·NOTES·

Subject: Machine Learn

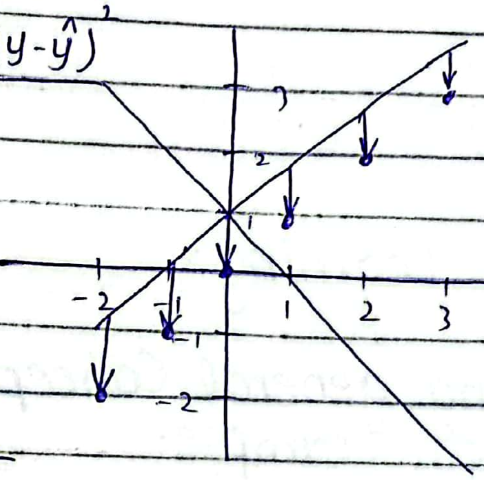
Sr. No.	Date	Topic

ML:-

To find the function(f)
Here we estimate m & c as they are unknown.

$m = -1, c = 1$

x	y	$\hat{y} = -x+1$	$E = (y-\hat{y})^2$
-2	-2	3	25
-1	-1	2	9
0	0	1	1
1	1	0	1
2	2	-1	9
3	3	-2	25
			70



$\frac{70}{6} = 12$ Avg

let $m = 1, c = 1$

$\hat{y} = x+1$	$E = (y-\hat{y})^2$
-1	1
0	1
1	1
2	1
3	1
4	1
	6

$\frac{6}{6} = 1$ Avg

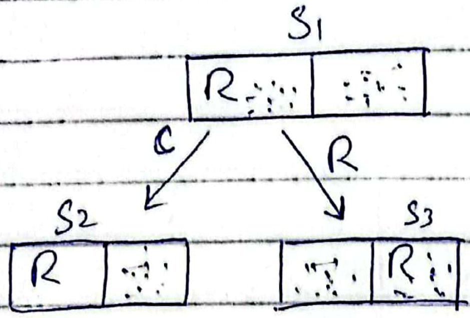
Reinforcement Learning:-

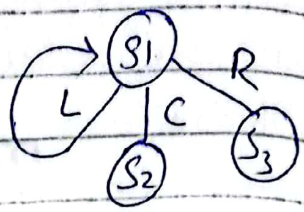
→ Agent
→ Environment.

hypothesis → Assumptions.
↘ h

All posibilities that can occur is called hypothesis space

Environment → Set of states.





Concept Learning:-

→ Extracting general Concept from training example.

*→ Learning function ~~must~~ would be Boolean.

	? Temp	? Humidity	X ? Weather	Wind	Play/Decision
D1	Medium	Low	Sunny	medium	YES
D2	" "	"	Overcast	medium	YES
D3	low	high	Rainy	medium	NO
D4	low	high	Sunny	~~high~~ medium	YES
	ϕ 2 (4)	× 2 (4)	× 3 (5)	× 2 (4) → bcz of ϕ, ?	

Hypothesis → h

h = <temp, Humidity, weather, wind>

h = < a_1, a_2 ---- , a_4 >.

24 hypothesis (2×2×3×2 = 24)

h_1 = <med, low, suny med>

h_2 = <low ---------->.

⋮

h_{24} → All these are collectively known as hypothesis space and denoted by H.

? → Kisi b value sa affect nhi prta.
↳ General concept (no importance)

Find S Algorithm (1st algo work on this phenomena)

Instance = $\{a_1, a_2, \ldots, a_4\} \rightarrow D_1, D_2, D_3 \ldots$
hypothesis = $\{a_1, a_2, \ldots a_4\}$.

Target function $c: X \rightarrow \{0,1\}$.

Initial h,

$h_0 = \langle \phi, \phi, \phi, \phi \rangle$
$h_1 = \langle$med, low, sunny, med$\rangle$ Yes D1
$\langle$ med, low, overcast, med $\rangle$ Yes D_2
$h_2 = \langle$med, low, ?, med$\rangle$
$\langle$ low, high, sunny, med$\rangle$ Yes D_4.
$h_3 = \langle$?, ?, ?, med $\rangle$

If wind = medium, you can play.

Assembly

CS → code segment
IP → Instruction Pointer.
SS → Stack Segment.
SP → Stack Pointer.
BP → Base Pointer.
SI → Source Index
DI → Destination Index.
D.S → Data Segment
ES → Extra Segment

Flags Register Size → 16 bits
↳ Uses for indicates event occurring.
↳ Indication 1 bit sa hoti hai.

Entropy → Jb value of boht zyada randomness ho.
↓
E(S) → Huffman coding

flags:-

CF → Carry flag] 17 bit^th store nhi ho skti Or register 16 bits ka hai.

Candidate Elimination Algorithm.
↘ CEA.

H: Hypothesis Space:-

Vs: Version space

↳ All set of hypothesis which are consistent with our data are known as Vs.

Sky	AirTemp	Humidity	Wind	Water	Forcast	PlaySport
		~~normal~~	Strong			
Sunny	warm	normal	"	Normal	Same	Yes
"	"	High	"	"	"	Yes
rainy	cold	"	"	"	change	no
Sunny	warm	"	"	cool	"	yes.

<sky, AirTemp, Humidity, wind, water, forcast>

→ wo hypothesis find krny hain jo Vs k part ho mtlb es data ko classify kr rhay hon.

→ we have to define boundries @ up & down through CFA.

→ Two types of boundries can be; most general & most specific boundry.

S → most specific
G → " General

(List & Elimination Method)

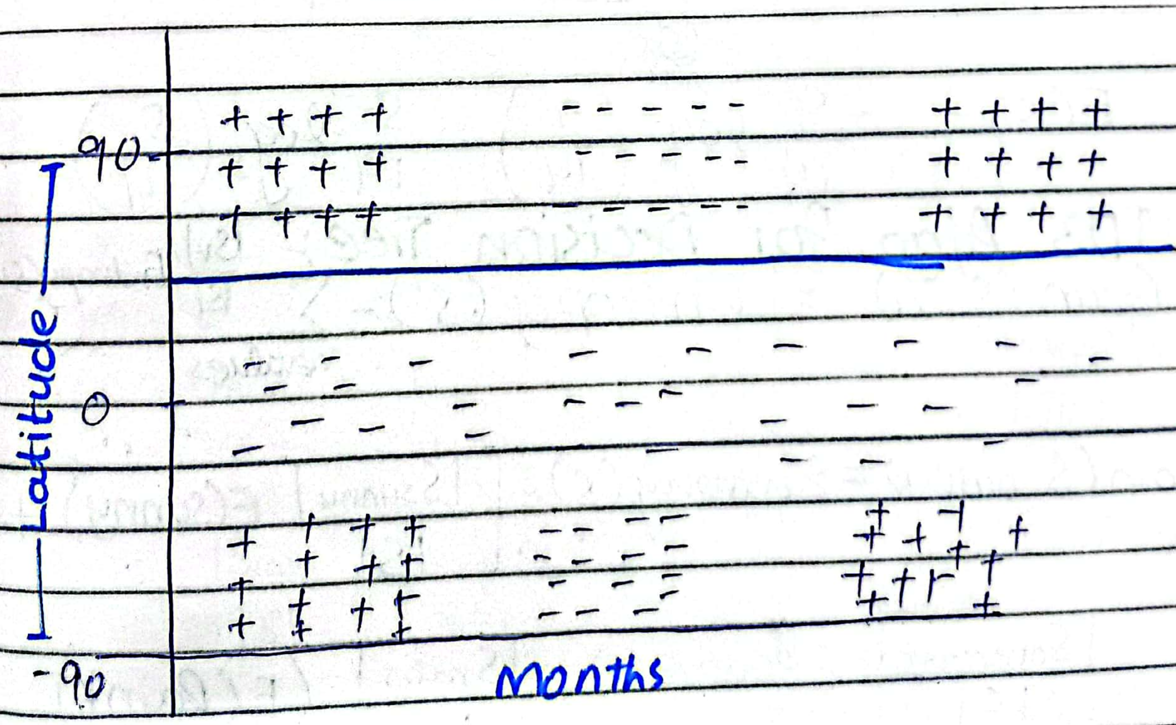
$S_0 = \langle \phi, \phi, \phi, \phi, \phi, \phi \rangle$

(D1) S_1 = <sunny, warm, normal, strong, normal, same>
S_2 = <sunny, warm, ?, strong, normal, same>
S_3 = <sunny, warm, ?, strong, ?, ?>

<sunny, warm, ?, ?, ?, ?> <sunny, ?, ?, strong, ?, ?>
<?, warm, ?, strong, ?, ?>
→ ~~<?, warm, ?,~~ <sunny, warm, ?, strong, ?, ?>

→ <sunny, ?, ?, ?, ?, ?> <?, warm, ?, ?, ?, ?> ~~<?, ?, normal, ?, ?, ?>~~ accept for up 2
~~<?, ?, ?, ?, cool, ?>~~ <?, ?, ?, ?, ?, same> Not accept for 4th
→ $G_2 = G_1 = G_0$ = <?, ?, ?, ?, ?, ?>

Decision Tree:-



$$E(S) = -P_{\oplus} \log_2 P_{\oplus} - P_{\ominus} \log_2 P_{\ominus}$$

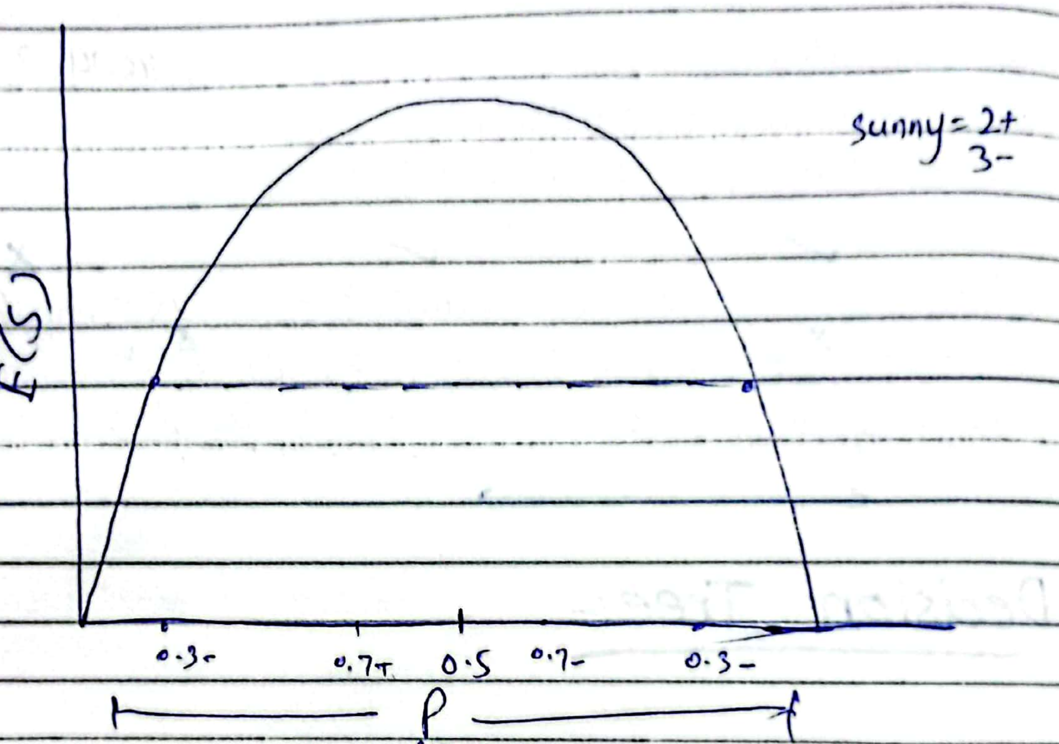
$S' \rightarrow$ data set

$A \rightarrow$ Attribute

$S_v \rightarrow$ the set that has the value v

→ If P of + is zero, it will return P = zero & vice versa.

→ If P of + is 50% & P of − is also 50%, it will return P = 1



$$E(S) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right)$$

ID3 Algo for Decision Tree:-

$$Gain(S', A) = Entropy(S') - \sum_{v \in values} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, outlook) = Entropy(S) - \left[\frac{|S_{sunny}|}{|S|} E(sunny) + \frac{|S_{overcast}|}{|S|} E(overcast) + \frac{|S_{rain}|}{|S|} (E(Rain))\right]$$

$$= 0.91 - \left[\frac{5}{14} \times \left|-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\left(\frac{2}{5}\right)\right|\right]$$

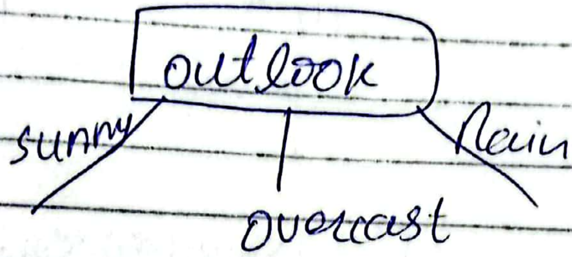
All yes for
→ that's why zero = p.

$$+ \frac{4}{14}\left[0\right] + \frac{5}{14} \times \left[-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right]$$

$$= 0.134$$

Gain (S, temp) = 0.032

Gain (S, Humidity) = 0.112



ID3 :-

$$\text{Entropy} = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}$$

$$\text{General Entropy} = -\sum_{i=1}^{n} P_i \log_2 P_i$$

$$= -P_{\oplus}\log_2 P_{\oplus} - P_{\ominus}\log_2 P_{\ominus}$$

$$IG(S', A) = E(S) - \sum_{v \in values} \frac{|S_v|}{S} E(S_v)$$

$$IG(S, Outlook) = E(S) - \left[\frac{|S_{sunny}|}{|S|}E(S_{sunny}) + \frac{|S_{overcast}|}{|S|}E(S_{over}) + \frac{|S_{rain}|}{|S|}E(S_{rain})\right]$$

sunny: 2+, 3- ; overcast: 4+, 0- ; 3+, 2-

$$= 0.948 - \left[\frac{5}{14} \times (0.971) + \frac{4}{14} \times (0) + \frac{5}{14}(0.971)\right]$$

$$= 0.247$$

$$IG(S, Temp) = E(S) - \left[\frac{|S_{HOT}|}{|S|} \times E(S_{HOT}) + \frac{|S_{mild}|}{|S|} \times E(S_{mild}) + \frac{|S_{cool}|}{|S|} \times E(S_{cool})\right]$$

$$= 0.948 - \left[\frac{4}{14} \times \underset{2+,2-}{\overset{Hot}{(1)}} + \frac{6}{14} \times \underset{4+,2-}{\overset{mild}{(0.918)}} + \frac{4}{14} \underset{3+,1-}{\overset{cool}{(0.819)}}\right]$$

$$= 0.028$$

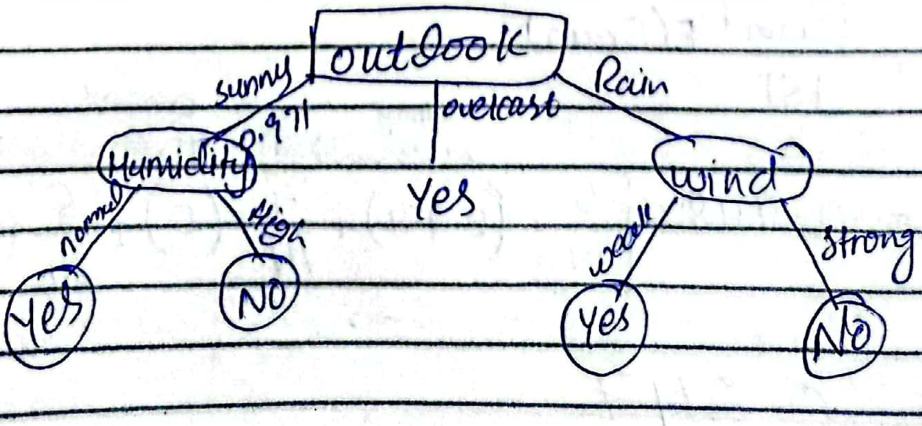
$$IG(S, Humidity) = E(S) - \left[\frac{|S_{High}|}{|S|} \times E(S_{High}) + \frac{|S_{Normal}|}{|S|} \times E(S_{Normal})\right]$$

$$= 0.948 - \left[\frac{7}{14} \times \overset{3+,4-}{(0.985)} + \frac{7}{14} \times \overset{6+,1-}{(0.592)}\right]$$

$= \cancel{0.151}$ 0.048

$$IG(S, Wind) = E(S) - \left[\frac{|S_{strong}|}{|S|} E(S_{strong}) + \frac{|S_{weak}|}{|S|} E(S_{weak})\right]$$

$= \cancel{0.048}$ 0.151



For Sunny:

$$IG(S_{sunny}, Humidity) = E(S_{sunny}) - \left[\frac{|S_{High}|}{|S|} E(S_{High}) + \frac{|S_{normal}|}{|S|} E(S_{normal})\right]$$

$$= 0.971 - \left[\frac{3}{5}(0) + \frac{2}{5}(0)\right]$$

$$= 0.971$$

$IG(S_{sunny}, Temp) = 0.571$

$IG(S_{sunny}, Wind) = 0.971$

FOR RAIN:

IG(

Linear Regression:-

Regression generate output value on real ~~value~~ line.

→ Classification ma YES or NO hota hai.

Regression Problem:-

Input → Output $\in R$

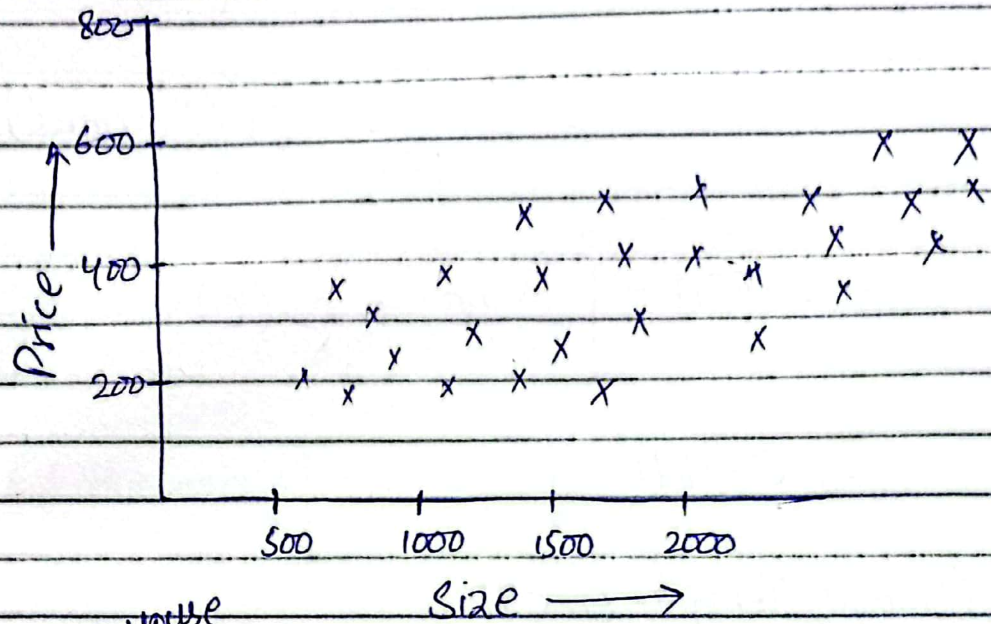
Classification:-

Input → Output $\in$ Discrete

$x^{(3)}$ $y^{(3)}$
(700, 600)

Suppose Supervised Learning for LR:

Model Representation;



Model could be straight line

Size of house $\nearrow$ X	Price $\nearrow$ Y
400	800 $\rightarrow x^{(1)}, y^{(1)} \rightarrow (400, 800)$
600	650
700	600 $\rightarrow$
350	

$\boxed{m \,\&\, c \text{ as } \theta_0, \theta_1}$

$y = mx + c$

$(x^{(i)}, y^{(i)})$ $\quad h_\theta(x) = \theta_0 + \theta_1 x$

m = total no. of exmp in dataset.
So here, m = 4

$\boxed{\text{Training Set}}$
$\downarrow$
$\boxed{\text{Learning Algo}}$
$\downarrow$

c = 4 Position (up or down).

Size of the house $\xrightarrow{\text{Input}}$ $\boxed{h_\theta(x)}$ $\xrightarrow{o/p}$ Price of House

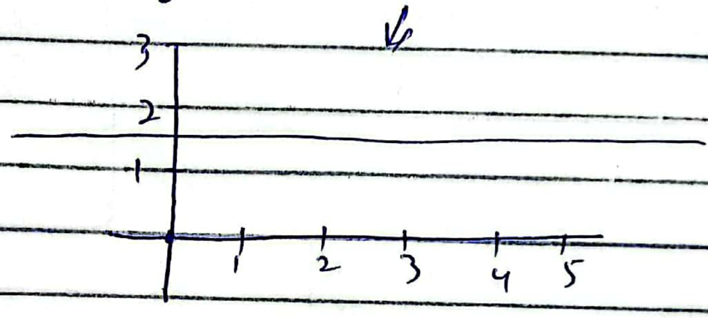
Our objective is to find the value of m and c on which it has minimum error from all the data.

Example:-

x	y
1	1
2	2
3	3

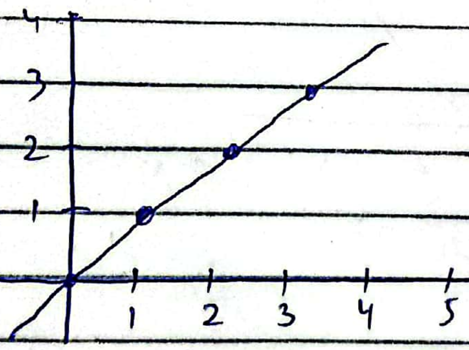
model $= \theta_0 + \theta_1 x$.

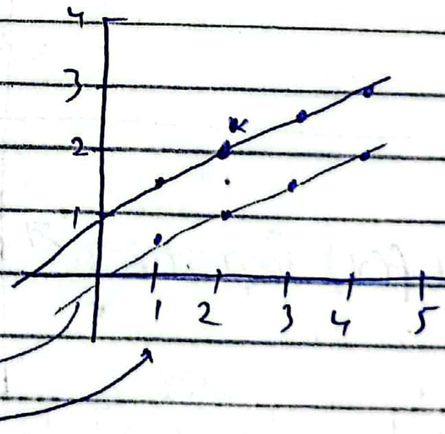
$h_\theta(x) = 1.5 + 0x$.



let say,

$h_\theta(x) = 0 + 0.5x$. $h_\theta(x) = 1 + 0.5x$.



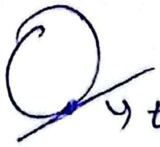


$\rightarrow h_\theta(x)$ $\rightarrow$ MSE

x	y	$\hat{y} = 2x+1$	$\frac{1}{2m}(\hat{y} - y)^2$
1	1	3	4
2	2	5	9
3	3	7	16

$29/3 = 9.7$

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x) - y\right)^2$$



↳ tangent line

Minimize θ_0, θ_1

Cost function:-

$$h_\theta(x) = \overset{0}{\theta_0} + \theta_1 x$$

let say,

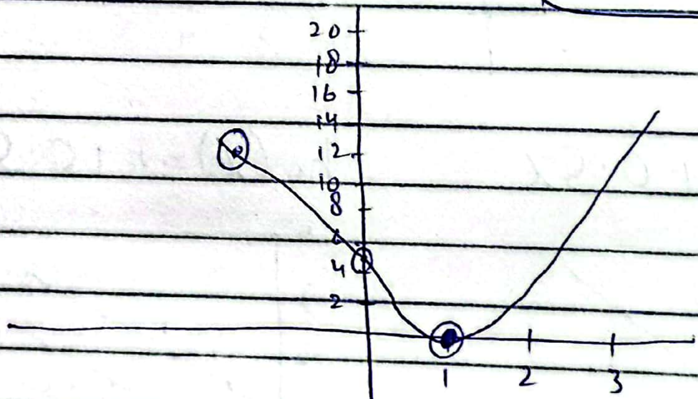
$$h_\theta(x) = \theta_1 x$$

$\theta_1 = 0$

Error,

$$h_\theta(x) = 0$$

x	y	$\hat{y} = 0$ $\hat{y} = x$	$(\hat{y} - y)^2$
1	1	0 / -1	1 / 4
2	2	0 / -2	4 / 16
3	3	0 / -3	9 36
			$14/3 = 4.7$
			$56/3 = 18.7$



Cost f(x) Represtation ↗ $\theta_1 \rightarrow$

$-5x$

x	y	$h_\theta(x)$	$(h_\theta(x) - y) \times x$
1	1	-5	$(-5-1) \times 1 = -6$
2	2	-10	$(-10-2) \times 2 = -24$
3	3	-15	$(-15-3) \times 3 = -54$
4	4	-20	
5	5	-25	θ
6	6	-30	

36
144
324
576
900
1296

$\theta_0 = 0, \theta_1 = 2.$

$h_\theta(x) = \theta_0 + \theta_1 x$

$h_\theta(x) = 0 + 2x$

$\frac{d}{dx} \rightarrow$ ~~isesa~~ hum tangent line find kr rhy hoty hain mtlb slope.

$$J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

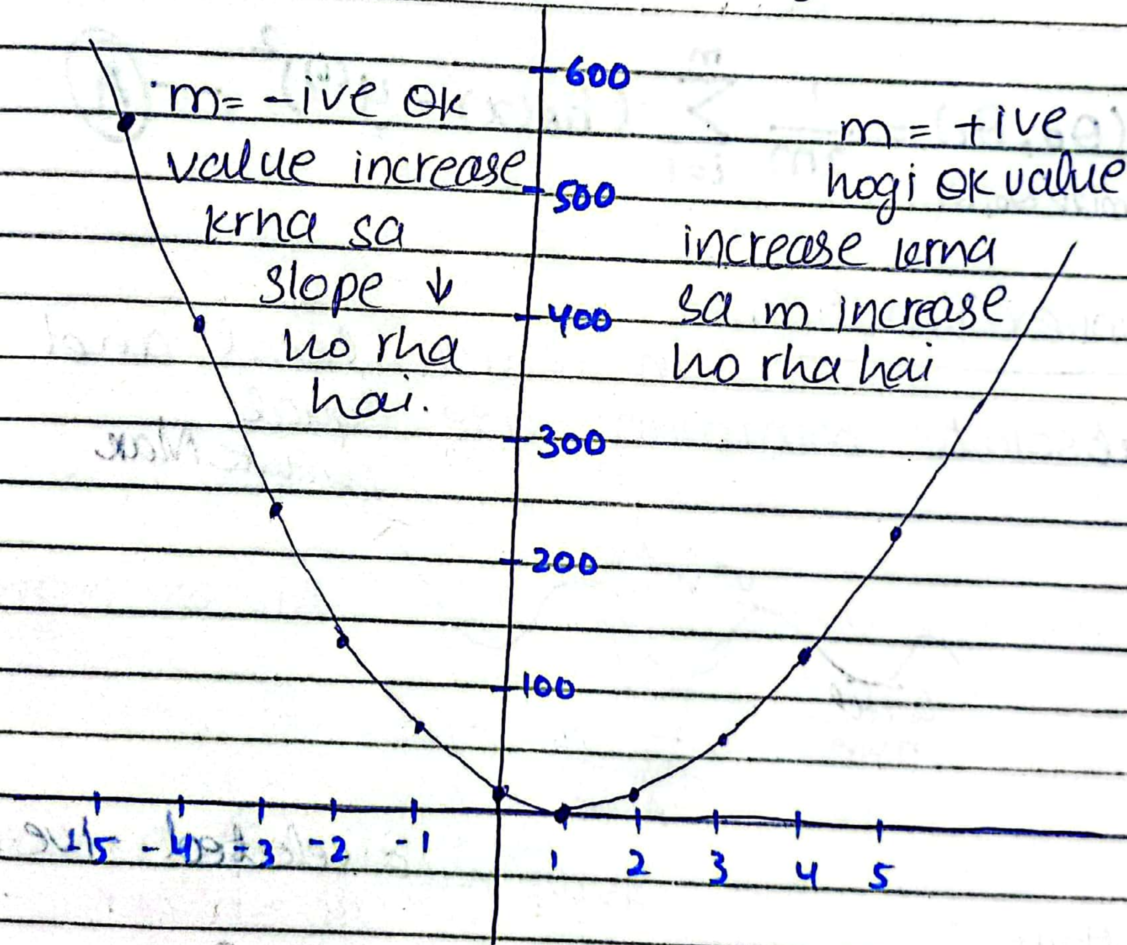
we are taking values b/w -5 — +5 of θ_1

Errors:

-6x	-5x	-4x	-3x	-2x	-x	0	x	2x	3x	4x	5x
	546	379	242	136	61	15	0	15	61	136	242

379

Drawing cost function; → it calculate the cost of our model



→ -ive side ma + krna hai and +ive side sa - krna hai.

$$\theta_1(new) = \theta_1(old) \pm \text{update}$$

$\frac{d\theta}{dx}$ of cost f(x)

$\theta_1(new) = \theta_1(old)$ — update

Gradient descent Algorithm:-

① Initialize θ with random number.
② Repeat until convergence:

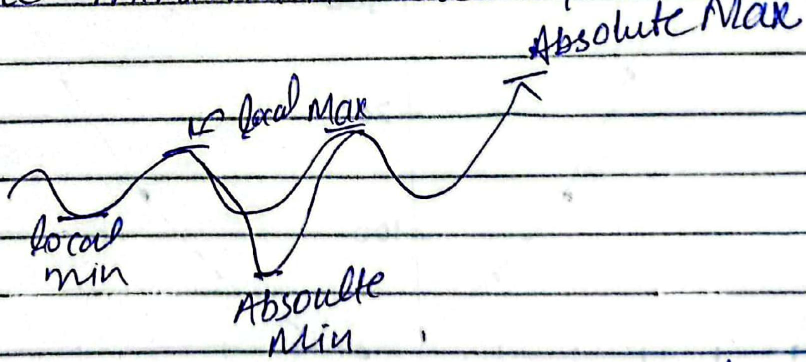
$$\theta_j = \theta_j - \alpha \frac{\partial J(\theta_0, \theta_1)}{\theta_j}$$

$$h_\theta(x) = \underbrace{\theta_0 + \theta_1}_{\text{unknown}} x^{\text{known}}$$

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x) - y^{(i)})^2 \quad \text{— (A)}$$

Minimize θ_0, θ_1

Convex function:- In which local and absolute minimum are equal



Continue:

As we have selected -ive sign

$$\theta_1(new) = \theta_1(old) - \alpha \Delta\theta_1$$

$$= \theta_1(old) - \alpha(\Delta\theta_1)$$

1. Initialize θ_j value randomly.
2. Repeate until Convergence

α = learning rate (fast or slow learn krty hain).
why use of α? ⟷ takeh bara change ekdm sa na aye. → learning slow hoti hai but divergence k chance khtm ho jaty hain — (B) if α is less.

3- $$\theta_j = \theta_j - \alpha \frac{\partial J(\theta_0,\theta_1)}{\partial \theta_j}$$

$$\theta_0 = \theta_0 - \alpha \frac{\partial J(\theta_0,\theta_1)}{\partial \theta_0}$$

$$\theta_1 = \theta_1 - \alpha \frac{\partial (J(\theta_0,\theta_1))}{\partial \theta_1}$$ — (C)

From eq (A), wrt θ_0.

$$= \frac{1}{2m}\sum_{i=1}^{m}\frac{\partial}{\partial\theta_0}(h_\theta(x)-y)^2$$

$$= \frac{1}{2m}\sum_{i=1}^{m}\frac{\partial}{\partial\theta_0}(\theta_0+\theta_1 x-y)^2$$

$$= \frac{1}{\cancel{2}m}\sum_{i=1}^{m}\cancel{2}(\theta_0+\theta_1 x-y)$$

From eq (B)

$$\theta_0 = \theta_0 - \alpha\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})\cdot 1$$

From eq (A), wrt θ_1.

$$= \frac{1}{2m}\sum_{i=1}^{m}\frac{\partial}{\partial\theta_1}(\theta_0+\theta_1 x-y)^2$$

$$= \frac{\cancel{2}}{\cancel{2}m}\sum_{i=1}^{m}(\theta_0+\theta_1 x-y)\frac{\partial}{\partial\theta_1}(\overset{0}{\cancel{\theta_0}}+\theta_1 \overset{x}{x}-\overset{0}{\cancel{y}})$$

In Previous Table

From eq ⓒ,

$$\theta_1 = \theta_1 - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)x^{(i)}\right]$$

Variant of Gradient descent Algorithm:-

model => $\theta_0 + \theta_1 x = h_\theta(x)$

Repeat until convergence {

$$\theta_j = \theta_j - \alpha \frac{\partial J(\theta_0, \theta_1)}{\partial \theta_j}$$

$$\theta_0 = \theta_0 - \alpha\left[(h_\theta(x) - y)\right] \quad (1)$$

$$\theta_1 = \theta_1 - \alpha\left[(h_\theta(x) - y)x\right] \quad (2)$$

convergence hum MSE sa identify krty hain.

$-1+4x$

x	y	$h(x)$ ($2x$)	MSE	$\Delta\theta_0$: $h_\theta(x)-y$	$\Delta\theta_1$: $[h_\theta(x)-y]x$	θ_0	θ_1
1	1	2	1	1	1	-1	4
2	2	4	4	2	4		
3	3	6	9	3	9		
4	4	8	16	4	16		
			$\frac{30}{4} = 7.5$	$\frac{10}{4} = 2.5$			

Put these values in eq (1) & (2) to update the θ_0 & θ_1 values.

Benefits of this Algor

Robust to noise.

→ Disadvantage ye hai k is ko boht zyada memory chahia load krna k liye

1- Gradient descent → $\boxed{m = 1 \text{ always}}$

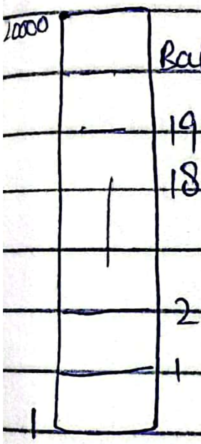
2- Stochastic Gradient descent (For fast learning) → Hr iteration pa ye value memory ma update kry ga.

Advantage → memory wala maslo es na solve kr diya sath sath learn kry ga.

→ Disadvantage ye hai k noisy entry model ko distrack kr dega.

Dono ma apny apny masly hain so dono ko mila k koi solution dekhty hai.

③ Batch Gradient descent (sol.)



Batch20 → es ma batch ka ek size hota hai according to our memory. Ek batch process hoga phr wo remove hoka agla aye ga.

→ Batch completely divisible hona chahia or size.

→ Batch size should be maximum phr thora thora km kr k fix krna hai, jis pa model run ho jaye phr wo size rakh dena hai batch ka.

Early stoping Criteria

① fixed no. of iterations.

② Ek specific value de dyn aur agr error us sa km ho jaye to stop ho jaye.

Multiple Linear Regression:-

$((x_1, x_2, x_3, x_4, \cdots x_n), y)$

$\underbrace{x_1, x_2, x_3, x_4, \cdots x_n}_{X}$

x_1	x_2	x_3	x_4	y
4	3	2	1	10
3	2	2	2	12

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n$$

$$\theta_0 = \theta_0 - \alpha \left[\frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)} \right) \right] \cancel{x}\ 1$$

$$\theta_1 = \theta_1 - \alpha \left[\quad " \quad \right] x_1$$

$$\theta_2 = \theta_2 - \alpha \left[\quad " \quad \right] x_2$$

$$\underset{X}{\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & \ddots & & \\ \vdots & & \ddots & \\ x_{m1} & & & x_{mn} \end{bmatrix}} \quad \underset{\text{model}}{\begin{bmatrix} \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}} \quad \underset{Y}{\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}}$$

$$h_\theta(x) = \overset{=1}{x_0}\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots$$

Expt:- [illegible]

$$\begin{bmatrix} 1 & 4 & 1 \\ 1 & 3 & 2 \\ 1 & 5 & 6 \end{bmatrix} \begin{bmatrix} \theta_0 = 0.5 \\ \theta_1 = 1 \\ \theta_2 = 1 \end{bmatrix}$$

$$h = \begin{bmatrix} (1\times0.5)+(4\times1)+(1\times1) \\ (1\times0.5)+(3\times1)+(2\times1) \\ (1\times0.5)+(5\times1)+(6\times1) \end{bmatrix} = \begin{bmatrix} 5.5 \\ 5.5 \\ 11.5 \end{bmatrix}$$

$1 = x_0 \xrightarrow{\theta_0} y$

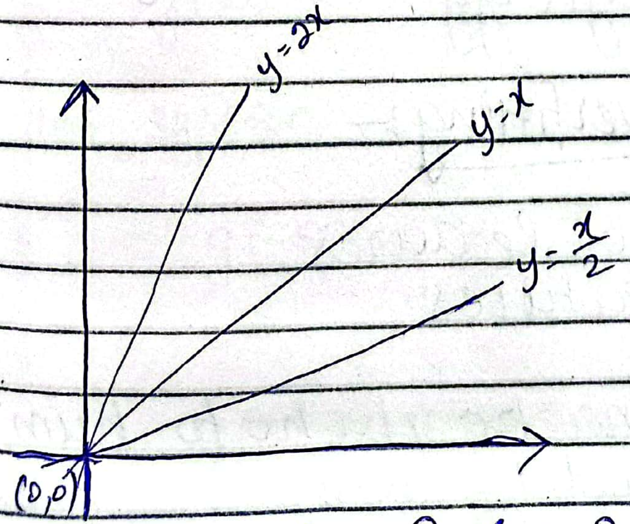
$x_1 \xrightarrow{\theta_1 x_1} y$

$x_2 \xrightarrow{x_2\theta_2} y$

$x_3 \xrightarrow{x_3\theta_3} y$

$x_0 = 1$ always in multileanear Regressio

x_1	x_2	x_3	$\theta_1 x_1$ $(x_1)^2$	$\theta_2 x_2$ $(x_2)^2$	$\theta_3 x_3$ $(x_3)^2$	y
3	4	2	9	16	4	
7	3	2	49	9	4	
6	2	1	36	4	1	
1	1	1	1	1	1	



Ye hum na multiple linear f(x)s ko combine kiya hai.

$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \quad -(1)$$

$\theta_0 \to 0$, $\theta_1 \to 0.5$, $\theta_2 \to 1$, $\theta_3 \to 2$

MLR → zero pa nhi move kr skty

Validation → Jb training error & validation error same direction ma ho to es ko hum generalization kehty hain

when $x=1$ in eq ① kehty hain

$0+0.5+1+2 = 3.5$

The above phenomenun is known as multipolynomial Regression it is like multilinear regression.

→ Overfitting:-

→ Jb training error km ho rha ho and validation b km ho rha ho to hum training continue rakhyn gy.

→ Jb training error km ho rha ho & validation error bhr rha ho to ye unknown data k liya nhi thk aur ye ab overfitting krna lg gya hai.

↳ Vice versa to this bhi training continue rakhyn gy.

To overcome Overfitting:-

① To manipulato features.
② To remove features.

→ Jb generalization na ho rhi ho to hum training stop kr lety hain

$m \times n + 1$

5000, 5 features

$\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4 + \theta_5 x_5$

5000, 100 features

$\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_{100} x_{100}$

no. of feature + 1 = Parameters.

→ Parameters = 101.

When we increase degree of polynomial, we will go towards fitness.

D → Data

$\mathbb{R}$ → Real

Z → Integers.

$D \in \mathbb{R} \longrightarrow H(x) \xrightarrow{O \in Z}$

Logistic Regression

Transformation of Linear Regression into 0 to 1.

$g(z) = \frac{1}{1+e^{-z}}$ → Mapping function.

let suppose $-50 - 0 - 50$

$g(-50) = \frac{1}{1+e^{-(-50)}} = \frac{1}{1+e^{50}} \approx 0$

$g(0) = \frac{1}{1+e^{0}} = \frac{1}{1+1} = \frac{1}{2} = 0.5$ → Threshold

$g(50) = \frac{1}{1+e^{-50}} = \frac{1}{1+\frac{1}{e^{50}}} = \frac{1}{1+0} = 1$

$\underline{LR}$ → ① Data → (x, y) → $-\infty < x < +\infty$
inp o/t $\quad -\infty < y < +\infty$ In Algebra → $\theta^T X$ → Matrix
② model $\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$
③ cost function = MSE = $J(\theta_0, \theta_1, \dots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x)^{(i)} - y^{(i)})$

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

Logistic Regression:-

$$(x^{(i)}, y^{(i)})$$

$-\infty < x < +\infty$ → output → $[0, 1]$ ← discrete

Model:- Logistic ma linear regression ko as an input pass krty hain.

$$g(\theta^T X) = \frac{1}{1 + e^{-\theta^T X}}$$

e.g.

x	y	$\theta_0 + \theta_1 x$ ($\theta_0 = 0$, $\theta_1 = 1$) → z	$1/1+e^{-z}$ = Logistic Reg
-5	0	-5	$1/1+e^{5} = 0.002$
5	1	5	$1/1+e^{-5} = 0.999$
1	1	1	$1/1+e^{1} =$
-2	0	-2	$1/1+e^{2} =$
-3	0	-3	$1/1+e^{3} =$
-1	1	-1	$1/1+e^{1} =$

{ why logistic f(x)? Threshhold define
kerny k liye.]

Lazy Learner Algorithm:-

K-nearest Neighbour :- → es ma pura data load krna prta hai.

1- Load the data

Test < 5.9 3 5.1 1.8 > (t_1 t_2 t_3 t_4)

	SL (x_1)	SW (x_2)	PL (x_3)	PW (x_4)	Species			Distance
1	5.1	3.5	1.4	0.2	0	Iris Setrosa		4.14
2	4.9	3	1.4	0.2	0	"		4.15
3	4.7	3.2	1.3	0.2	0	"		4.3
4	4.6	3.1	1.5	0.2	0	"		4.15
5	6.4	3.2	4.5	1.5	1	Iris Vercolor	2	0.86
6	6.9	3.1	4.9	1.3	1	"	3	1.07
7	5.5	2.3	4	1.3	1	"	5	1.45
8	6.7	2.5	5.8	1.8	2	Iris Virginica	4	1.17
9	7.2	3.6	6.1	2.5	2	"		1.88
10	6.5	3.2	5.1	2	2	"	1	0.66

2- Find the distance b/w test samples and traning data.

$$|S| = \sqrt{(x_1-t_1)^2+(x_2-t_2)^2+(x_3-t_3)^2+(x_4-t_4)^2}$$

3. Pick K-example with minimum distance from training.
   (i) Count each label for K-example.

4. Assign class label with max count to test example.

$$S = \sqrt{(5.1-5.9)^2 + (3.5-3)^2 + (1.4-5.1)^2 + (0.2-1.8)^2}$$
$$= 4.14$$

→ Ab sa closest distance lena hai pehla and that is 0.66
→ 5 closest distances liya hain.

Ab example jo liya hain un ko voting deni hai.

0 =		0/5
1 =	1+1+1	3/5
2 =	1+1	2/5

Ab en ko prob find krni hai.

→ As the highest prob 3/5 belongs to 1. So we can say that <5.9 3 5.1 1.8> belongs to Iris Vercolor.

→ K should be odd, not greater than the data set size.

→ Agr tie ho rha hai to k ki value increase kryn gy.

Evaluation Metrices:- (Performance metrices)

A1 } How accurate is your algorithm.
A2 }

X	Actual	model
x_1 x_2 - x_n	1	1
	0	1
	1	0
	0	0
	0	1
	1	1
	0	0
	1	0

Confusion Metrix.

Actual label

model Predicted	1	0
1	2	2
0	2	2

	1	0
1	True Positive (TP)	False Positive (FP)
0	False Negative (FN)	True Negative (TN)

① Accuracy:-

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

→ Only accuracy is not enough
e.g.
Test 100.
95 +
5 − but sys es ko bhi +ive consider krdy to TrueNeg ki value zero he

③ Recall

→ Class imbalance k waqt accuracy enough nhi hai.
→ If using accuracy measure then classes balance honi chahia.
→ If classes are imbalance phr hum classes/data ko balance kryn gy ya koi aur method use kryn gy. (Precision).

② Precision:-

$$\text{Precision} = \frac{TP}{TP+FP}$$

③ Recall:-

$$\text{Recall} = \frac{TP}{TP+FN}$$

e.g

60	15
10	15

↑ 100
70% 30%
60 10 15 15

$$\text{Accuracy} = \frac{60+15}{100} = 0.75$$

$$\text{Precision} = \frac{60}{75} = 0.80$$

$$\text{Recall} = \frac{60}{70} = 0.85$$ es ka mtlb ye k FN ma zyada value flip hoi hain.

④ F1 Score:-

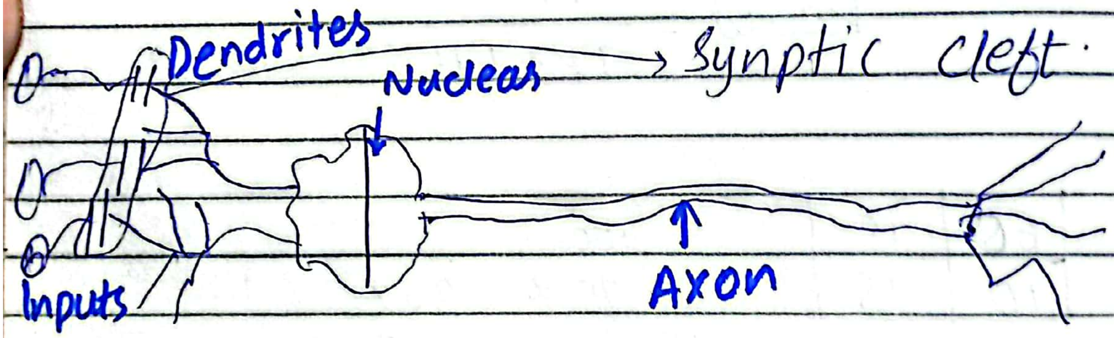
$$= 2 \cdot \frac{P \cdot R}{P + R}$$

$$= 2 \frac{0.8 \times 0.85}{0.8 + 0.85}$$

Neural Networks:-

→ Fully connected networks.
→ CNN
→ RNN.

→ "Inspiration from human brain".



Neuron's Functionalities:-

→ It calculates net sum. (Jitny b inputs aa rhy hain un sb ko sum krta hai and they can be negative or +ive and each neuron has a volt.
~~aur phr~~
→ And then threshold bnata hai ek specific.

Synptic Cleft is connection b/w two neurons.

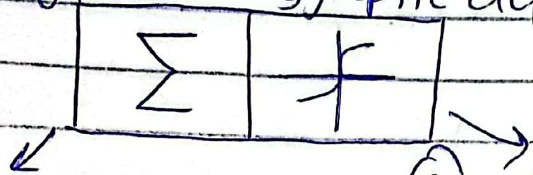
Learning phenomena → synptic cleft (sc)

∵ w_0 → weights

→ Jb syniptic cleft pa br br current pass ho k aktivate hota hai to resistance km hogi & vice versa

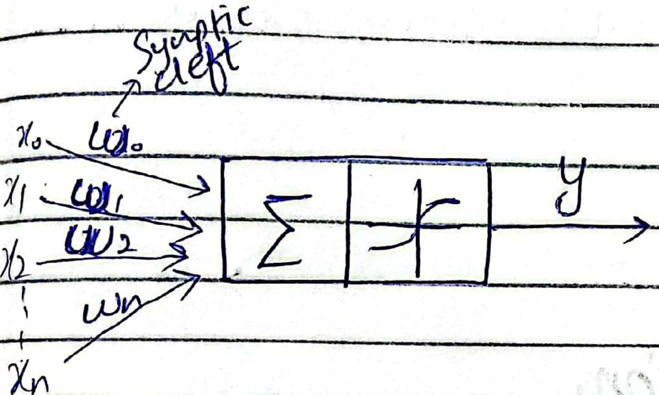
→ e.g. SC input & output dono sides pa ho skty hain.

→ weight ki jga hum synptic cleft rakhty hain



① Summation Function (weighted sum)

② Activation function (threshold)



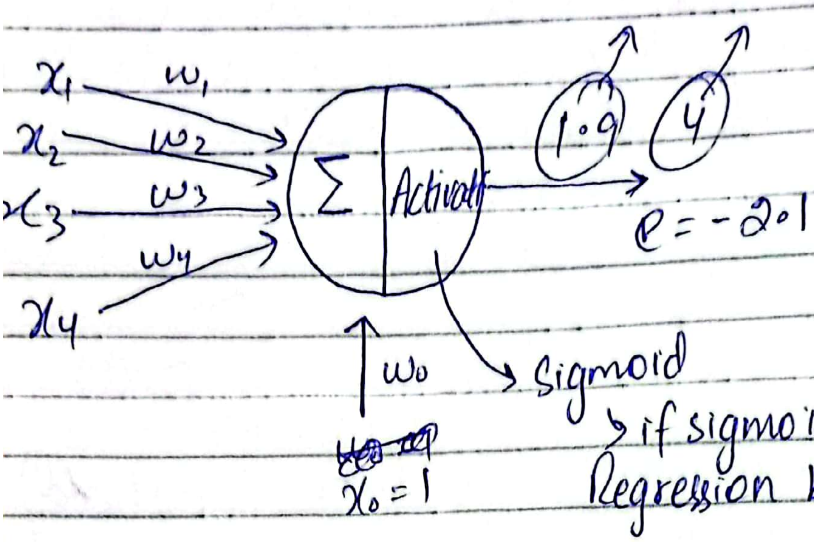
In multilinear Regression, → labels/classification

x_1	x_2	x_3	Y	Y
1	4	4	4	1
2	2	3	3.2	1
3	1	2	2.5	1
1	6	1	2	0
2	3	2	5.2	1

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

→ Learning parameters are weights here
→ Learning Algo gradient descent hi hoga.

In Neural Network,



→ if sigmoid to ye logistic Regression bn jayegi.

$$y_{input} = x_0 w_0 + x_1 w_1 + x_2 w_2 + x_3 w_3 + x_4 w_4$$

Activation function

$$y = y_{input}$$

In Logistic Regression,

Activation ⇒ $y = \begin{cases} 1 & 1/1+e^{-y_{input}} \geq 0.5 \\ 0 & \text{otherwise} \end{cases}$

Now Applying Gradient Descent;

① Initialize Randomly weights.

$w_0 = 0.2, w_1 = 0.1, w_2 = 0.2, w_3 = 0.3, w_4 = 0.1$

x_1, x_2, x_3 are from previous table.

$x_1 = 1, x_2 = 4, x_3 = 4$

MSE $(x^{(i)} - y^{(i)})^2$

$= 0.1 + 0.8 + 1.2 - 0.2$

$= 1.9$

$1.9 - 4 = -2.1 =$

error $= -2.1$

Updation,

$$\theta_0 = \theta_0 - \alpha(h_\theta(x) - y)\, x_j$$

(error written over $(h_\theta(x) - y)$)

$$w_i = w_i - \alpha(h_w(x)^i - y^{(1)})\, x_i$$

$-2.1 \times 1 = -2.1 = w_1$

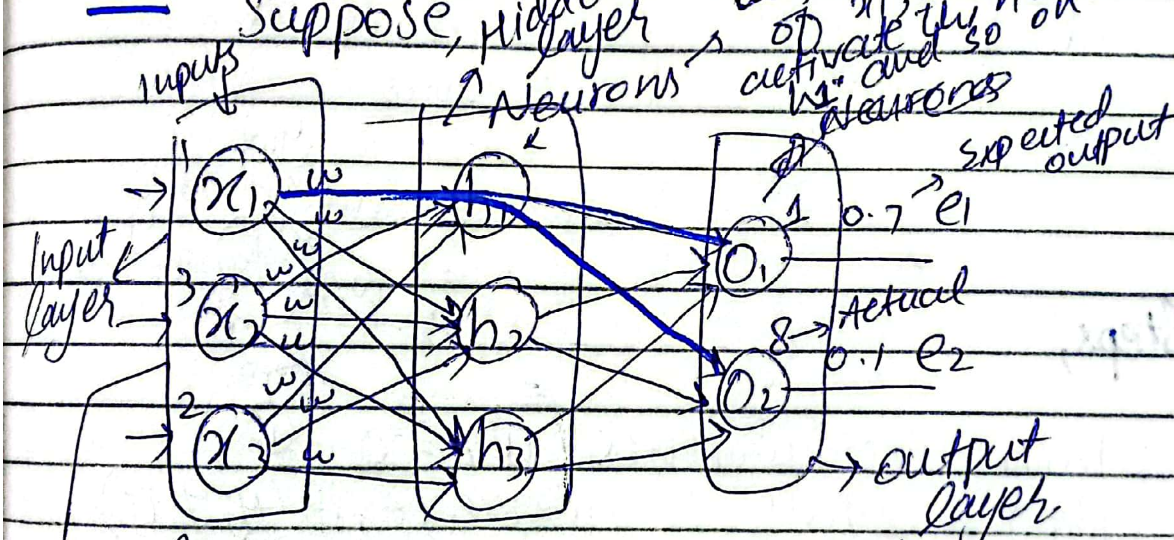
$-2.1 \times 4 = -8.4 = w_2$

$-2.1 \times 4 = -8.7 = w_3$

$-0.2 + 2.1 \times 1 = 1.9 = w_0$

Exp.

Suppose,



Ab ye complex problem hai, hamary pas 2 errors (e_1, e_2) hain aur ye nhi pta k kahan sa aarhy hain so we will use backPropagation algorithm to calculate & find this.

these are known as layers and it is 3 layer network & it is fully connected

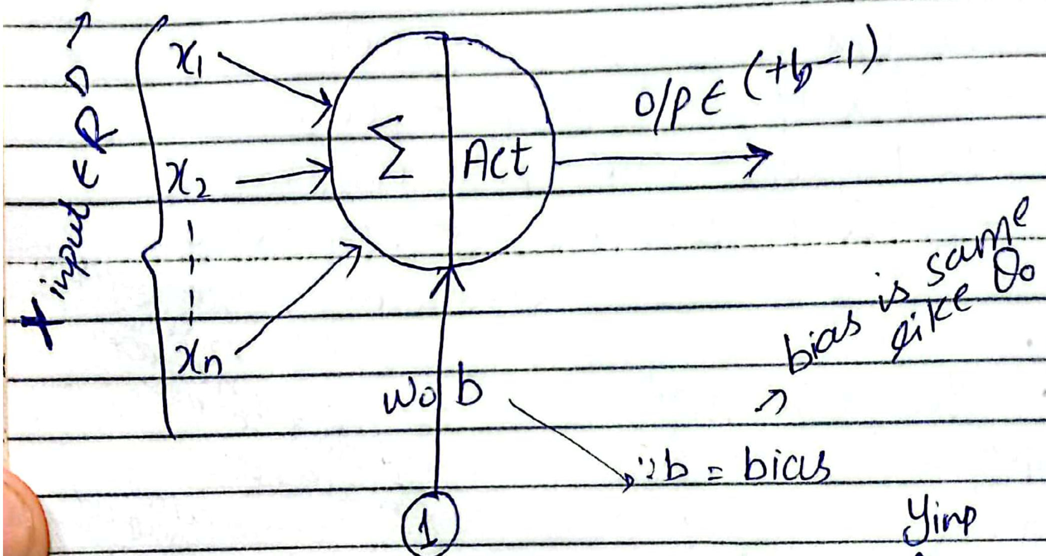
t_j = Target
y_j = Actual

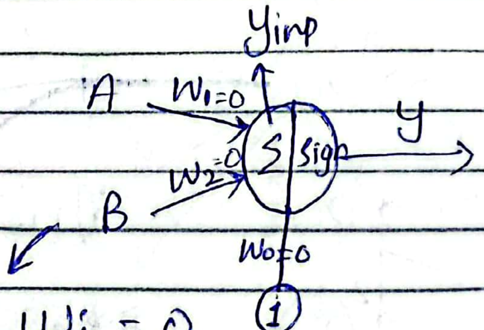
Perceptron Learning Algorithm:-

Sign function,

$$= \begin{cases} +1 & \text{if } h_\theta(x) \geq 0 \\ -1 & \text{otherwise} \end{cases} \quad \begin{array}{|l} \text{if } W^T X \geq 0 \\ \text{if } W^T X < 0 \end{array}$$

where D is no. of inputs





Steps,

1. Initialize weights $w_i = 0$
2. Repeat until convergence, if $(y_j \neq t_j)$

$$w_j = w_j + \alpha t_j x_j \longrightarrow (A)$$

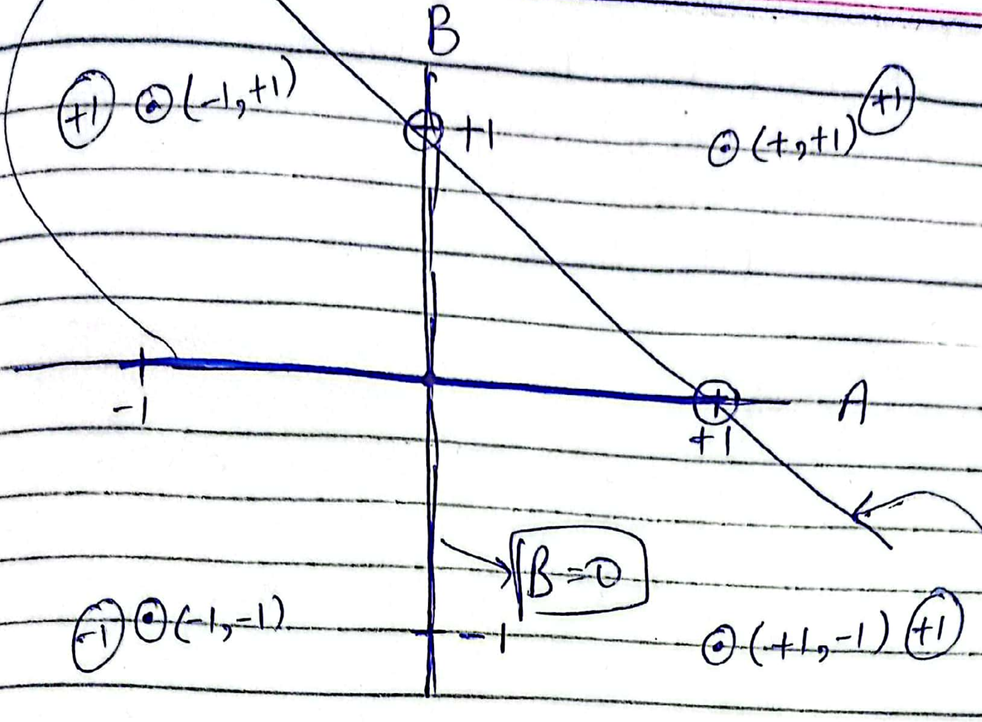
Example:, OR Gate $w^T x$

			$= w^T x$		0	0	0	t	$t \times A$	$t \times B$
A	B	t	y_{in}	y	W_0	W_1	W_2	ΔW_0	ΔW_1	ΔW_2
−1	−1	−1	0	+1	−1	1	1	−1	1	1
−1	+1	+1	−1	−1	0	0	2	+1	−1	+1
+1	−1	+1	−2	−1	+1	+1	+1	+1	+1	−1
+1	+1	+1	3	+1	1	1	1	+1		

Epoch 2,

$BW_2 = -AW_1 - W_0$.

$\boxed{B = 0}$ → Base line.



$y_{in} = W_0 + A W_1 + B W_2 + \cdot$

$0 + (-1)(1) + (1)(1)$

$-1 + \cancel{-1} + \cancel{1} = 0$

(1) $y_{in} = 0 + (-1)(0) + (-1)(0) = 0$

$= \begin{cases} +1 & \text{if } w^T x \geq 0 \\ -1 & \text{if } w^T x < 0 \end{cases}$ So $\boxed{y = +1}$

AS ω from eq (A).

$y_i \neq t_j \quad (-1 \neq +1)$

So $\quad w_j + w_j + \alpha t_j x_j$ where $\alpha = 1$

$W_0 = 0 + (-1) \to \Delta W_0 \quad = -1$

$W_1 = 0 + 1 \to \Delta W_1 \quad = +1$

$W_2 = 0 + 1 \to \Delta W_2 \quad = +1$

$0 + -1 = (-1)$

updated w,

$w_0 = w_0 + \Delta w_0$

$w_1 = w_1 + \Delta w_1 = 1$

$W_2 = W_2 + \Delta W_2$

$0 \quad 1 = 1$

$AW_1 + BW_2 + W_0 = 0$

$A + B - 1 = 0$

$\boxed{B = 1 - A}$

A	B
1	0
0	1

(+1, -1)

(-1, +1)

XOR is inap.

(2) $AW_1 + BW_2 + W_0 = 0$
$A(0) + B(2) + 0 = 0$

$\boxed{B = 0}$

(3)

(4) In example 4, $y = t_j$ so update nhi krun gy ab, previous weights hi update krun gy.

Epoch 2.

A	B	t	Yin	Y	w_0	w_1	w_2	Δw_0 (= t)	Δw_1 (t×A)	Δw_2 (t×B)
−1	−1	−1	−1	−1	+1	+1	+1	−1		
−1	+1	+1	+1	+1				+1		
+1	−1	+1	+1	+1				+1		
+1	+1	+1						+1		

BackPropagation:-

It manages gradient flow.

O_1 } output neurons
O_2 }

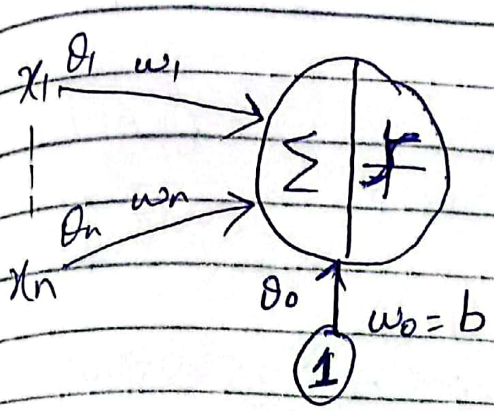
(Kx100)

sum h_1 = Net sum
out h_1 = Sigmoid

ANN Training:-

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$

$$out\ h_1 = \frac{1}{1+e^{-sum}}$$



Vanishing Gradient Problem

$$\theta_j = \theta_j - \alpha \frac{\partial cost}{\partial \theta_j}$$

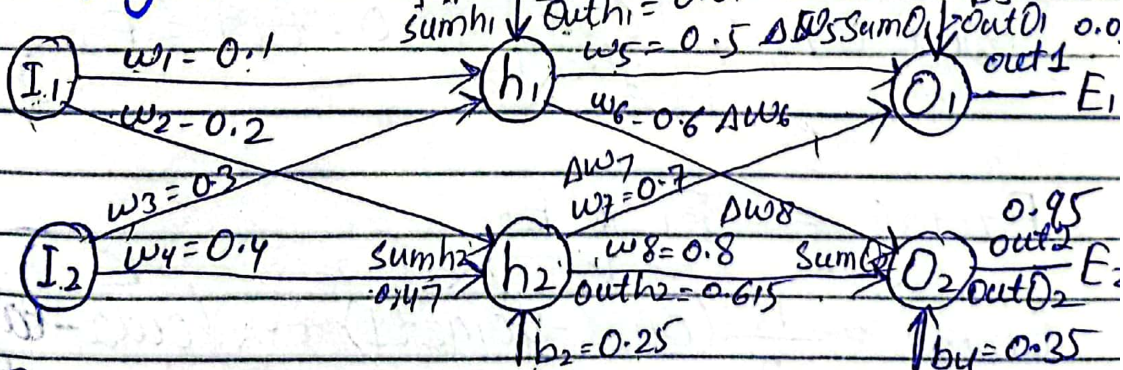
$$\boxed{(h_\theta(x) - y)\cdot x}$$

Example:-

mean that we have 2 output neurons

X		Target	
I_1	I_2	target1	target2
0.1	0.5	0.05/0	0.95/1

Input layer



$E_{Total} = E_1 + E_2$
↳ sum of all E_n if greater than 2 E

$$\frac{dy}{dx} = \frac{dy}{dz} * \frac{dz}{dx}$$

forward Pass

$$\text{sum } h_1 = w_1 I_1 + w_3 I_2 + b_1$$
$$= 0.1 \times 0.1 + 0.3 \times 0.5 + 0.25 = 0.41$$

$$\text{Out } h_1 = \frac{1}{1+e^{-\text{sum } h_1}} = \frac{1}{1+e^{-0.41}} = 0.601$$

$$\text{sum } h_2 = w_2 I_1 + w_4 I_2 + b_2$$
$$= 0.2 \times 0.1 + 0.4 \times 0.5 + 0.25 = 0.47$$

$$\text{Out } h_2 = \frac{1}{1+e^{-\text{sum } h_2}} = \frac{1}{1+e^{-0.47}} = 0.615$$

$$\text{Sum } O_1 = \text{out } h_1 \times w_5 + \text{out } h_2 \times w_7 + b_3$$
$$= 0.601 \times 0.5 + 0.615 \times 0.7 + 0.35 = 1.0197$$

$$\text{out } O_1 = \frac{1}{1+e^{-\text{sum } O_1}} = \frac{1}{1+e^{-1.0197}} = 0.735 = E_1$$

$$\text{Sum } O_2 = \text{out } h_1 \times w_6 + \text{out } h_2 \times w_8 + b_4$$
$$= 0.601 \times 0.6 + 0.615 \times 0.8 + 0.35 = 1.263$$

$$\text{Out } O_2 = \frac{1}{1+e^{-1.263}} = 0.779 = E_2$$

$$E_{Total} = E_1 + E_2$$

$$E_{total} = \underbrace{\frac{1}{2}(\text{out1} - \text{target1})^2}_{E_1} + \underbrace{\frac{1}{2}(\text{out2} - \text{target2})^2}_{E_2}$$

1

If we want derivative wrt w_5 then we found Δw_5 and if w_6 then Δw_6 and so on:

Chain Rule,

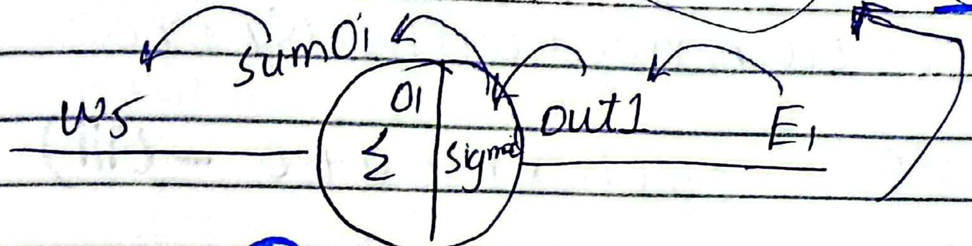
If we have $y = f(x)$

and $u = g(x)$

$$y = f(g(x))$$

$$\frac{dy}{dx} = \frac{dy}{du} \cdot \frac{du}{dx}$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial Out1} \times \frac{\partial out1}{\partial sumO_1} \times \frac{\partial sumO_1}{\partial w_5} \quad \text{—— (A)}$$



From eq (1)

$$\frac{\partial E_{total}}{\partial Out1} = \frac{\partial}{\partial out1}\left[\frac{1}{2}(out1 - targt1)^2\right] + \frac{\partial}{\partial out2}\left[\frac{1}{2}(out2 - targt2)^2\right]$$

$$= \frac{\cancel{2}}{\cancel{2}}(out1 - targt1) \cdot \frac{\partial}{\partial out1}(out1 - targt1) +$$

$$\frac{\cancel{2}}{\cancel{2}}(out2 - targt2)^{0} \cdot \frac{\partial}{\partial out2}(out2 - targt2)^{0}$$

$$= (out1 - targt1) \cdot 1 + 0 \quad \text{—— (i)}$$

Sigmoid of x
$\sigma(x)$

$$y = \frac{1}{1+e^{-x}} \Rightarrow \frac{1}{1+e^{-sumO_1}}$$

$$\frac{dy}{dx} = \frac{1}{1+e^{-x}}\left(1-\frac{1}{1+e^{-x}}\right)$$

sumh1   sumh1

$$= \sigma(x)(1-\sigma(x))$$

$$\boxed{\frac{\partial out1}{\partial sumO_1} = out1(1-out1)} \text{ — (ii)}$$

(A)

$$sumO_1 = outh_1 \times w_5 + outh_2 \times w_7 + b3.$$

$$\frac{\partial sumO_1}{\partial w_5} = outh_1 \times 1 + 0 + 0 \text{ — (iii)}$$

Put (i) (ii) & (iii) in eq (A),

$$\frac{\partial E_{total}}{\partial w_5} = (out1 - target1) \times out1(1-out1) \times outh_1$$

$$\Delta W_5 = (0.735-0.05) \times 0.735(1-0.735) \times 0.601$$

$$\Delta w_5 = 0.0806 \quad \text{Ans.}$$

As $w_5 = 0.5$, so

$$= w_5 - \Delta w_5$$

$$= 0.5 - 0.0806$$

$$\text{New } w_5 = 0.4794.$$

Now wrt w_7

these both are same

$$\frac{\partial E_{total}}{\partial w_7} = \frac{\partial E_{total}}{\partial out_1} \times \frac{\partial out_1}{\partial sumO_1} \times \frac{\partial sumO_1}{\partial w_7}$$

$$\frac{\partial sumO_1}{\partial w_7} = out\,h_2$$, Now put in the above equation.

wrt w_8

$$\Delta w_8 = \frac{\partial E_{total}}{\partial out_2} \times \frac{\partial out_2}{\partial sumO_2} \times \frac{\partial sumO_2}{\partial w_8}.$$

$$\frac{\partial out_2}{\partial sumO_2} = \frac{1}{1+e^{-sumO_2}}$$

$$= \sigma(sumO_2)(1-\sigma(sumO_2))$$

$$= out_2(1-out_2).$$

wrt w_6

$$\Delta w_6 = \frac{E_{total}}{\partial w_6} = \frac{\partial total}{\partial out_2} \times \frac{\partial out_2}{\partial sumO_2} \times \frac{\partial sumO_2}{\partial w_6}$$

ll

wrt w_1

Complex a little bit

Δw_1

$$E_{total} = E_1 + E_2$$

$$\Delta w_1 = \frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_1}{\partial w_1} + \frac{\partial E_2}{\partial w_1} \quad \text{B}$$

From eq (B),

$$\frac{\partial E_1}{\partial w_1} = \frac{\partial E_1}{\partial out1} \times \frac{\partial out1}{\partial sum O_1} \times \frac{\partial sum O_1}{\partial out h_1} \times \frac{\partial out h_1}{\partial sum h_1} \times \frac{\partial sum h_1}{\partial w_1} \quad -(1)$$

$$+$$

$$\frac{\partial E_2}{\partial w_1} = \frac{\partial E_2}{\partial out h_2} \times \frac{\partial out h_2}{\partial sum O_2} \times \frac{\partial sum O_2}{\partial out h_1} \times \frac{\partial out h_1}{\partial sum h_1} \times \frac{\partial sum h_1}{\partial w_1} \quad -(2)$$

Put (1) & (2) in eq (B)
then find eqs of each part in eq B.

$$\frac{\partial E_1}{\partial out1} = \cancel{(out1 - targt1)}$$

$$\frac{\partial out1}{\partial sum O_1} = out_1(1 - out1)$$

$$\frac{\partial sum O_1}{\partial out h_1} = w_5$$

$$\frac{\partial out h_1}{\partial sum h_1} = out h_1(1 - out h_1)$$

$$\frac{\partial sum h_1}{\partial w_1} = I_1$$

$$\frac{\partial E_2}{\partial out_{h_2}} = (out2 - target2).$$

$$\frac{\partial out2}{\partial sumO_2} = out2(1-out2).$$

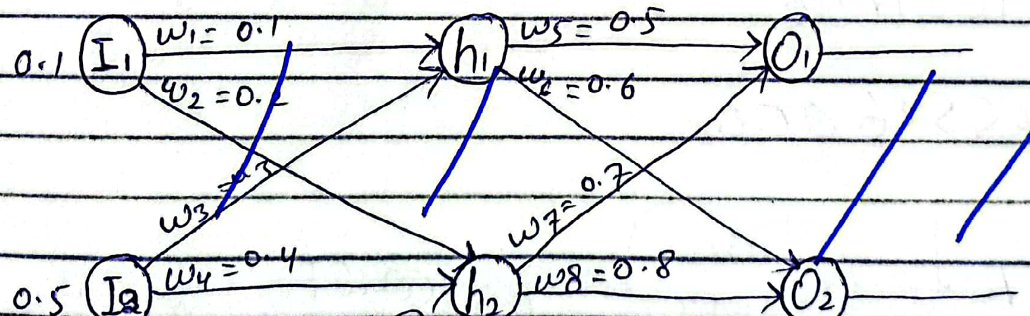
$$\frac{\partial sumO_2}{\partial out\,h_1} = w_6.$$

$$\frac{\partial out\,h_1}{\partial sum\,h_1} = out\,h_1(1-out\,h_1)$$

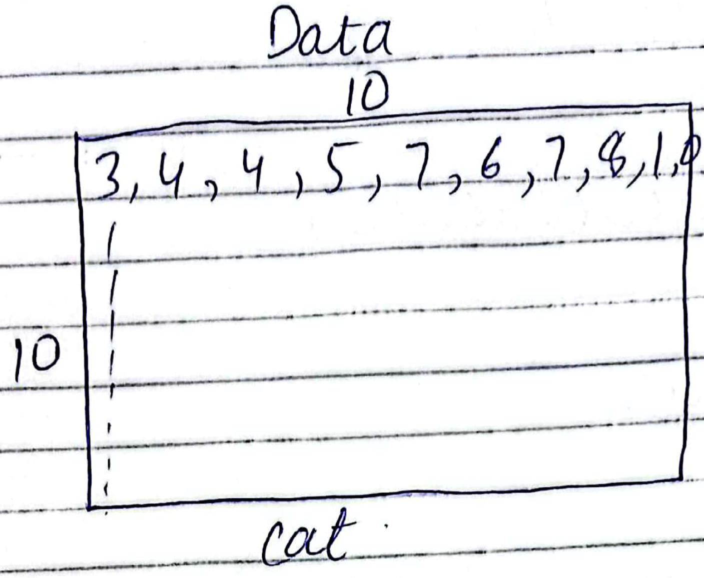
$$\frac{\partial sum\,h_1}{\partial w_1} = I_1$$

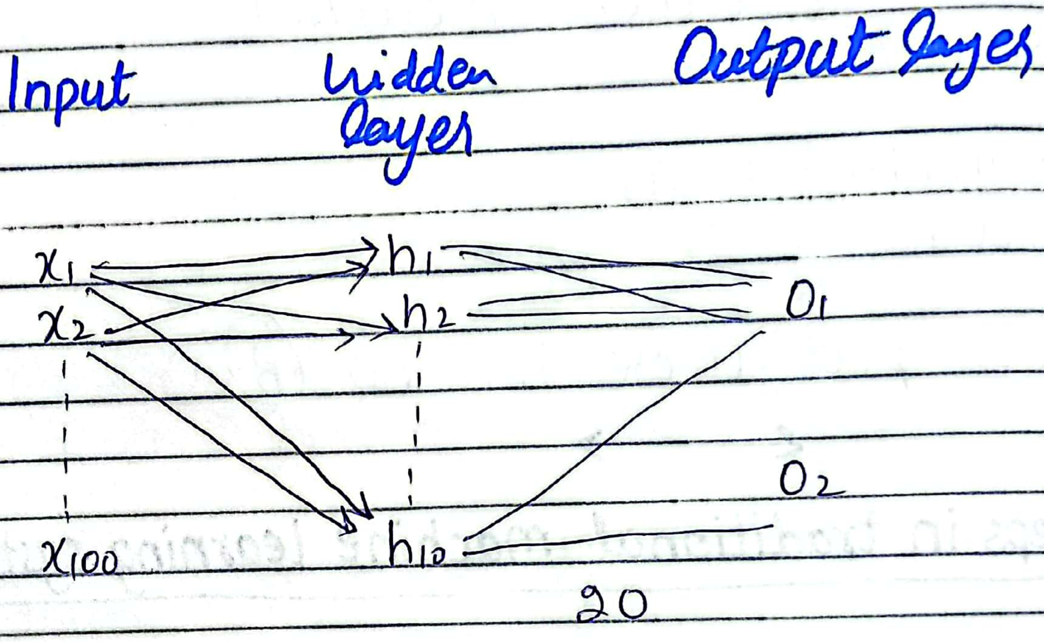
Now put these in eq (B).

Steps in traditional machine learning system



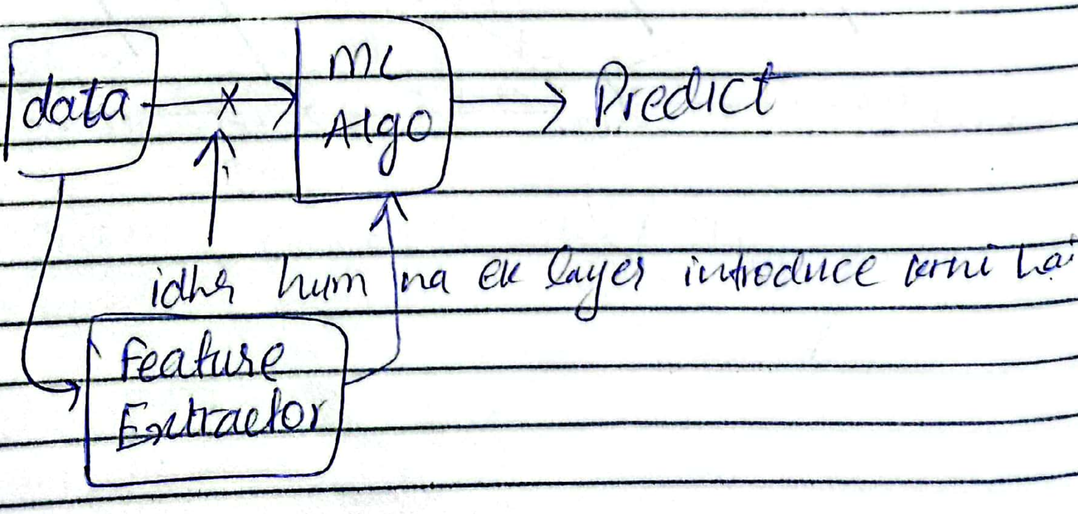
$\boxed{\Delta w_1 = 1}$





w=1000

65536000



R	G	B	
255	255	255	→ white
0	0	0	→ black
255	0	0	Red.

x_1		mean(R)	mean(B)	mean(G)	
200	I_1	255	240	239	dog
200	I_2	254	230	240	dog
60	I_3	160	150	135	cat
2	|				
2	|				

In

$\boxed{V=0}$ Bigger data se features extract kry gy complexity ko reduce kr liya

① → If no variance then discard that feature

X_2 → es trh hum ek 2 inputs nhi dey so es ka corelation find kryn gy.

180
180
50
|
|

② → If X_1 & X_2 has high co-relation mean that both are same, we will discard one of them.

③ If relation is exist b/w class & feature, it means its best

Convolution:-

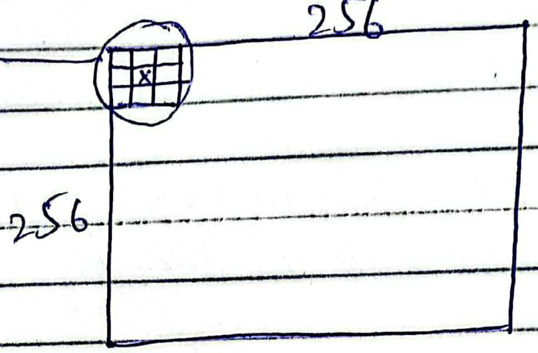
Automatic Feature Extraction:-

x_1	x_2	x_3
x_4	x_5	x_6
x_7	x_8	x_9

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

3x3

$$z = x_1 w_9 + x_2 w_8 + x_3 w_7 + x_4 w_6 + x_5 w_5 + x_6 w_4 + x_7 w_3 + x_8 w_2 + x_9 w_1$$



$$\text{Co-relation} = z = x_1 w_1 + x_2 w_2 + \cdots + x_9 w_9$$

$$= \sum_{i=1}^{9} x_i w_i$$

180°

5	3	2
1	4	8
3	2	1

Non-Symmetric

0	-1	0
-1	5	-1
0	-1	0

Symmetric

If symmetric, then convolution is same to corelation

Data,

CNN

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8 & 3 & 6 & 16 & 0 \\ 0 & 0 & 42 & 13 & 20 & 0 \\ 0 & 6 & 26 & 0 & 8 & 0 \\ 0 & 44 & 47 & 49 & 27 & 0 \\ 0 & 5 & 42 & 42 & 19 & 0 \\ 0 & 11 & 50 & 25 & 48 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

$$\begin{bmatrix} 1 & 1 & 1 \\ -1 & -1 & -1 \\ 0 & -1 & -1 \end{bmatrix}$$

(8) → (-1)

hum krty correlation hn lekin convolution ~~use~~ name use krty hn.

pehly 8 → -1 sy main points rkhty hn jab calculation ho jaye to points position move kr jati ha.

next (3) → (-1) hoga.

jab 3 → 6 py jaen sy aor (-1) bi saath phr hum neech shift honey.

8

(0) → (-1) sy next row ka 1st element start krna ha.

CNN. Classification Regression.

$\boxed{\text{Fully Extractor}}$ — $\boxed{\text{Fully connected Network}}$

- → Convolution Layer.
- → Pooling Layer.

Convolution layer:-

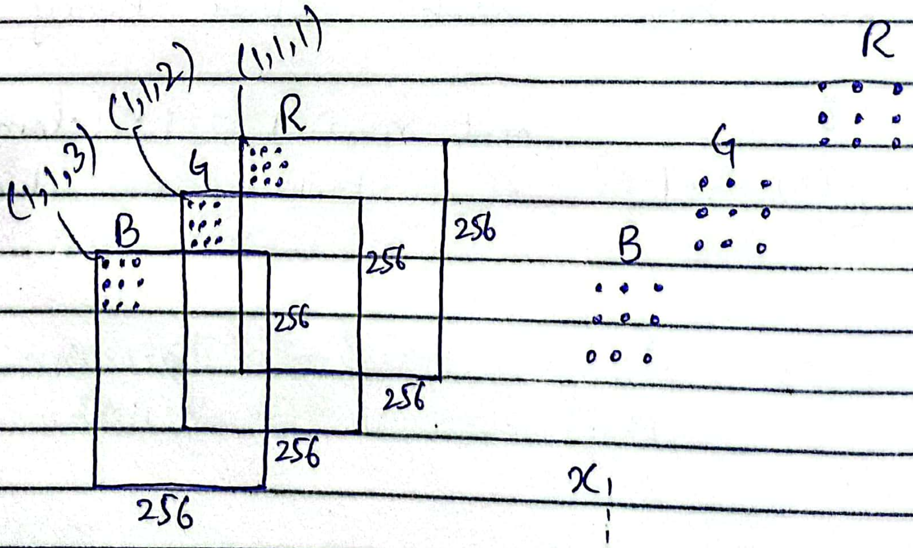
- → Filter size (How much region or Actual size en 2 sa actual network define hota hai.)
- → Number of filters
- → Padding
- → Stride.

Pooling Layer:-

- → Pooling size
- → Stride

Let Say, I 256x256x3

cap/shirt



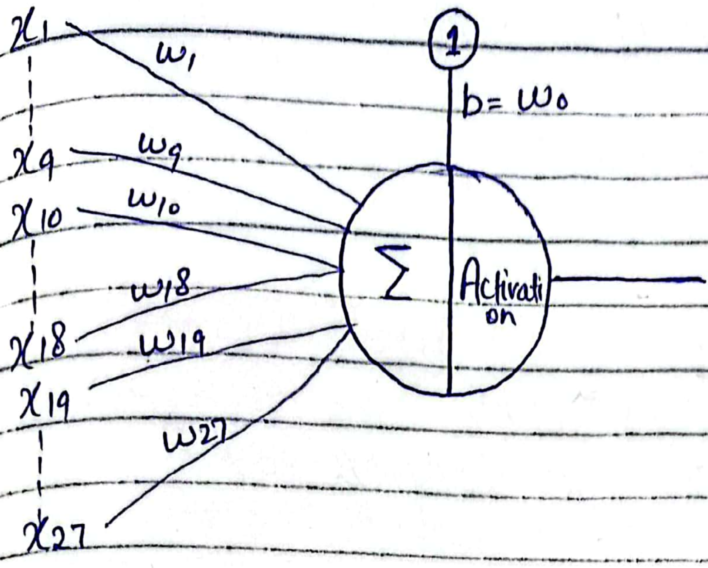
x_1

x_9

x_{10}

x_{18}

x_{19}



Pooling Layer → [Pool size, Stride]

conv1
FS → 3x3
NF → 10
P = False
S = 1

↓ ① ← output

Pool1
PS = 2x2
S = 2

→ ②

FS = 3x3
NF → 8
P = True
S = 2

② ↑

Pool2
PS = 2x2
S = 2

①

$$H' = \frac{(H+F+2P)}{S} + 1$$

$$= \frac{256-3+0}{1} + 1$$

$$= 254$$

and eye 254x254x10 mtlb 10 matrices ~~ate~~ generate hongy.

127 127

② As here input = 254

$$H' = \frac{(254-2+0)}{2} + 1$$

$$= 127$$

and es ma 127x127x10 aur pooling layer ma depth change nhi hogi same 10 hi hogi.

③ As now input = 127.

$$H' = \frac{127-3+2(1)}{2} + 1$$

$= 64$.

and now $64 \times 64 \times 8$ as there are 8 filters so depth = 8.

④ For Pooling.

$$H' = \frac{64-2+0}{2} + 1$$

$= 32$.

and now its $32 \times 32 \times 8$ as depth will remain same.

⑤ Conv 3

F = 3 x 3
NF = 4.
S = 1
Padding = T

$32 \times 32 \times 4$.

⑥ Pooling:-

PS = 2x2
S = 2

$16 \times 16 \times 4$.

If 5x5 & we dont want overlapping then stride will be 5

Clustering :-

Convolution Networks

(i) Convolution.
(ii) Pooling
- a) Max Pool
- b) Average Pool.

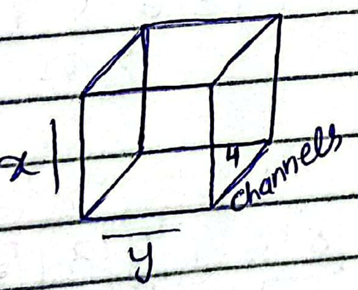
Attention Mechanisms:-

- Special Attention.
- Channel Attention.

Channel Attention:-

It returns a vector.

→ Input will a feature matrix.



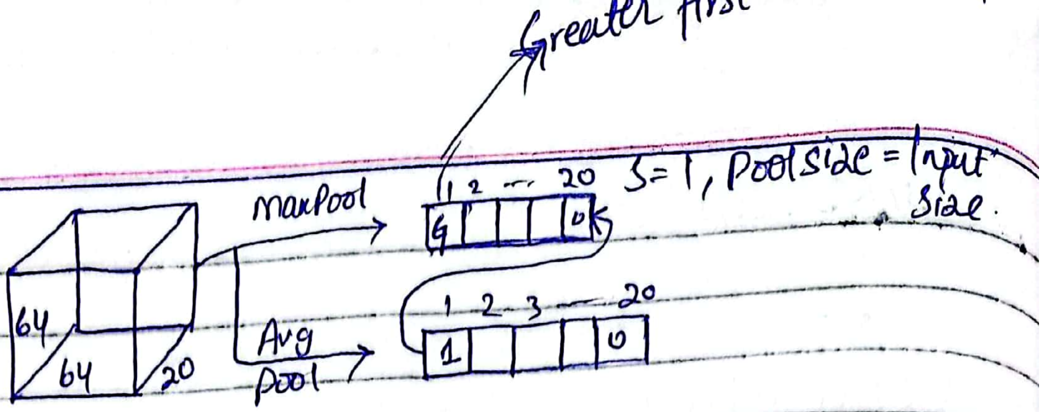
If value is near to 1 mean that its important and if near to zero mean that not important.

1	2	3	4
0	0.8	0.1	0.2

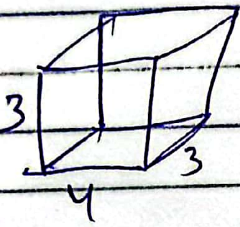
e.g. 64x64x20

1	2	3	4		20
0	0.8	0.1	0.2	- -	0.3

18



e.g.



3	4	8	1
4	3	2	1
3	2	4	1

4 2

0.8

3	4	8	1
4	3	2	1
3	2	4	1

Avg Poo $= \frac{36}{12} = 3$

0.7

4	2	1	6
3	2	1	4
4	8	2	1

$\frac{38}{12} = 3.1$

0.2

4	8	2	1
9	1	6	2
3	8	5	7

$\frac{56}{12} = 4.5$

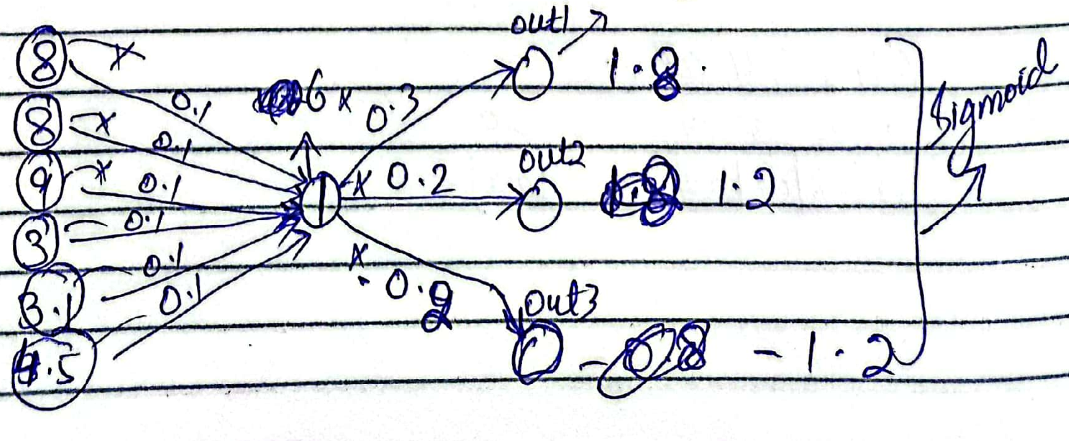
size = 3x4
maxPool → 8 8 9

8	8	9	3	3.1	4.5

$\underbrace{8\ 8\ 9}_{\text{max}}$ $\underbrace{3\ 3.1\ 4.5}_{\text{Average}}$

size = 3x4
Avg Pool → 3 3.1 4.5

∴ we have 3 channels.



$= 0.8 + 0.8 + 0.9 + 0.6 + 0.9 + 2 = 6$

$$= \frac{1}{1+e^{-1.8}} = 0.8$$

$$= \frac{1}{1+e^{-1.2}} = 0.7$$

$$= \frac{1}{1+e^{1.2}} = 0.2.$$

1

2.4	3.2	6.4	0.8
3.2	2.4	1.6	0.8
2.4	1.6	3.2	0.8

2

2.8	1.4	0.7	4.2
2.1	1.4	0.7	2.8
2.8	5.6	1.4	0.7

3

0.8	1.6	0.4	0.2
1.8	0.2	1.2	0.4
0.6	1.6	1.0	1.4

New channels (Updated).

Special Attention:-

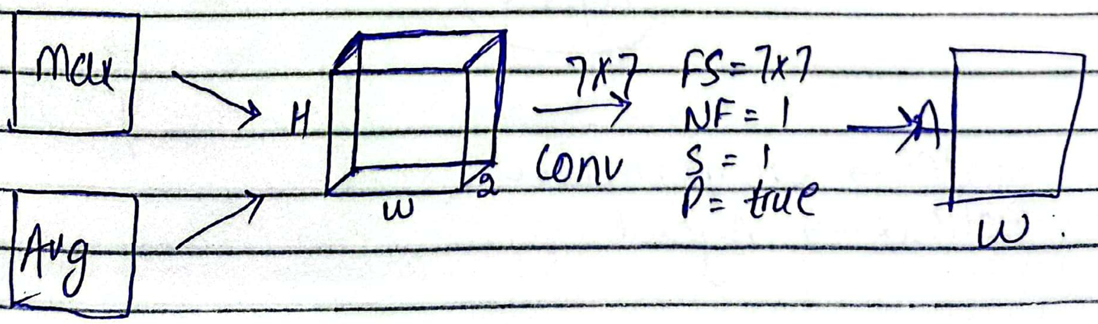
3	4	8	1
4	3	2	12
3	2	4	1

4	2	1	6
3	2	1	42
4	8	2	1

4	8	2	1
9	1	6	22
3	8	5	7

Max Pool →

Avg Pool →



General

Ab es matrix ko one to one
un peehle 3 matrixes se
multiply krna hai to
ab new matrixes ~~generate~~
hongi.

Max →
$$\begin{bmatrix} 4 & 8 & 8 & 6 \\ 9 & 3 & 6 & 4 \\ 4 & 8 & 5 & 7 \end{bmatrix}$$

Avg →
$$\begin{bmatrix} 3.7 & 4.7 & 2.4 & 2.4 \\ 5.2 & 2 & 3 & 2.1 \\ 3.2 & 6 & 3.2 & 3 \end{bmatrix}$$

$$\begin{bmatrix} 0.3 & 0.2 & 0.4 & 0.3 \\ 0.2 & 0.9 & 1 & 0.8 \\ 0.2 & 1 & 1 & 1 \end{bmatrix}$$ → 1 map region

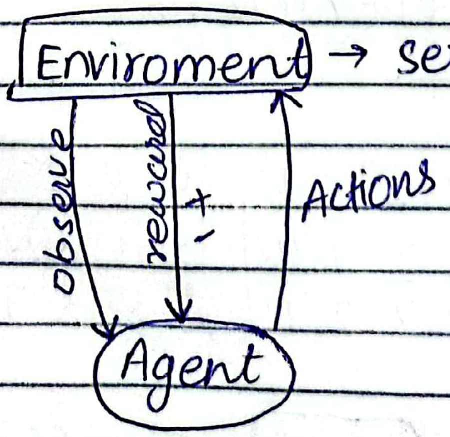
1 →
$$\begin{bmatrix} 0.9 & 0.8 & 3.2 & 0.3 \\ 0.8 & 2.7 & 2 & 0.8 \\ 0.6 & 2 & 4 & 1 \end{bmatrix}$$

2
$$\begin{bmatrix} 1.2 & 0.4 & 0.4 & 1.8 \\ 0.6 & 1.8 & 1 & 3.2 \\ 0.8 & 8 & 2 & 1 \end{bmatrix}$$

3
$$\begin{bmatrix} \end{bmatrix}$$

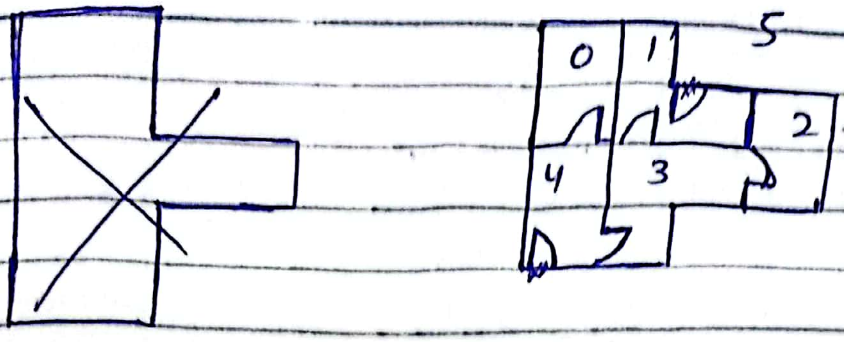
Q-Learning:-

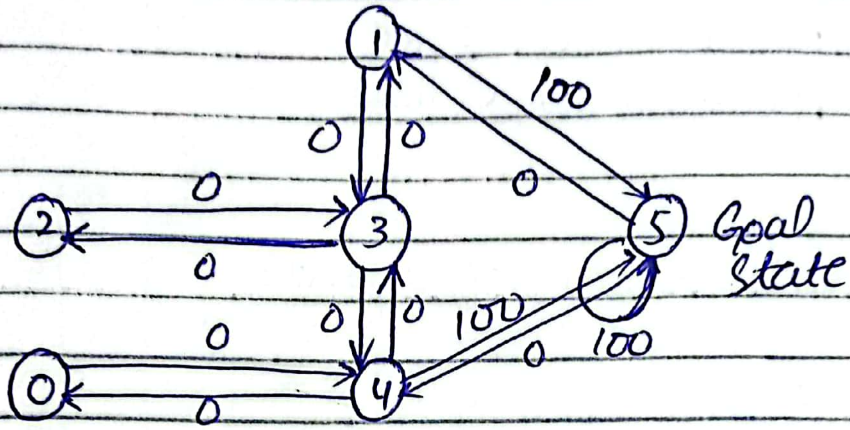
Reinforcement Learning:- Agent based learning.



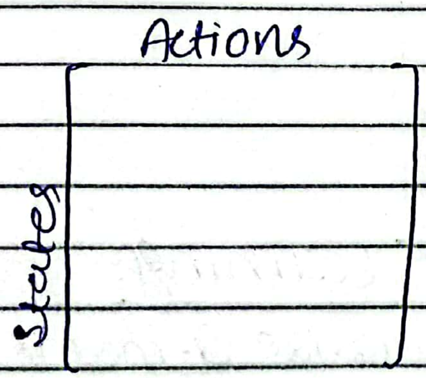
Enviroment → set of states.

States → environment
Actions → Agent



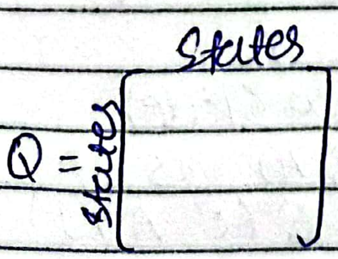


; Reward Matrix.



Rows = States.
Columns = Actions.

; Q- Matrix.



→ Square matrix.

Reward Matrix.

Actions →

States	0	1	2	3	4	5
0	-1	-1	-1	-1	0	-1
1	-1	-1	-1	0	-1	100
2	-1	-1	-1	0	-1	-1
3	-1	0	0	-1	0	-1
4	0	-1	-1	0	-1	100
5	-1	0	-1	-1	0	100

Q- matrix (Brain of agent)

states

states	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0

Temporal difference Learning:-

1. For each (S, A), initialize Q-matrix to zero
2. Observe the current state
3. DO
   - (i) Select action A & excute it.
   - (ii) Recieve a reward r.
   - (iii) Observe the S′ (new state).
   - (iv) Update Q-Matrix $Q(S,A)$ as

$$Q(S,A) = r + \gamma(\max[Q(S',A')])$$

$\gamma = 0.8$

Ep-1.

Let say, Agent is at 1 &
action is "5"

$S = 1, \; S' = 5$

$S = Q(1,5) = 100 + 0.8(\max[Q(5,1), Q(5,5), Q(5,4)]$

$= Q(1,5) = 100 + 0.8(\max[0,0,0])$.

$S = Q(1,5) = 100$

Q-Matrix.

	0	1	2	3	4	5
0	0	0	0	0	80	0
1	0	0	0	0	0	100
2	0	0	0	51	0	0
3	0	80	64	0	80	0
4	64	0	0	64	0	100
5	0	80	0	0	80	100

Ep-2. Ab ye updated matrix ko check krengy.
let say, Agent is at 3 &
action is "4".

$Q(3,4) = 0 + 0.8(\max[Q(4,0), Q(4,3), Q(4,5)])$.

$= 0 + 0.8(\max[0,0,0])$.

$Q(3,4) = 0$ update Q-matrix.

Ep-3.

~~S=4~~,

$Q(4,0) = 0 + 0.8 \times 0 = 0$.

Ep-4. $S = 4, \; S' = 5$.

$Q(4,5) = 100 + 0.8(Q(5,1), Q(5,5), Q(5,4))$.

$= 100 + 0.8(\max(0,0,0))$.

$Q(4,5) = 100$