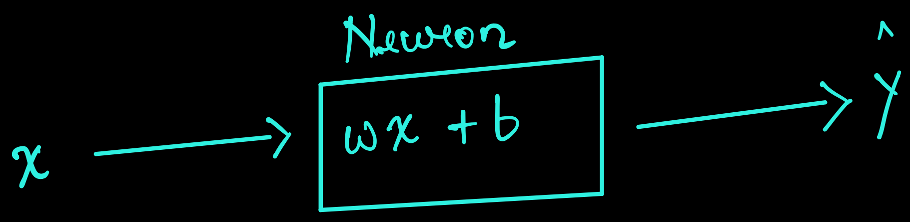
Today's Agenda

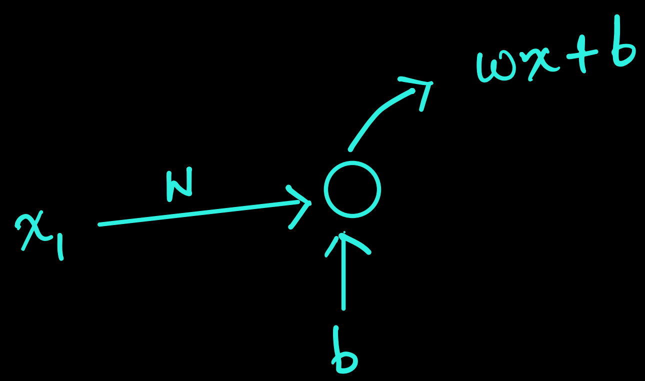
Perceptron Model

$\hat{y}$ = prediction

y = actual



$$x = \begin{Bmatrix} x_0 \\ x_1 \\ x_2 \end{Bmatrix}$$



Loss Calculation

$$\text{Squared Error} = (\hat{y} - y)^2 = \text{Loss}$$

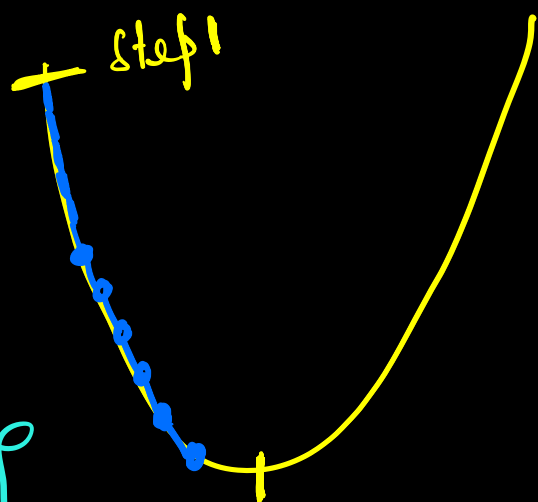
Error $\longrightarrow$ Reducing the error close to 0

$x \rightarrow$ Fixed

$\{w, b\} \rightarrow$ change / learnable params

Tune $\{w, b\}$ so that the loss comes close to zero.

<u>Update w & b</u>

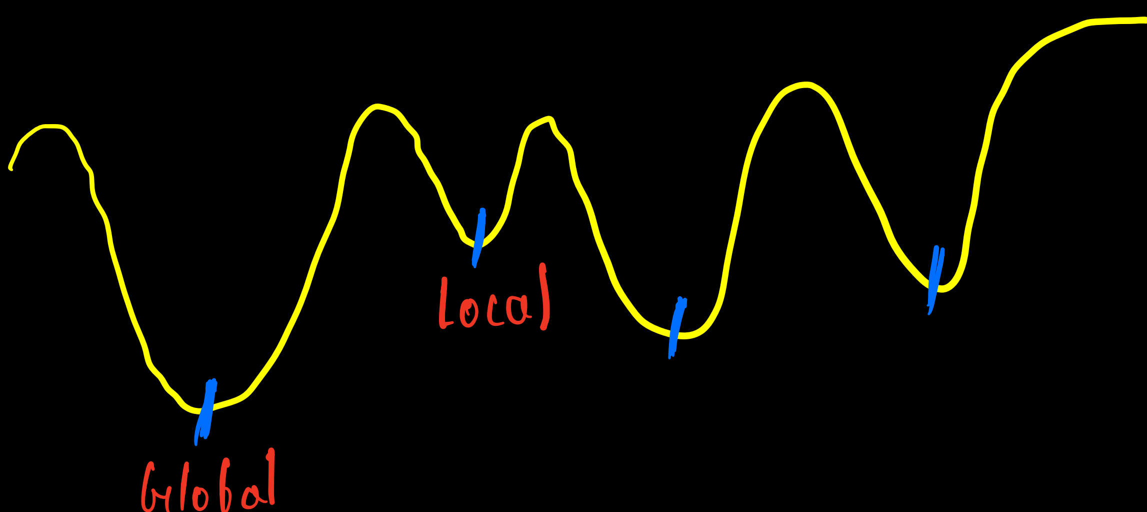


Global Minima

* Local Minima

{Gradient Descent} → optimization technique

w & b $\approx$ loss (0) close



w & b Update Rule

L = E
Loss = Errore

L·R → Learning Rate
(0·01, 0·001, 0·0001)

$$W_x = W_{x'} - \eta \left\{ \frac{\partial L}{\partial W_x} \right\}$$

W_x → New weight

$W_{x'}$ → old weight

{·0001}

$\frac{\partial L}{\partial W_{x'}}$ or $\frac{\partial E}{\partial W_{x'}}$ → Derivative of Errore wrt weight

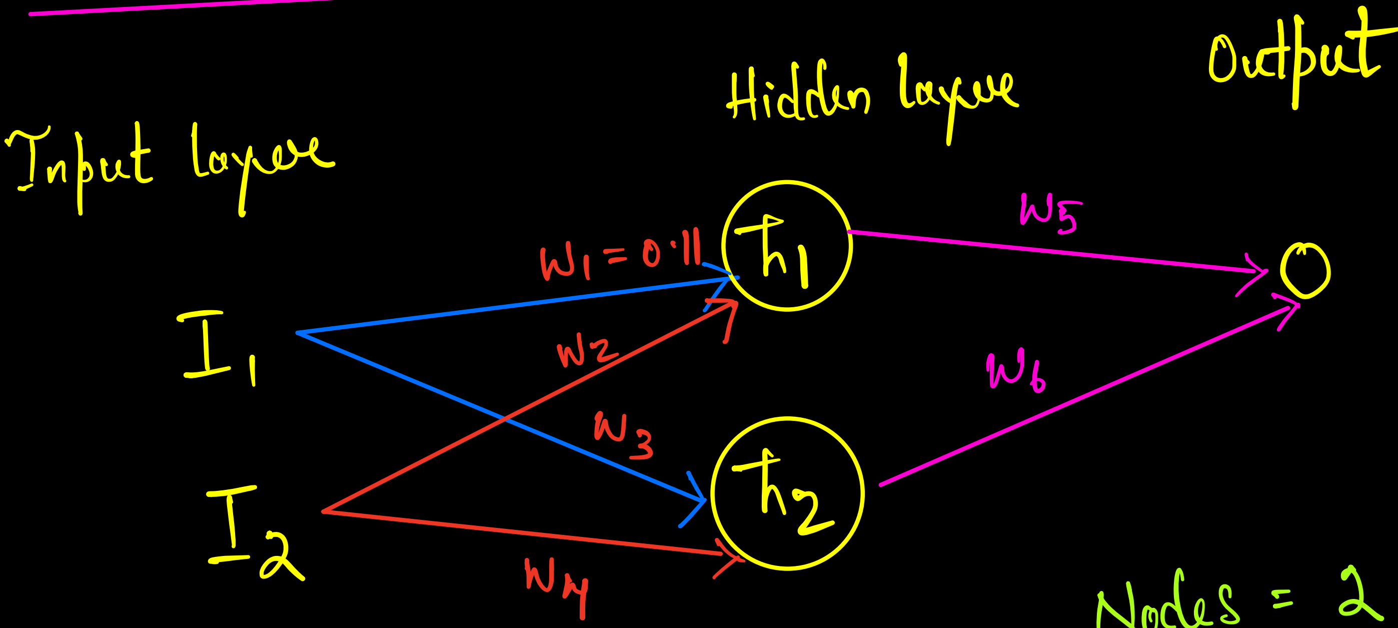
Bias

$$b_{new} = \underline{b_{old}} - \eta \frac{\partial L}{\partial b_{old}}$$

L·R → hyperparameter

Weight Update Rule

Forward Propagation



Nodes = 2

$w_1 = 0.11$

$w_2 = 0.21$

$w_3 = 0.12$

$w_4 = 0.08$

$w_5 = 0.14$

$w_6 = 0.15$

No of nodes
⇓
hyperparameter

No of layers
⇓
hyperparameter

$X = [i_1, i_2] = [2, 3]$

$y = 1$

Forward Propagation

(i) Input ⟶ Hidden layer

$$x \cdot W$$

$$[2, 3] \begin{bmatrix} .11 & .12 \\ .21 & .08 \end{bmatrix}$$

$$h_1 = 2 \times .11 + 3 \times .21 = 0.85$$

$$h_2 = 2 \times .12 + 3 \times .08 = 0.48$$

$$= \begin{array}{c} \begin{array}{cc} h_1 & h_2 \end{array} \\ \begin{bmatrix} 0.85 & 0.48 \end{bmatrix} \end{array}$$

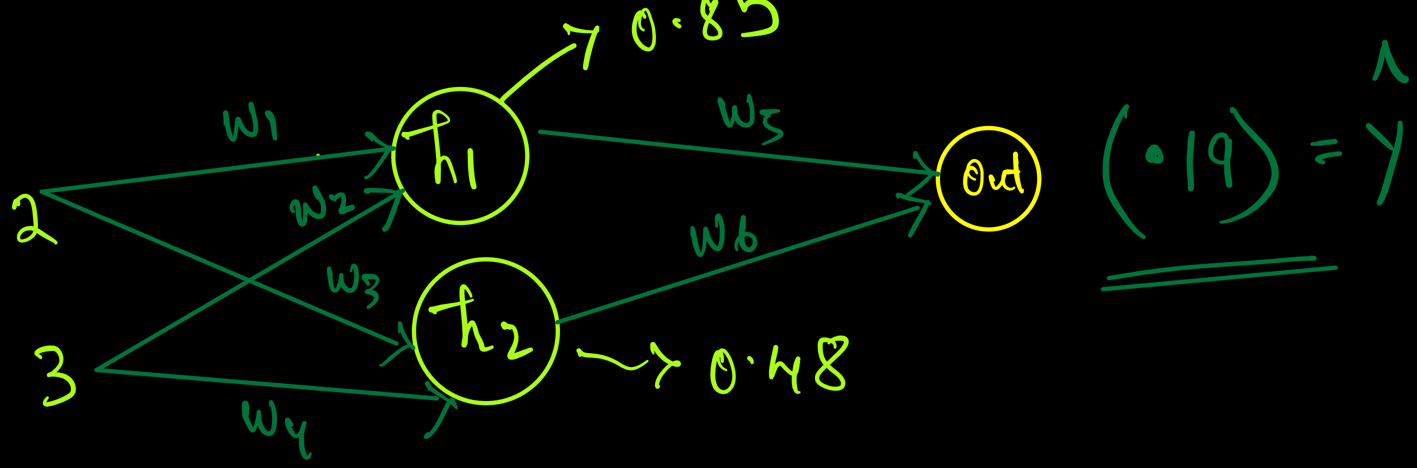
(ii) <u>Hidden layer</u> $\rightarrow$ Output layer

$$= \begin{bmatrix} 0.85 & 0.48 \end{bmatrix} \begin{bmatrix} 0.14 \\ 0.15 \end{bmatrix}$$

$$= 0.85 \times 0.14 + 0.48 \times 0.15$$

$$= [.191]$$

<u>Calculate Error</u>



Error Calculation

$$\text{Loss Function} = \frac{1}{2}(\hat{y} - y)^2$$

$$= \frac{1}{2}(.19 - 1)^2$$

$$\text{Error} = 0.327$$

Reducing the Error

$$\text{Prediction} = \text{output}$$

$$= (h_1) w_5 + (h_2) w_6$$

Phase, Output layer $\longrightarrow$ Hidden layer

Reverse

$h_1 = i_1 w_1 + i_2 w_2$

$h_2 = i_1 w_3 + i_2 w_4$

$$\text{Prediction} = (i_1 w_1 + i_2 w_2) w_5 + (i_1 w_3 + i_2 w_4) w_6$$

To change prediction value, weights.

How to change $\longrightarrow$ Backpropagation

1) Reduce the error
2) Change the weight

Weight Update $\longrightarrow$ Gradient Descent

{ Iterative optimization algorithm for finding the minimum of a function. }

Backprop with Gradient Descent

Weight Update :

$$w_{new} = w_{old} - \eta\left(\frac{\partial E}{\partial w_{old}}\right)$$

As per our Network

$$* \; w_6 = w_6 - \eta\left(\frac{\partial E}{\partial w_6}\right)$$

$$* \; w_5 = w_5 - \eta\left(\frac{\partial E}{\partial w_5}\right)$$

$w_6 =$

$\eta = .001$

* Chain Rule *

$$\left\{ \frac{\partial \text{Error}}{\partial w} = \frac{\partial \text{Error}}{\partial \text{prediction}} * \frac{\partial \text{prediction}}{\partial w_6} \right\}$$

$$\text{Error} = (\hat{y} - y)^2$$

{prediction}

Prediction → weight

Loss vs Cost Function

1) MSE (Regression)

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

→ actual

→ predicted

no of samples.

2) Log loss (Binary Cross Entropy)

Logistic Regression

$$\text{Loss} = -\frac{1}{n}\sum_{i=1}^{n}\left[y\log(\hat{y}) + (1-y_i)\log(1-\hat{y})\right]$$

Loss Function

1) Sample

Cost Function

1) Dataset

Average loss on the entire dataset

Gradient Descent

Optimization Algorithm

Update

$$\theta = \theta - \alpha \frac{\partial J(\theta)}{\partial \theta}$$

$\theta(w, b)$

$J(\theta)$: cost function

α - L.R

$\alpha = 0.001$

$w_0 = 0.6$

$$w_{new} = w_{old} - 0.001 \frac{\partial E}{\partial w_0}$$

$$= 0.6 - (0.001 \times 5)$$

$$= 0.6 - (\quad)$$

$$w_{new} = ?$$

Gradient Descent Types

* 1) Batch Gradient Descent : Full Dataset for every update

* 2) SGD (Stochastic Gradient Descent)

* 3) Mini Batch Gradient Descent

SGD → One data point at a time
(SLOW) Very slow

* Mini-Batch :- A small batch of dataset

Batch size :- (2^n) 8, 16, 32, 64, 128, 256, 512, 1024
↓
hardware

Framework :- Keras / Pytorch

Mini Batch Gradient Descent.

(SGD)

Data Terminologies

1) Batch :- Subset of your data

2) Iteration :- 1 update weight = 1 batch

Batch :- 1000 training sample

Batch size :- 100

Total No of batches :- 10

Iterations :- 10 iterations

3) Epoch :- full cycle of the entire dataset

All students taught in a class → 1 epoch

1 epoch = Network has seen every

training sample once.

Total Samples :- 1000

Batch size :- 100

Total Batch :- 10

Total Iteration :- 10

1000

Epochs = 1 , 10 Iteration / 10 Batches

1000

Epoch = 2 , 20 Iteration / 10 Batch

Epoch = 10 , 100 iterations

4) Step

1 step = weight update

DL Frameworks :- Tensorflow & Pytorch

1 weight update :- At every batch, the weight update takes place.