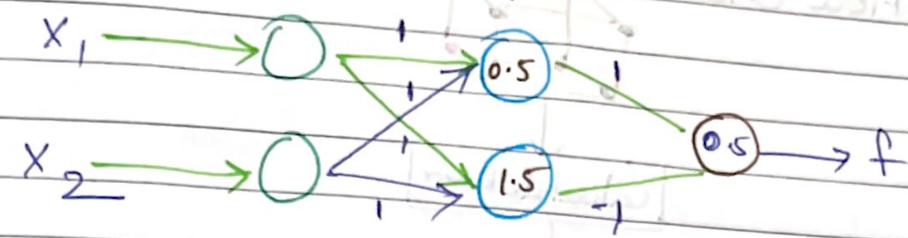
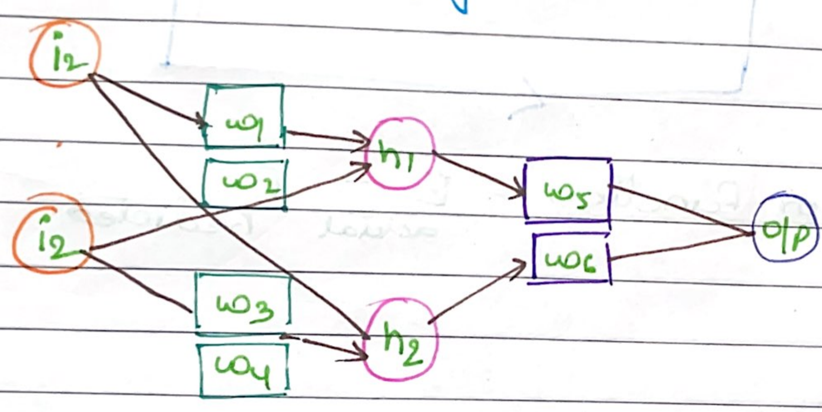
⇒ linear non-separable.



⇒ MLFFNN should be used

BACK PROPOGATION

→ Why Back Propogation



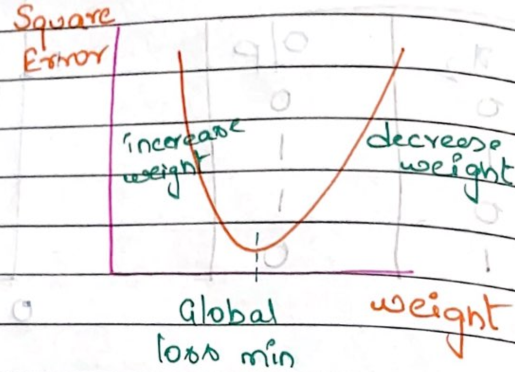
→ 1. The weights assigned will lead to correct o/p

2. Intialize weights

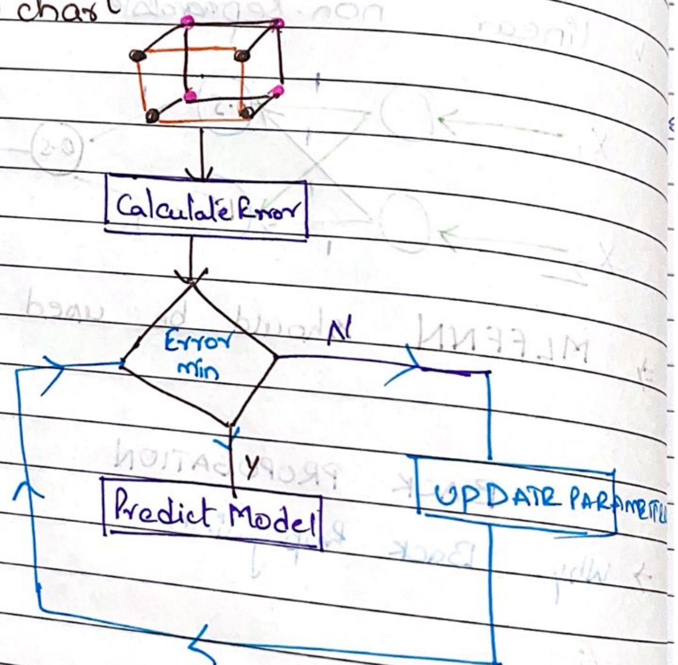
3. change parameters that error is min.

Error = E_{real} − $E_{predicted}$

Gradient Descent :-



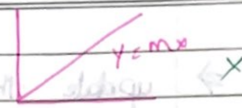
Flow chart



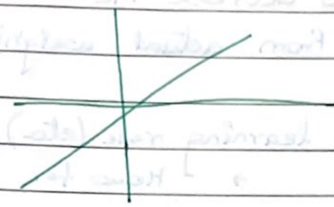
Error function = $E_{actual} - E_{Predicted}$

Why bias.

→ $y = w_1 * x + b$



Y-intercept



$y = w_1 x + b$.

→ Diagram in PPT

Step 1 :-

Forward Pass.

$net\,h_1 = w_1 \times i_1 + w_2 \times i_2 + 1 \times b_1$

$= 0.3775$

$net\,h_2 = i_2 w_3 + i_2 \times w_4 + 1 \times b_1$

⇒ Squash = (o/p from 0 to 1)

$out\,h_1 = \frac{1}{1+e^{-(net\,h_1)}}$

Step 2

→ $o_1 = h_1 \times w_5 + h_2 \times w_6 + b_2$

$o_2 = h_1 \times w_6 + h_2 \times w_7 + b_2$

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

$\frac{d}{dx} f(x) = \frac{e^{-x} \cdot (1+e^{-x}) - e^{-x} \cdot e^{x}}{(1+e^{-x})^2} = \frac{e^{x}}{(1+e^{x})^2} = f(x)(1-f(x))$

DATE

⇒ update the weight

→ To decrease the error, Subtract this from actual weight

η = learning rate (eta)

→ How far to go?

⇒ Unsupervised Learning.

Competitive Neural Network

→ Form of unsupervised learning in ANN, nodes compete for the right to respond to the subset of input data

→ increasing the specialization of each node in the n/w.

⇒ Hebbian Learning

→ Proposed by Donald Hebb

$A \xrightarrow{w_{AB}} B$

1. Neuron A is near B, fires or excites B, metabolic changes happen both in A and B, (A efficiency in B increases)

2. If A and B are simultaneously excited (Synaptic strength increases)

3. If A and B are activated unsynchronised, (strength of synapse decreases)

VISION

PAGE