

Day 1 : Deep Learning

7pm - 9pm IST

Motive : { Clear their Basics, Maths, Interview Preparation }

Agenda

- (1) Deep Learning → Perceptron { AI VS ML VS DL VS DS }
- (2) Forward Propagation
- (3) Backward Propagation
- (4) Loss function
- (5) Activation functions
- (6) Optimizers.

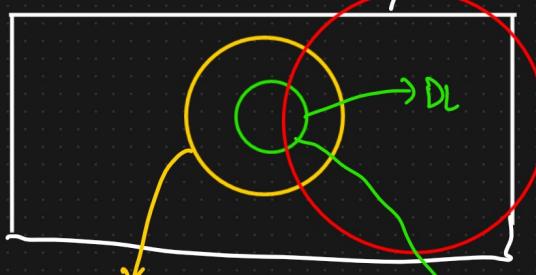
Prerequisite

- (1) Python
- (2) ML
- (3) Stats

→ Deep learning

AI VS ML VS DL VS DS

{ ML is a subset
of AI }



Goal

{ application which can do its own task
Without any human Intervention }

(1) Netflix App

(2) Self Driving Cars

(3) Amazon Application

(4) Sofia

RTX
3090

ML → Statstool to analyze
the data, visualize the data,

Predictions, forecasting, clustering

Researchers (1958)

Multi Layered Neural Network
{ Mimic the human brain }

{ Perceptron }

- * Why Deep Learning Is Becoming Popular?

① 2005 → ORKUT, FACEBOOK, Instagram, WhatsApp, LinkedIn Twitter

DATA → Exponentially ↑↑↑

2008 → {Big DATA} → Efficiently

Years {2013-2014}

Netflix

2013 → Company had huge amount of Data

↓ {AI → popular}

Scans → Products

Panasonic : AC's, TV's, Refrigerator {Data}



Model → Reduce the Electricity Bills ↓

↓
Subscription Basis

{RTX Titan}



{Current Revenue, Better Decisions} {RTX 3090} ↑

② Hardware Advancement (Nvidia) → GPU's → TRAINING THE MODEL.

GPU's → Cost ↓↓↓

④ Perceptron {Single Layered Neural N/W}.

I/P layers

Study
 x_1

play
 x_2

Sleep
 x_3

HL 1

O/p Layer

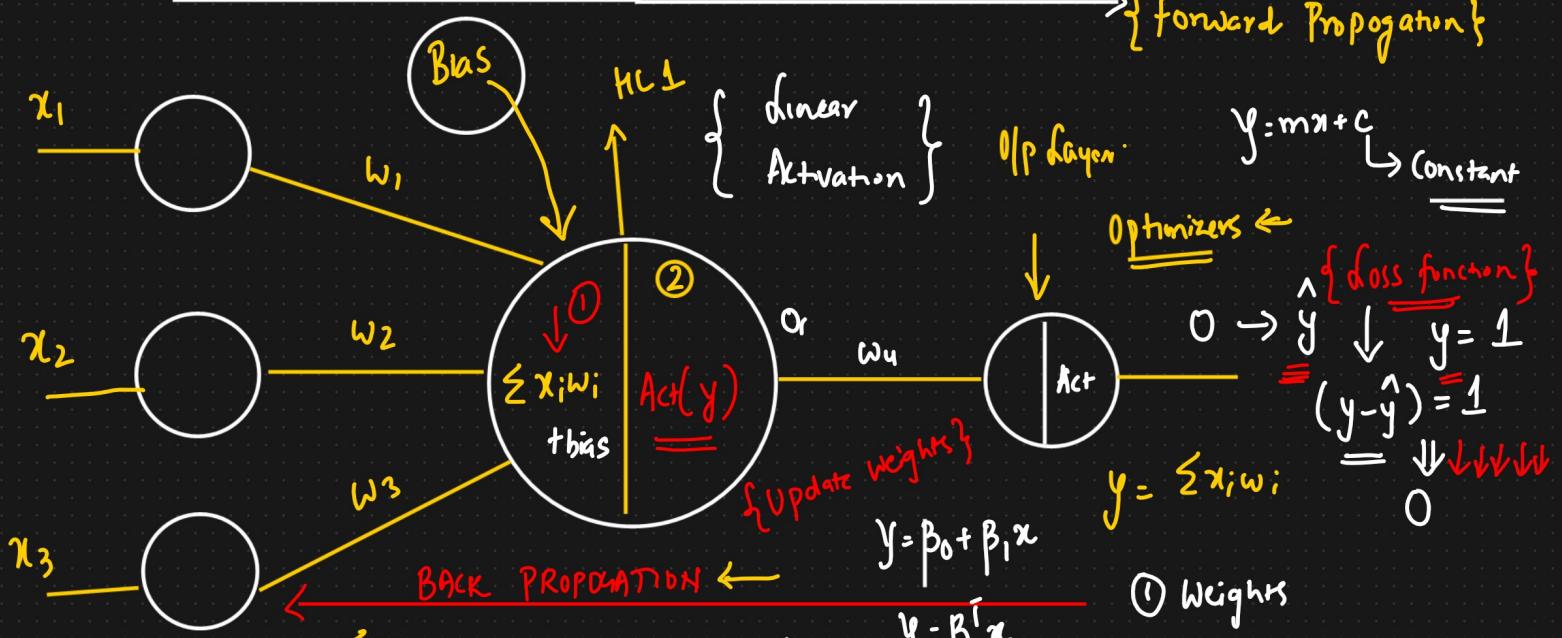
Output
Neuron

Binary Classification

Data

Study	Play	Sleep	Pass/Fail
7	3	7	1 ←
2	5	8	0
4	3	7	1

Forward Propagation



$$y = mx + c$$

Constant

Optimizers ←

d loss function

$0 \rightarrow \hat{y} \downarrow \hat{y} = 1$

$(\hat{y} - y) = 1 \downarrow \downarrow \downarrow \downarrow \downarrow \downarrow$

$y = \sum x_i w_i \quad 0$

- ① Weights
- ② Neurons
- ③ Activations

$$\text{Binary Classification} = \underline{\underline{w^T x}}$$

$$\text{Sigmoid} = \frac{1}{1 + e^{-y}} = \frac{1}{1 + e^{-(\sum x_i w_i + b)}}$$

0 to 1

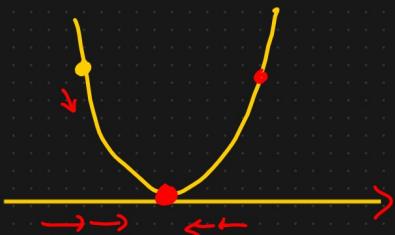
$$\begin{cases} \geq 0.5 \rightarrow 1 \\ < 0.5 \rightarrow 0 \end{cases}$$

$$\boxed{0 \text{ or } 1}$$

- ① ANN
- ② CNN
- ③ RNN
- ④ Object Detection

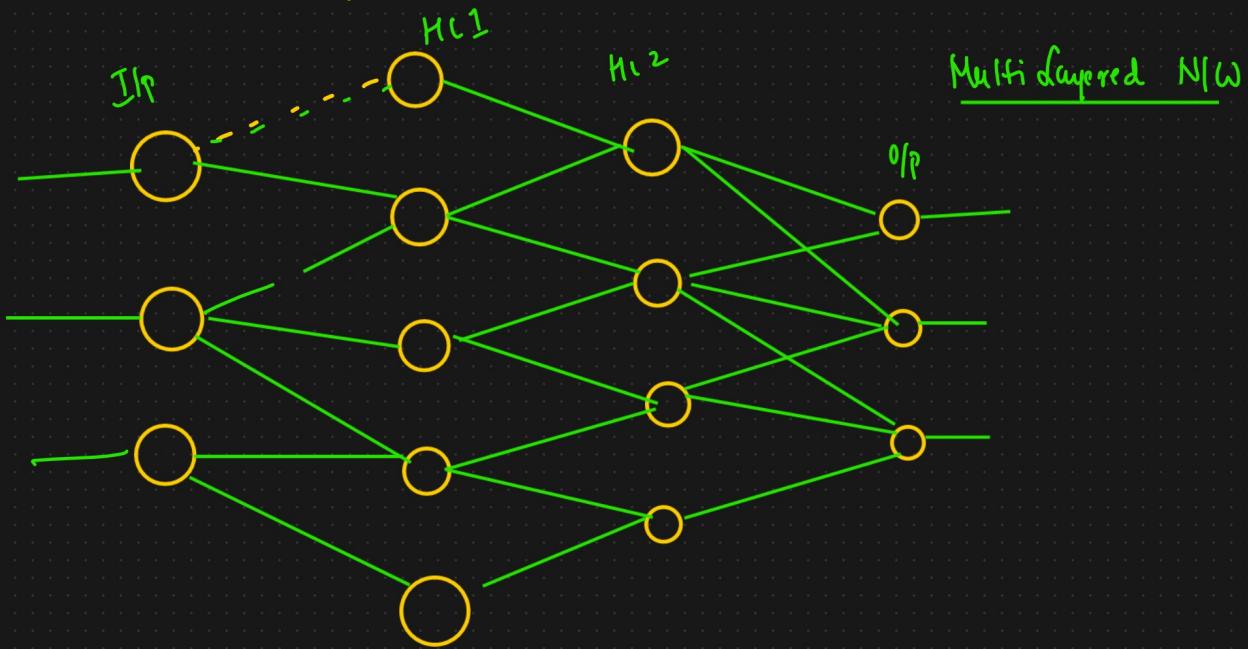
GRADIENT DESCENT ⇒ Kind of Optimizers

$$\hat{y} - y \Rightarrow \text{minimal}$$



Conclusion

- ① I/P layers
 ② Weights
 ③ Bias Day 2
 ④ Activation function \times
 {
 ⑤ Loss function $\times \{ \hat{y} - y \}$ } $\downarrow \text{Wb}$
 ⑥ Optimizers
 ⑦ Update the weight } Day 3
- } Forward Propagation
- } Backward Propagation
- (300Bs)
Scribble Ink
ANN
Chain Rule of Differentiation



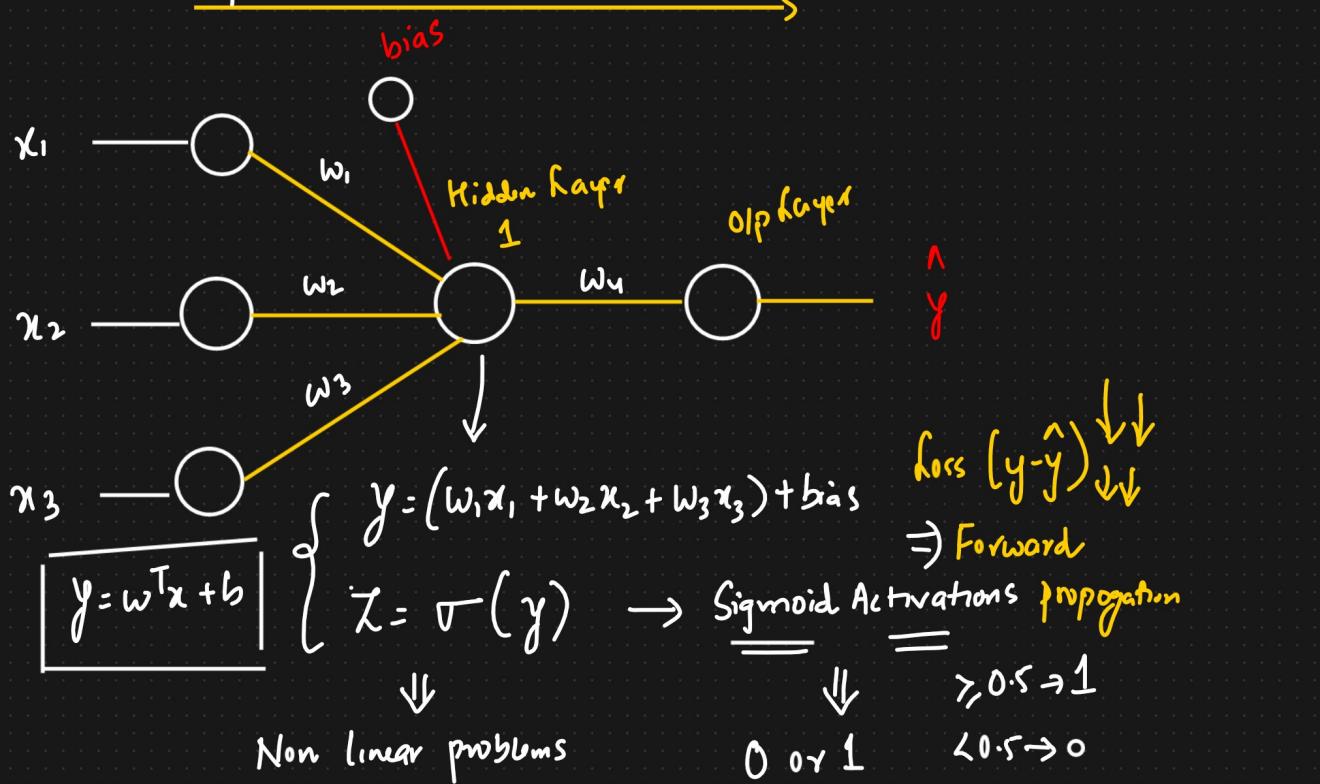
- ① ANN
 ② CNN
 ③ RNN
- } NLP 7 days

Day 2 - Deep Learning

Agenda

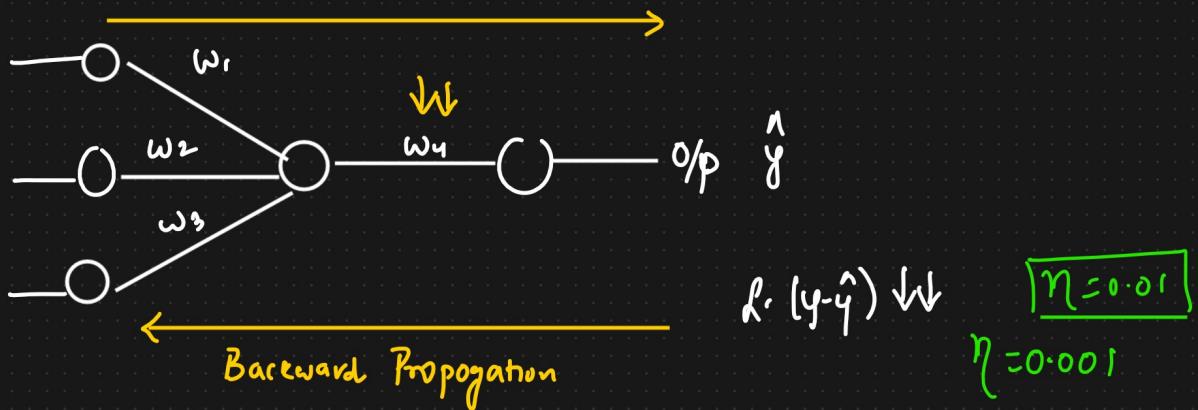
- ① Forward Propagation ✓
- ② Chain Rule of Differentiation ✓
- ③ Vanishing Gradient Problem ✓
- ④ Loss functions ✓

Activation functions



② Backpropagation

- ① Weight update formula
- ② Chain Rule of Differentiation ✓



① Weight updation formula

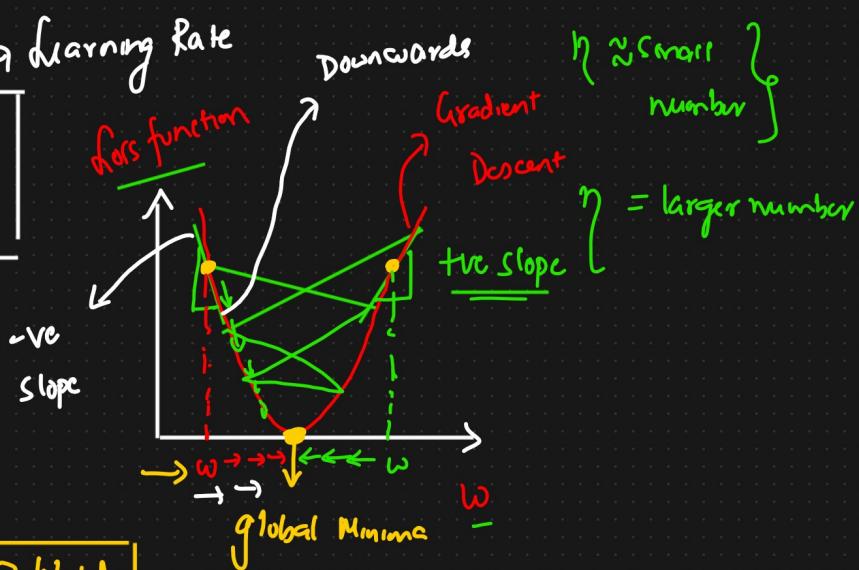
$$W_{\text{new}} = W_{\text{old}} - \eta \left[\frac{\partial h}{\partial w_{\text{old}}} \right]$$

\nearrow Slope

$$\frac{\partial h}{\partial w_{\text{old}}} = \boxed{-\text{ve slope}}$$

$$W_{\text{new}} = W_{\text{old}} - \eta (-\text{ve}) \quad \downarrow$$

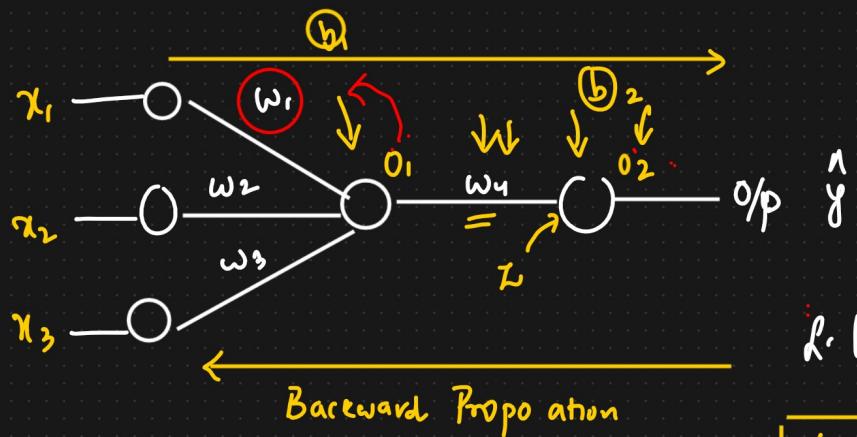
$$= W_{\text{old}} + \eta (\text{ve}) \quad \boxed{W_{\text{new}} > W_{\text{old}}}$$



$$W_{\text{new}} = W_{\text{old}} - \eta (+\text{ve})$$

$$\boxed{W_{\text{new}} << W_{\text{old}}}$$

② Chain Rule of Differentiation



$$W_{\text{new}} = W_{\text{old}} - \eta \left[\frac{\partial L}{\partial w_{\text{old}}} \right]$$

$$L = \sigma(o_1 w_4 + b)$$

$$R. (y - \hat{y}) \downarrow$$

$$w_4_{\text{new}} = w_4_{\text{old}} - \eta \left[\frac{\partial L}{\partial w_4_{\text{old}}} \right]$$

$$b_2_{\text{new}} = b_2_{\text{old}} - \eta \left[\frac{\partial L}{\partial b_2_{\text{old}}} \right]$$

{ Chain Rule of
Derivative }

$$\frac{\partial h}{\partial w_{\text{old}}} = \frac{\partial L}{\partial o_2} * \frac{\partial o_2}{\partial w_{\text{old}}}$$

$$\omega_{1\text{ new}} = \omega_{1\text{ old}} - \eta$$

$$\frac{\partial h}{\partial \omega_{1\text{ old}}}$$

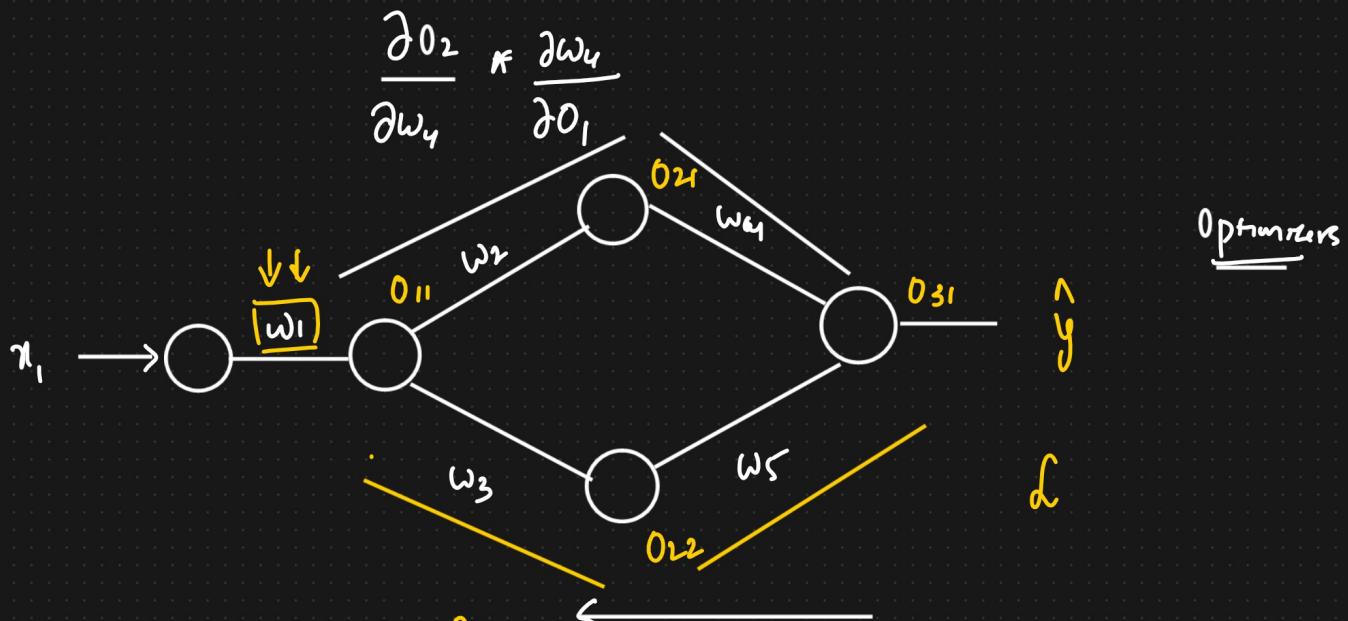
$$\omega_{2\text{ new}} = \omega_{2\text{ old}} - \eta$$

$$\frac{\partial h}{\partial \omega_{2\text{ old}}}$$

\rightarrow loss

$$\frac{\partial h}{\partial \omega_{1\text{ old}}} = \frac{\partial h}{\partial o_2} * \frac{\partial o_2}{\partial o_1} * \frac{\partial o_1}{\partial \omega_{1\text{ old}}}$$

↓

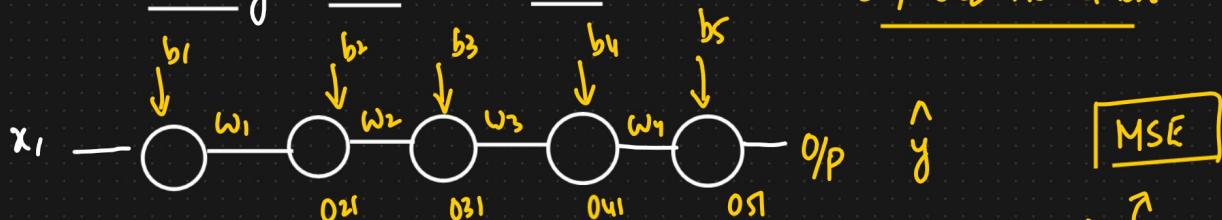


$$\omega_{1\text{ new}} = \omega_{1\text{ old}} - \eta \frac{\partial h}{\partial \omega_{1\text{ old}}}$$

$$\frac{\partial h}{\partial \omega_{1\text{ old}}} = \left[\frac{\partial h}{\partial o_{31}} * \frac{\partial o_{31}}{\partial o_{21}} * \frac{\partial o_{21}}{\partial o_{11}} * \frac{\partial o_{11}}{\partial \omega_{1\text{ old}}} \right]$$

$$+ \left[\frac{\partial h}{\partial o_{31}} * \frac{\partial o_{31}}{\partial o_{22}} * \frac{\partial o_{22}}{\partial o_{11}} * \frac{\partial o_{11}}{\partial \omega_{1\text{ old}}} \right]$$

③ Vanishing Gradient Problem



Sigmoid Activation

MSE

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

$$O_{51} = \sigma \left[(O_{41} * w_4) + b \right]$$

↓
Sigmoid Activation

$$w_{i,\text{new}} = w_{i,\text{old}} - \eta \left[\frac{\partial L}{\partial w_{i,\text{new}}} \right]$$

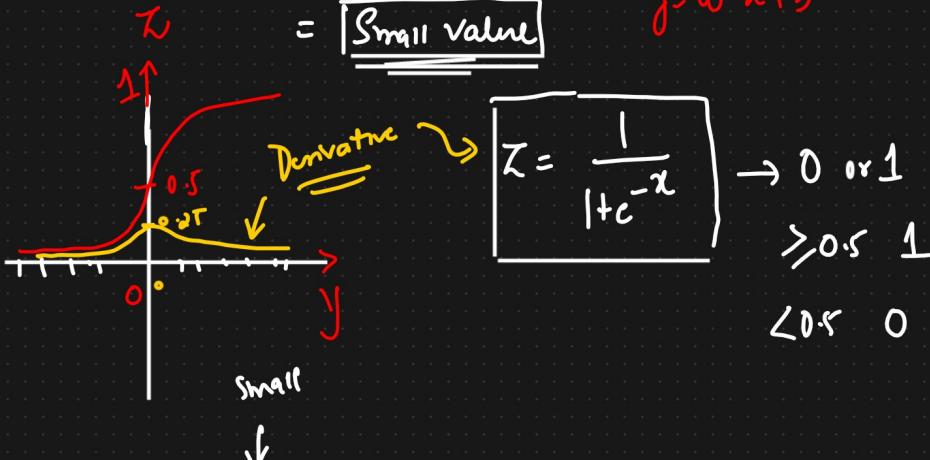
$$\frac{\partial L}{\partial w_{i,\text{new}}} = \frac{\partial L}{\partial O_{51}} * \frac{\partial O_{51}}{\partial O_{41}} * \frac{\partial O_{41}}{\partial O_{31}} * \frac{\partial O_{31}}{\partial O_{21}} * \frac{\partial O_{21}}{\partial w_i}$$

$$= 0.25 * 0.15 * 0.10 * 0.05 * 0.02$$

$$z = \boxed{y = w^T x + b}$$

Derivative:

$$0 \leq f(y) \leq 0.25$$



$$w_{i,\text{new}} = w_{i,\text{old}} - \eta \quad (\text{small number})$$

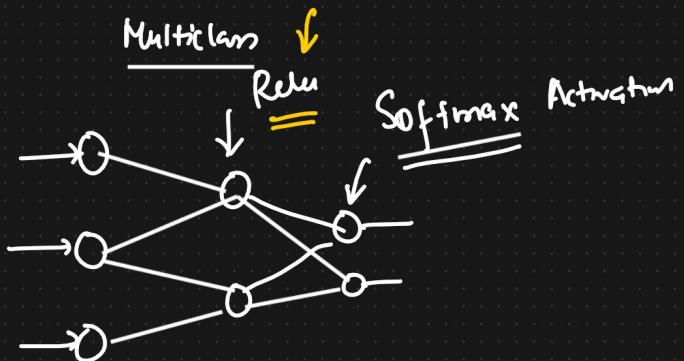
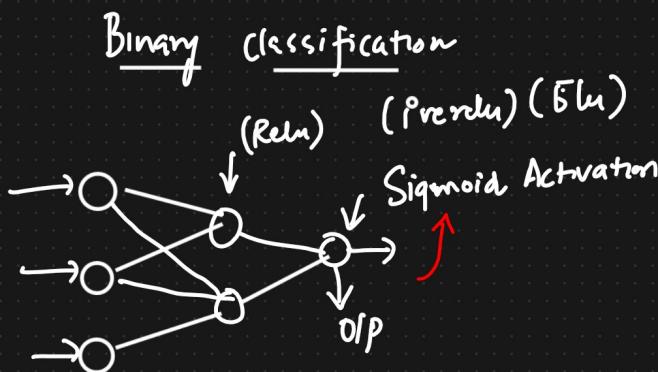
Another Activation
function.

$$\boxed{w_{i,\text{new}} \approx w_{i,\text{old}}} \Rightarrow \text{Vanishing gradient problem}$$

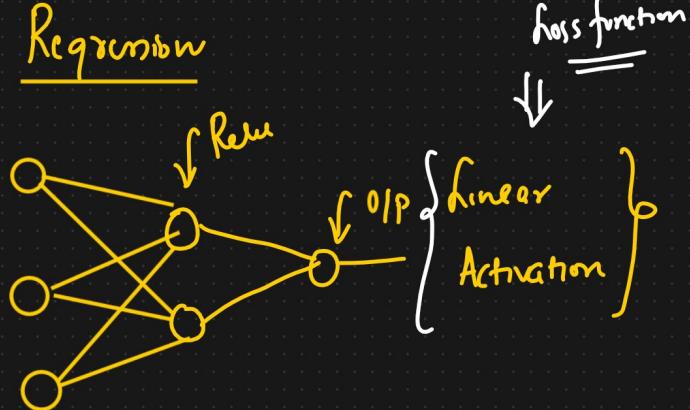
↓
No change in weights

- ① Sigmoid
- ② Tanh
- ③ ReLU
- ④ Leaky ReLU
- ⑤ PReLU

Technique Which Activation fn we should Use



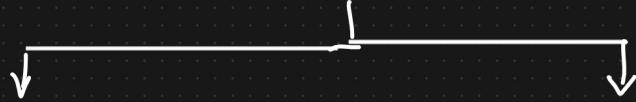
Regression



(f)

loss functions

Deep Learning (ANN)



loss functions

Regression

Exp	Degree	Salary	O/P
10	Phd	—	
—	—	—	
—	—	—	

Regression Problem

Classification

Play	Study	Pas/Fail
10	2	Fail
4	3	Fail
5	5	Maybe
2	7	Pas

① Regression

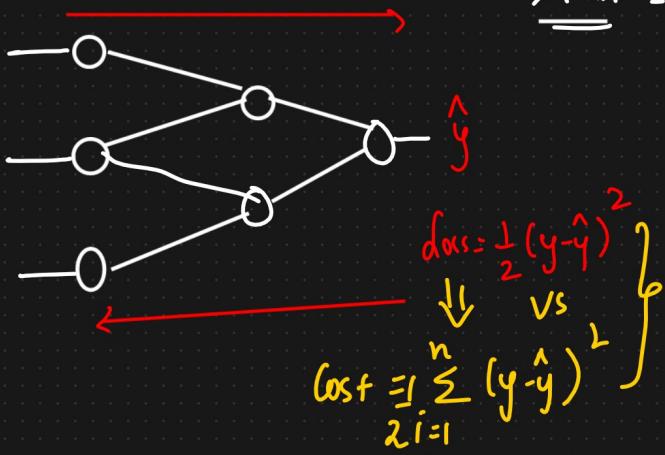
- ① MSE {Mean Squared Error}
- ② MAE {Mean Absolute Error}
- ③ Huber Loss

Loss function AND

(Cost Function)

10 records
↓
=

Dataset = 100 records



① Mean Squared Error (MSE)

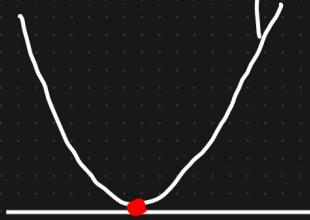
Loss function = $\frac{1}{2} (y - \hat{y})^2$ Cost function = $\frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2$

↓ Error

Quadratic Equation

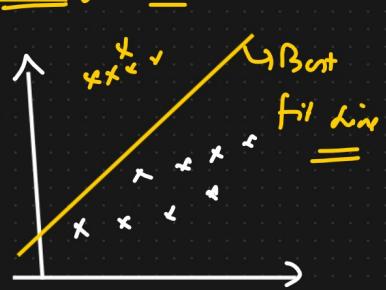
$$(a-b)^2 = a^2 - 2ab + b^2$$

$$a x^2 + b x + c$$



Gradient Descent

Penalizing Error

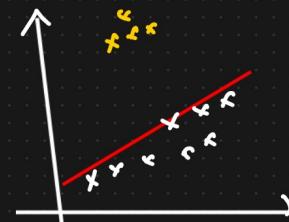


Advantages

- ① Differentiable
- ② It has only 1 local or Global Minima.
- ③ It converges faster

① Not Robust to outliers

Disadvantage



② Mean Absolute Error

$$\text{loss fn} = \frac{1}{2} |y - \hat{y}|$$

$$\text{Cost fn} = \frac{1}{2} \sum_{i=1}^n |y_i - \hat{y}_i|$$

① Robust to outliers

\Rightarrow Time consuming ↑↑

\Rightarrow Subgradient



③ Huber loss

① MSE

② MAE

outliers are not
present

Hyperparameter

$$\text{loss} = \begin{cases} \frac{1}{2} (\hat{y} - y)^2 & \text{if } |\hat{y} - y| \leq \delta \\ \delta |\hat{y} - y| - \frac{1}{2} \delta^2, & \text{otherwise} \end{cases}$$

③ Classification

Cross Entropy → Binary Cross Entropy → Binary Classification

Cross Entropy → Categorical Cross Entropy → Multiclass Classification.

① Binary Cross Entropy

$$\text{loss} = -y * \log(\hat{y}) - (1-y) * \log(1-\hat{y}) \Rightarrow \text{Logistic Regression}$$

↓

$$\text{loss} = \begin{cases} -\log(1-\hat{y}) & \text{if } y=0 \\ -\log(\hat{y}) & \text{if } y=1 \end{cases} \quad \text{Binary Classification}$$

$$\hat{y} = \frac{1}{1 + e^{-x}}$$

② Categorical Cross Entropy {Multi-class Classification Problem}

f_1	f_2	f_3	O/P	Good	Bad	Neutral	$i = \text{Row}$
2	3	4	Good	[1]	0	0	$j = \text{Column}$
5	6	7	Bad	0	1	0	
8	9	10	Neutral	0	0	1	

$$L(x_i, y_i) = - \sum_{j=1}^C y_{ij} * \ln(\hat{y}_{ij})$$

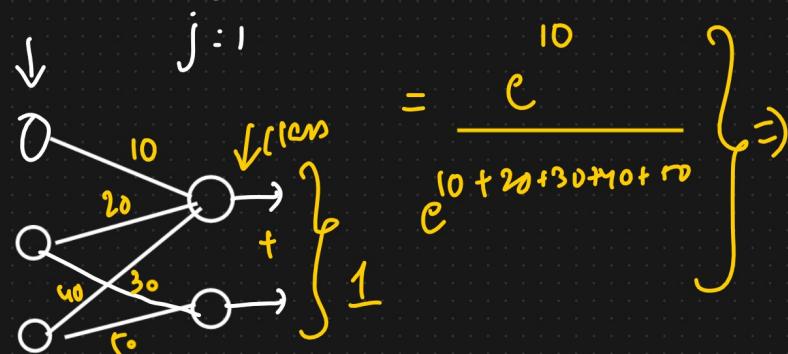
$$y_i = [y_{i1}, y_{i2}, y_{i3}, \dots, y_{ic}]$$

$$y_{ij} = \begin{cases} 1 & \text{if the element is in class.} \\ 0 & \text{Otherwise} \end{cases}$$

\hat{y}_{ij} = Softmax Activation $\xrightarrow{\text{O/P Layer}}$

$$f(z) = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad \begin{cases} \Rightarrow \text{Softmax} \\ \text{Activation} \end{cases}$$

0.4, 0.5, 0.6



Conclusions

ReLU, Softmax \Rightarrow MultiClass } \rightarrow Categorical Cross Entropy

ReLU, Sigmoid \Rightarrow Binary } \rightarrow Binary Cross Entropy.

Linear Regression

ReLU, Linear Activation \rightarrow MSE, MAE, Huber Loss

Day 3 - Deep learning.

Agenda

① Optimizers

- i) Gradient Descent
- 2) SGD (Stochastic Gradient Descent)
- ③ Mini Batch SGD
- ④ SGD with Momentum
- ⑤ Adagrad
- ⑥ RMSprop
- ⑦ Adam Optimizer

Batch, Epochs, Iterations

\Downarrow
ANN
 \equiv .

GRADIENT DESCENT \rightarrow Optimizer

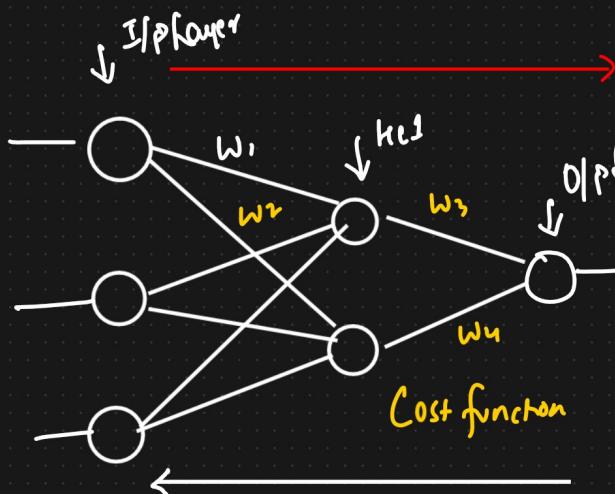
Weight Updation Formula

$$W_{\text{new}} = W_{\text{old}} - \eta \frac{\partial h}{\partial W_{\text{old}}}$$

loss / cost



Global Minimum.



$$\text{MSE} \quad \text{Optimizers}$$

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

Epoch $\xrightarrow{\hspace{1cm}} \{ \}$ 1 Epoch $\xleftarrow{\hspace{1cm}} \{ \}$

$\{ 1000000 \}$

Disadvantage

① Resource Extensive {huge RAM}

100000

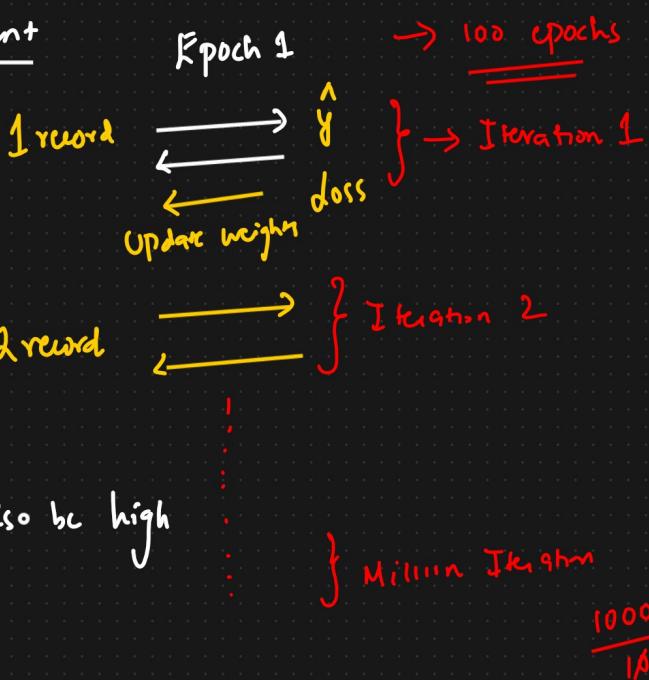
② Stochastic Gradient Descent

- ## ① RAM ↓↓

Disadvantage

- (f) Convergence will }
be very slow }

- ⑥ Time complexity will also be high



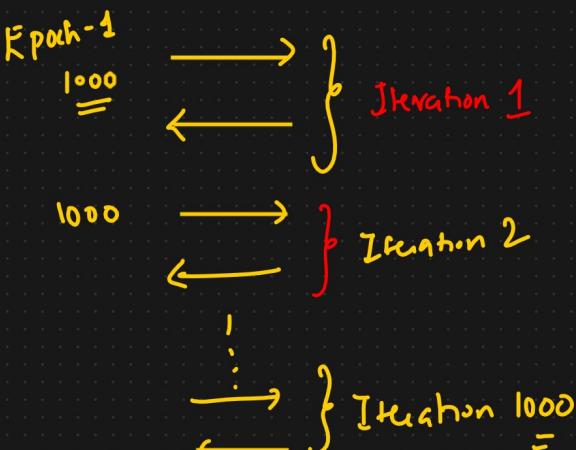
③ Mini batch SGD

1000000 batch-size = 1000

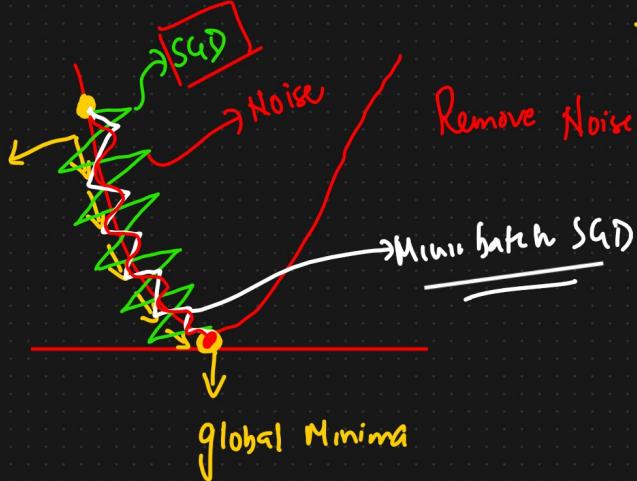
- ## A Resource Intensive

- ② Convergence will be better

- ③ Time complexity will improve



四



④ SGD With Momentum

{ Exponential Weighted Average }

$$w_{\text{new}} = w_{\text{old}} - \eta \left[\frac{\partial L}{\partial w_{\text{old}}} \right]$$

$$b_{\text{new}} = b_{\text{old}} - \eta \left[\frac{\partial L}{\partial b_{\text{old}}} \right]$$

↓
Time Series
↓

ARIMA, ARMA,

$$\boxed{w_t = w_{t-1} - \eta \left[\frac{\partial L}{\partial w_t} \right]}$$

Exponential Weighted Average

{ Forecasting }

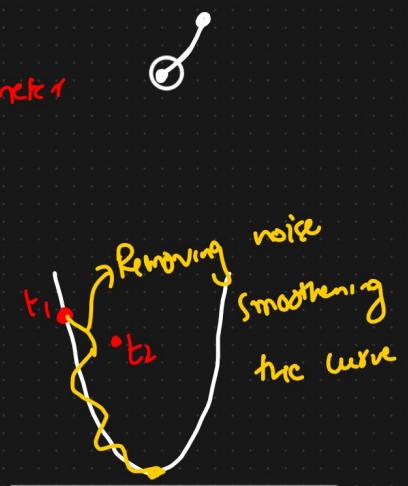
$$t_1 \quad t_2 \quad t_3 \quad t_4 \quad \dots \quad t_n$$

$$a_1 \quad a_2 \quad a_3 \quad a_4 \quad \dots \quad a_n$$

$\beta \Rightarrow$ Hyper parameter

$$\beta = 0 \text{ to } 1$$

$$0.95$$



$$V_{t_1} = a_1$$

$$V_{t_2} = \beta * V_{t_1} + (1-\beta) * a_2$$

$$= (0.95) * V_{t_1} + (0.05) * a_2$$

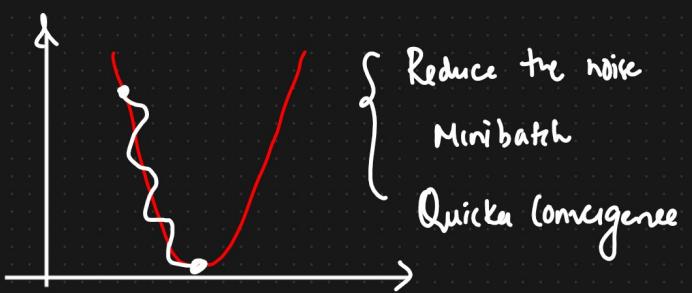
$$V_{t_3} = \beta * V_{t_2} + (1-\beta) * a_3$$

Exponential Weighted Avg

↑

$$w_t = w_{t-1} - \eta V_{dw}$$

$$\boxed{V_{dw} = \beta * V_{dw_{t-1}} + (1-\beta) * \frac{\partial L}{\partial w_{t-1}}}$$

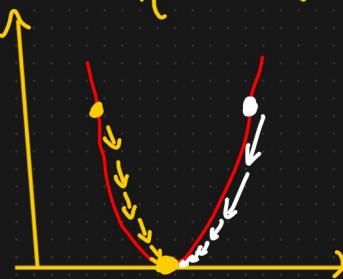


Recap

- ① Gradient Descent
- ② SGD
- ③ Mini batch SGD
- ④ SGD with Momentum
- ⑤ Adagrad → Adaptive Gradient Descent

{ fixed } \Rightarrow optimization

η = Learning Rate



Global Minima

Minima



$\eta \downarrow$

Decreasing ↓↓↓

$$\eta = \eta - 0.01$$

$$\Rightarrow \sqrt{d_t + \epsilon}$$

$$d_t = \sum_{i=1}^t \left(\frac{\partial L}{\partial w_i} \right)^2$$

Huge
number

$t=1 \quad t=2 \quad t=3$

$$\eta = 0.01 \quad \eta = 0.005 \quad \eta = 0.002$$

- ⑥ Adadelta and Rmsprop

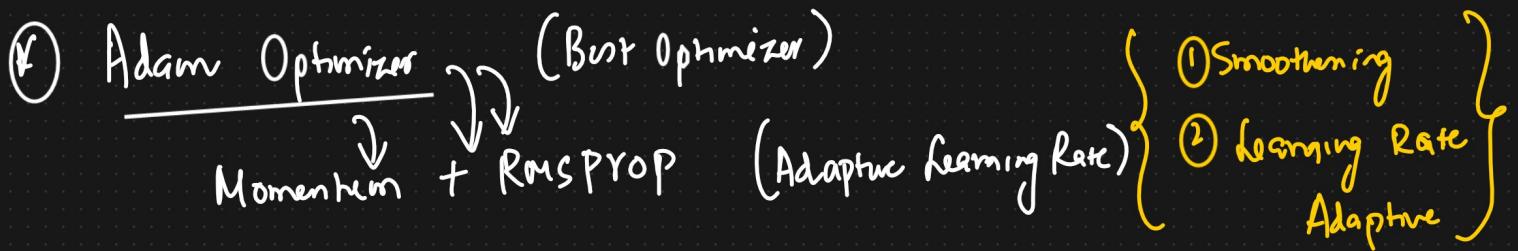
Exponential Weighted Average

$$\eta^t = \frac{\eta}{\sqrt{S_{dw} + \epsilon}}$$

$$S_{dw}^t = \beta S_{dw}^t + (1-\beta) \left(\frac{\partial L}{\partial w_{t-1}} \right)^2$$

$$\beta = 0.95$$

$$S_{dw_t} = (0.95) S_{dw_{t-1}} + (0.05) \left(\frac{\partial L}{\partial w_{t-1}} \right)^2$$



$$V_{dw,0} V_{db,0} S_{dw} = 0 \quad S_{db} = 0$$

↳

$$\boxed{w_t = w_{t-1} - \eta^t V_{dw}}$$

$$b_t = b_{t-1} - \eta^t V_{db}$$

$$\eta^t = \frac{\eta}{\sqrt{S_{dw} + \epsilon}}$$

$$\boxed{V_{dw,t} = \beta \times V_{dw,t-1} + (1-\beta) \frac{\partial L}{\partial w_{t-1}}}$$

$$\boxed{V_{db,t} = \beta \times V_{db,t-1} + (1-\beta) \frac{\partial L}{\partial b_{t-1}}}$$