Machine Intelligence

Artifical Intelligence :-

→ Simulation of human intelligence processes by machines.

Machine Learning :-

→ Type of artificial intelligence that allows software to become more accurate at predicting outcomes without explicitly being programmed.

Types of ML :-

1> Supervised learning : algorithm is trained on a labeled dataset, map input data to output

2> Unsupervised learning : algorithm is given an unlabeled data, tries to find patterns

3> Semi - supervised learning :

4> Reinforcement learning :

Four Categories Views of Intelligence :-

1> Thinking humanly
2> Acting humanly
3> Thinking rationally
4> Acting rationally

SURYA Gold
Date ______ Page ______

Intelligent Agent and its Types :-

→ Can be software, human or robots.

→ Main 4 rules all AI adhere to
- Perceive environment
- Observations used to make decisions
- Decisions take action
- Act rationally

→ PEAS (Characterisation of agents)
P = Performance Measure
E = Environment
A = Actuators
S = Sensor

→ Eg Vacuum cleaner

P: Ability to clean dirt, amount of dirt cleaned, power efficiency

E: Rooms of different sizes, dust of different types

A: Motor, tube, storage space, wheels/movement

S: Camera, dirt detection sensor

State :-

→ Configuration of an agent and its environment
→ Initial state

SURYA Gold
Date ______ Page ______

Actions :-

- Different choices / moves that an agent can make

Types of environment

1) Observability
2) Determinism (Opposite - Stochastic / Randomness)
3) Episodicity or Sequential
4) Dynamism
5) Continuity

Types of Intelligent Agents :-

1) Learning agent
2) Simple Reflex agent
3) Model-Based agent (uses previous history)
4) Goal based agent
5) Utility agent

Machine Learning :-

- Learning: from any process by which a system improves performance from experience.

— Study of algo that
- improve performance
- at some task
- from experience

SURYA Gold

Date______ Page______

Conjunction :-

→ $h(x) = c(x) = 1$
(hypothesis) (concept)

→ Find S (Algorithm)
1) Only consider positive sample
2) Starts specific hypothesis
3) Reduce from specific → general

Q) $H(x)$ = Null ^ Null ^ Null ^ Null ^ Null

= Many ^ big ^ No ^ Exp ^ One

= Many ^ ? ^ No ^ Exp ^ ?

= Many ^ ? ^ No ^ ~~Ex~~ ? ^ ?

→ Drawbacks:
- Attributes which are not binary get '?'
- Does not consider negative sample.

Performance Learning :-

→ Expected output list (Y)
Predicted output list ($\hat{Y}$)

→ Mean Square Error: $\frac{1}{n} \sum (Y - \hat{Y})^2$

→ Accuracy $A = 1 - E$

SURYA Gold

Date______ Page______

→ Type 1
Type 2

→ Performance metric:
- Accuracy
- Error

→ Confusion matrix:
1) True Positive
2) True Negative
3) False Positive
4) False Negative

		Predicted +	Predicted −
Actual	+	True Positive	False Negative (Type 2)
	−	False Positive (Type 1)	True Negative

→ Accuracy = $\frac{(TP + TN)}{TP + TN + FP + FN}$

→ Precision = $\frac{TP}{(TP + FP)}$ (How many correct +ve cases did we catch)

→ Recall = $\frac{TP}{TP + FN}$ (How many +ve cases did we catch from all the cases)
(Sensitivity)
(True Positive rate)

→ F1 score = Harmonic Mean (Recall, Precision)

$= \frac{2 \times Recall \times Precision}{Recall + Precision}$

→ Specificity = $\frac{TN}{TN + FP}$

→ *

	A	B	C
A	TA	FB	FC
B	FA	TB	FC
C	FA	FB	TC
	-		

Accuracy = $\frac{TA+TB+TC}{Total}$

$Recall_A = \frac{TA}{TA+FB+FC}$

→ Confusion matrix

	A	B	C
A	2	2	0
B	1	2	0
C	0	0	3

Accuracy = $\frac{2+2+3}{2+2+1+2+3} = \frac{7}{10} = 70\%$

$R_A = \frac{2}{2+2} = 1/2$ $\quad P_A = \frac{2}{3} =$

$R_B = \frac{2}{3}$ $\quad P_B = \frac{1}{2}$

$R_C = \frac{3}{3}$ $\quad P_C = \frac{3}{3}$

Avg R = 13/18 $\quad$ Avg P = 13/18

SURYA Gold
Date______ Page______

→ For class Labrador: Pred → Actual

True positive: P(L, L)
True Negative: P(H, H) + P(B,B) + P(B,H) + P(H,B)
False Positive: ~~P(H, L) + P(B, L)~~ P(L,H) + P(L,B)
False Negative: P(H, L) + P(B, L)

→ Kappa score, MCC score, Macro F1

$P_B = 9/91$ $\quad P_B = 9/10$

$$F1_B = \frac{2 \times 9/91 \times 9/10}{9/91 + 9/10} = \boxed{0.1782}$$

→ False Positive Rate = 1 − Specificity

$$= 1 - \frac{TN}{TN+FP}$$

$$= \boxed{\frac{FP}{TN+FP}}$$

→ Receiving Operating Characteristic: compare threshold based on TPR and FPR

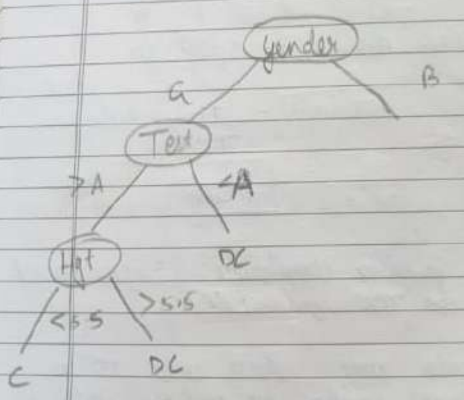
→ Area under curve: Area under TPR vs FPR curve signifies how good the model is.

SURYA Gold

Date ____ Page ____

Decision Trees Representation:-

G	H	Test Avg	Y(Binary)
G	<5.5	>A	C
G	>5.5	>A	DC
G	Nil	<A	DC
B			
B			
B			



Entropy:-

SURYA Gold

Date ____ Page ____

→ Entropy $= \frac{-m}{m+n}\log_2\left(\frac{m}{m+n}\right) - \frac{n}{m+n}\log_2\left(\frac{n}{m+n}\right)$

→ Information gain

$G(S, A) = \text{Entropy}(S) - I(A)$

$I(A) = \sum \frac{p_i+n_i}{p+n}\,\text{Entropy}(A)$

S → example

A → attribute

$E(S) = \frac{-p}{n+p}\log_2\left(\frac{p}{n+p}\right) - \frac{n}{n+p}\log_2\left(\frac{n}{n+p}\right)$

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

$= 0.94$

$E(\text{Outlook} = \text{Sunny}) = \frac{-2}{2+3}\log_2\left(\frac{2}{2+3}\right) \frac{-3}{3+2}\log_2\left(\frac{3}{3+2}\right)$

$= 0.971$

$E(\text{Outlook} = \text{Overcast}) = -1\log_2(+1) - 0\log_2(0)$

$= 0$

$E(\text{Outlook} = \text{Rainy}) = \frac{-3}{2+3}\log_2\left(\frac{+3}{2+3}\right) \frac{-2}{2+3}\log_2\left(\frac{2}{2+3}\right)$

$= 0.971$

$I(A) = \sum \frac{p_i+n_i}{p+n} = \frac{5}{14}(0.971) + 0 + \frac{5}{14}(0.971)$

$= \frac{10}{14} \times 0.971 = \boxed{0.693}$

$G(S,A) = 0.94 - 0.693$
$= \boxed{0.247}$

Q)

M	N	O	Y
A	C	Y	T
A	C	Z	E
A	D	X	T
A	D	Z	T
B	C	X	E
B	C	Z	F
B	D	X	F
B	D	Z	F

Ans. T = 3
E = 2
F = 3

$$E(S) = \frac{-3}{8}\log_{10}\left(\frac{3}{8}\right) \frac{-2}{8}\log_{10}\left(\frac{2}{8}\right) - \frac{3}{8}\log_{10}\left(\frac{3}{8}\right)$$

$= 0.0159 + 0.15 + 0.159$
$= \boxed{0.469}$

Decision Tree:

→ Determining decision tree:

1) Find Entropy (S)
2) Entropy (A)
Calculate Information (A)
Calculate gain (A)

3) Choose gain as the root node

Repeat

M	N	O	Y
A	C	X	T
A	C	Z	T
A	D	X	T
A	D	Z	T
B	C	X	T
B	C	Z	F
B	D	X	F
B	D	Z	F

$$E(S) = \frac{-5}{8}\log\left(\frac{5}{8}\right) - \frac{3}{8}\log_2\left(\frac{3}{8}\right)$$

$= 0.1275 + 0.1597 = \boxed{0.287}$

$I(A) = \sum \frac{[illegible]}{[illegible]} E(A)$

$E(M=A) = -1 \times \log(1) + 0 \times \log(0)$

$= \boxed{0}$

$E(M=B) = \frac{-1}{4}\log\left(\frac{1}{4}\right) - \frac{3}{4}\log\left(\frac{3}{4}\right)$

$= 0.15 + 0.093$

$= \boxed{0.24}$

$I(M) = \frac{4}{8} \times 0 + \frac{4}{8}(0.24)$

$= \boxed{0.12}$

$G(S, M) = 0.287 - 0.12$

$= \boxed{0.167}$

$E(N=C) = -\frac{3}{4}\log\left(\frac{3}{4}\right) - \frac{1}{4}\log\left(\frac{1}{4}\right)$

$= 0.24$

$E(N=D) = \cancel{I}\ \frac{-1}{2}\log\left(\frac{1}{2}\right) - \frac{1}{2}\log\left(\frac{1}{2}\right)$

$I(N) = \frac{1}{2} \times 0.24 + \frac{1}{2} \times 0.3$

$= \boxed{\cancel{0.62}}\quad 0.27$

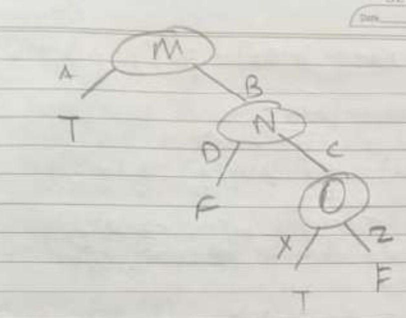
$G(S, N) = 0.017$

$E(O=X) = 0.24$

$E(O=Z) = 0.3$

$I(O) = 0.27$

$G(S, O) = 0.017$



$H = (A) \vee (B \wedge C \wedge X)$

Q)

Outlook	Temp	Humidity	Windy	Tennis
Sunny	High	High	W	N
Sunny	High	High	S	N
Overcast	High	High	W	Y
Rainy	Med	High	W	Y
Rainy	Cool	Normal	W	Y
Rainy ~~Overcast~~	Cool	Normal	S	N
Overcast ~~Sunny~~	Cool	Normal	S	Y
Sunny ~~Rainy~~	Med	High	W	N

$E(S) = \frac{4}{8}\log_2\left(\frac{4}{8}\right) + \frac{4}{8}\log_2\left(\frac{4}{8}\right)$

$= \boxed{1}$

$\cancel{E(O=S)}\ I(A) = \frac{3}{8}(0) + \frac{2}{8}(0) + \frac{3}{8}\left(0.389 + 0.528\right)$

$= \boxed{0.343}$

SURYA Gold

Date ______ Page ______

$$I(T) = \frac{3}{8}(0.918) + \frac{2}{8}(1) + \frac{3}{8}(0.918)$$

$$= \boxed{0.9335}$$

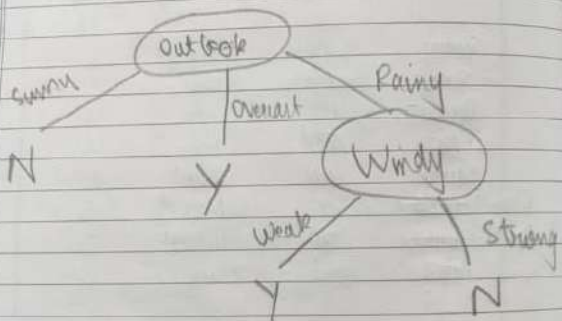
$$I(H) = \frac{5}{8}\left(\underbrace{\frac{3}{5}\log\left(\frac{3}{5}\right)}_{0.4421} + \underbrace{\frac{2}{5}\log\left(\frac{2}{5}\right)}_{0.5287}\right)$$

$$+ \frac{3}{8}\left(\underbrace{\frac{1}{3}\log\left(\frac{1}{3}\right)}_{0.528} + \underbrace{\frac{2}{3}\log\left(\frac{2}{3}\right)}_{0.389}\right)$$

$$= 0.6 + 0.343$$

$$= \boxed{0.943}$$

$$I(W) = \frac{5}{8}\left(\boxed{0.943}\right.$$



Decision Boundary :-

- Linearly seperable
- In n dimensions, boundary is (n-1) hyperplane.

- Non-linearly seperable

SURYA Gold

Date ______ Page ______

Bias -

→ Inability of the model measured by some difference or error occurring b/w model's prediction and an actual value

- Difference b/w actual and predicted values is known as error / bias error

$$Bias(Y) = E(Y') - Y$$

Variance :-

→ measure of spread of data across the mean.

$$Variance = E[(Y' - E(Y))^2]$$

$$E[(\hat{a}_0 - a_0)^2] = Variance + Bias^2 + Noise$$

Overfitting :-

→ Given a hypothesis space H, a hypothesis h ∈ H is said to overfit the training data if there exists some alternate hypothesis h', such that it has a smaller error data on the testing data space.

→ To avoid overfitting in decision tree: pruning the decision tree.

- Post pruning
- Rule pruning

SURYA Gold

Date ____ Page ____

K Nearest Neighbours :-

- Instance based learning stores training examples and delays processing until a new instance must be classified, that is, lazy evaluation.

→ Inductive bias: Set of assumption that the learner uses to predict output.

1) All instances correspond to points in the n-D space.
2) Points may belong to some class (classification) or some real value.
3) Given a query point find where it belongs in space.
4) Find K Nearest Neighbours.
5) Assign mode for classification and mean for regression.

$$D(x, y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$$

$p = 1$: Manhattan distance
$p = 2$: Euclidean distance

- Can start with $K = n+1$ (Number of class + 1)

- ~~Error~~ $e = 1$ when actual != predicted

$$Error(E) = \frac{1}{n} \sum e$$

SURYA Gold

Date ____ Page ____

$$MSE = \frac{1}{n} \sum (y_i - y)^2$$

Q)

	BP	Sugar	Harmo	WBC	Risk
A	100	120	12	6	No
B	110	130	14	5	Yes
C	120	110	11	7	Yes
D	100	140	13	7	No
E	115	140	11	6	Yes

$K = 3$

$x_q = 100, 135, 12, 8$

$x_{aq} = 15.13$ (No)
$x_{bq} = 11.74$ (Yes)
$x_{cq} = 25.03$
$x_{dq} = 5.196$ (No)
$x_{eq} = 18.96$

} No

Weighted KNN:-

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{K} w_i \, \delta(v, f(x_i))$$

$$w_i = \frac{1}{dist(x_q, x_i)^m}$$

Linear Regression :-

→ $Y = mx + b$

→ $$MSE = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - (mx_i + b) \right)^2$$

$$\frac{\partial}{\partial m} MSE = -\frac{2}{n} \sum_{i=1}^{n} \left(y_i - (m x_i + b) \right) \times x_i$$

$$\frac{\partial}{\partial b} MSE = -\frac{2}{n} \sum_{i=1}^{n} \left(y_i - (m x_i + b) \right) \times 1$$

→ training rate = α / L : (0, 1)
= 0.001 / 0.01

$$m = m - L \frac{\partial MSE}{\partial m}$$

$$b = b - L \frac{\partial MSE}{\partial b}$$

Q)

x	y	$\hat{y}$
10	5	60
20	7	100
30	9	140
40	11	180

$m = 4$
$b = 20$
$L = 0.001$

MSE = $\boxed{14349}$

$m = 4 + 0.001 \left[-\frac{2}{4} \sum (\dots) \right]$

$m = -651 \cdot 2.5$

$b = 20 - 0.001 \left[-\frac{2}{4} (\dots) \right]$

$= 19.776$

Logistic Regression :-

1) Setting up the problem : two possibilities : either spam (1) or not spam (0).

2) Basic Idea : Way to find a way to express the relationship b/w features and probability.

3) Sigmoid Function : Takes any fn input and squashes it between 0 and 1

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

$$\frac{dy}{dz} = \frac{-1}{(1+e^{-z})^2} \times e^{-z} \times -1 = \frac{e^{-z} + 1 - 1}{(1+e^{-z})^2}$$

$$= \boxed{\frac{1}{(1+e^{-z})} \left[1 - \frac{1}{(1+e^{-z})} \right]}$$

$$\frac{dy}{dz} = y(1-y)$$

→ Linear : $y = mx + b$
→ Logistic : $y = \frac{1}{1 + e^{-(mx+b)}}$

→ Log likelihood $= -y \log(\hat{y}) - (1-y) \log(1-\hat{y})$

4) Linear Combination : Linear relationship between features and log-odds.

5) Training the model : Adjusts the weights

SURYA Gold

Date ____ Page ____

$$\frac{\partial}{\partial m} MSE = -\frac{2}{n}\sum_{i=1}^{n}\left(y_i - (mx_i + b)\right) \times x_i$$

$$\frac{\partial}{\partial B} MSE = -\frac{2}{n}\sum_{i=1}^{n}\left(y_i - (mx_i + b)\right) \times 1$$

→ training rate = α / L = (0, 1)
= 0.001 / 0.01

$$m = m - L\frac{\partial MSE}{\partial M}$$

$$b = b - L\frac{\partial MSE}{\partial b}$$

Q)

x	y	$\hat{y}$	
10	5	60	m = 4
20	7	100	b = 20
30	9	140	L = 0.001
40	11	180	

MSE = $\boxed{14349}$

$$m = 4 + 0.001\left[-\frac{2}{4}\sum(\ldots)\right]$$

m = ~~-651~~ -2.5

$$b = 20 - 0.001\left[-\frac{2}{4}(\ldots)\right]$$

= 19.776

Logistic Regression :-

SURYA Gold

Date ____ Page ____

1) Setting up the problem: two possibilities: either spam (1) or not spam (0).

2) Basic Idea: Way to find a way to express the relationship b/w features and probability.

3) Sigmoid Function: Takes any f^n input and squashes it between 0 and 1.

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

$$\frac{dy}{dz} = \frac{-1}{(1+e^{-z})^2} \times e^{-z} \times -1 = \frac{e^{-z}+1-1}{(1+e^{-z})^2}$$

$$= \boxed{\frac{1}{(1+e^{-z})}\left[1 - \frac{1}{(1+e^{-z})}\right]}$$

$$\frac{dy}{dz} = y(1-y)$$

- Linear : $y = mx + b$
- Logistic : $y = \frac{1}{1 + e^{-(mx+b)}}$

- Log likelihood = $-y\log(\hat{y}) - (1-y)\log(1-\hat{y})$

4) Linear Combination: Linear relationship between features and log-odds.

5) Training the model: Adjusts the weights

assigned to each feature

$$y = \frac{1}{1-e^{-z}} \qquad z = mx+b$$

$$\hat{y} = \frac{1}{1+e^{-(mx+b)}}$$

$$\text{Error}(y-\hat{y}) = \underbrace{-y\log(\hat{y})}_{A} - \underbrace{(1-y)\log(1-\hat{y})}_{B}$$

$$\frac{\partial A}{\partial m} = \frac{\partial A}{\partial z} \times \frac{\partial z}{\partial m}$$

$$= \frac{\partial(-y(\log(\hat{y})))}{\partial m} = \frac{-y}{\hat{y}} \frac{\partial \hat{y}}{\partial z} \times \frac{\partial z}{\partial m}$$

$$= \frac{-y}{\hat{y}}\left(\hat{y}(1-\hat{y})\right) \times x$$

$$= -y(1-\hat{y})x \quad — ①$$

$$\frac{\partial B}{\partial m} = \frac{\partial B}{\partial z} \times \frac{\partial z}{\partial m}$$

$$= \frac{\partial(-(1-y)\log(1-\hat{y}))}{\partial z} \times \frac{\partial z}{\partial m}$$

$$= \frac{-(1-y)}{1-\hat{y}} \times \frac{-\partial \hat{y}}{\partial z} \times \frac{\partial z}{\partial m}$$

$$= \frac{-(1-y)}{1-\hat{y}} \times -(\hat{y}(1-\hat{y})) \times x$$

$$= (1-y) \times \hat{y} \quad — ②$$

$$\frac{\partial L}{\partial m} = ① + ②$$

$$= -y(1-\hat{y})x + x\hat{y}(1-y)$$

$$= -yx + \cancel{xy\hat{y}} + x\hat{y} - \cancel{xy\hat{y}}$$

$$= \boxed{-x(y-\hat{y})}$$

$$\frac{\partial A}{\partial b} = \frac{\partial A}{\partial z} \times \frac{\partial z}{\partial b}$$

$$= \frac{-y}{\hat{y}} \times \hat{y}(1-\hat{y}) \times 1$$

$$= -y(1-\hat{y}) \quad — ③$$

$$\frac{\partial B}{\partial b} = \frac{\partial B}{\partial z} \times \frac{\partial z}{\partial b}$$

$$= (1-y)\hat{y} \times 1 \quad — ④$$

$$\frac{\partial L}{\partial b} = ③ + ④$$

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

$$= -y + \cancel{\hat{y}y} + \hat{y} - \cancel{y\hat{y}}$$

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

$$\boxed{\begin{aligned} m &= m - \eta \frac{\partial L}{\partial m} \\ b &= b - \eta \frac{\partial L}{\partial b} \end{aligned}}$$

Generative and Discriminative Model :-

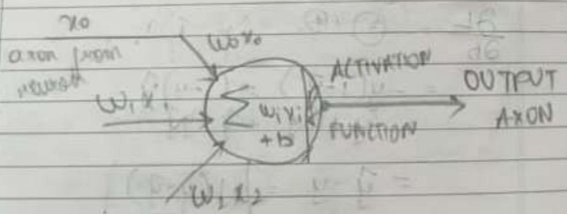
- Discriminative model : predictions based on conditional probability.
- Generative model :

- Generative model : Joint probability
Discriminative model : Conditional probability.

Artificial Neural Network :-

- Algorithm based on brain function.

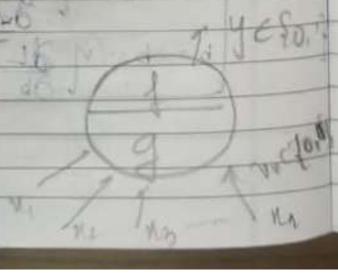
- Neural network learning provide approach to approximating real valued, discrete valued and vector valued.



- Input $\longrightarrow$ Hidden layer $\longrightarrow$ Output

McCulloch pitts neuron :-

$\rightarrow g(x_1, x_2 \ldots x_n)$
$= g(x)$
$= \sum_{i=1} x_i$



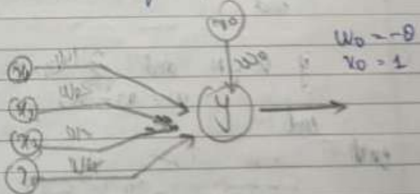
- Inputs can be excitatory or inhibitory

$$y = f(g(x)) = 1 \text{ if } g(x) > \theta$$
$$= 0 \text{ if } g(x) < \theta$$
Threshold

- MP neuron can be used to represent Boolean function which are linearly seperable.

Perceptron :-

- Introduction of numerical weights for input.
- Inputs are no longer limited to boolean values.



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

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

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

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

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

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

Perceptron learning algorithm:-

P ← inputs with label 1;
N ← inputs with label 0;

Initialise w ~~with~~ randomly:

while ! convergence do
    Pick a random $x \in P \cup N$;
    if $x \in P$ and $\sum_{i=0}^{n} w_i \times x_i < 0$ then
        $w = w + x$;
    end

    if $x \in N$ and $\sum_{i=0}^{n} w_i \times x_i \geq 0$ then
        $w = w - x$;
    end
end

$\rightarrow w = [w_0 \; w_1 \; w_2 \; \dots \; w_n]$
$x = [x_0 \; x_1 \; \dots \; x_n]$

$\rightarrow$ line $= w^T x = 0$
if $w^T x \geq 0 \Rightarrow y(x) = 1$
$w^T x < 0 \Rightarrow y(x) = 0$

$\Rightarrow w \perp x$

$\cos \alpha = \dfrac{w^T x}{|w| \, |x|}$

$\rightarrow w_{new} = w + x$
$\cos \alpha_{new} = w_{new}^T x$
$= (w + x)^T x$
$= w^T x + x^T x$
$\cos \alpha_{new} = \cos \alpha + x^T x$

As $\cos(\alpha_{new}) > \cos(\alpha)$
$\Rightarrow \alpha_{new} < \alpha$

Q) Four points $x_1 = (0, 1)$, $x_2 = (-1, -1)$, $x_3 = (2, 3)$, $x_4 = (4, -5)$. Labels are $-1, 1, -1, 1$. Initialise $w_0 = [0, 0]$.

Ans $w_0 x_1 = 0 \Rightarrow$ Positive.

$w_0 = [0 - 0, 0 - 1] = [0, -1]$

$w_0 x_1 = (0) + (-1) = -1 < 0 \Rightarrow$ Negative

$w_0 x_2 = [0(-1) + -1(-1)] = 1 =$ Positive

$w_0 x_3 = [0(2) + -1(3)] = -3 =$ Negative

$w_0 x_4 = [0(4) + -1(-5)] = 5 =$ Positive

SURYA Gold

— single perceptron fails when:
. data is not linearly seperable.

— gradient descent:

$\theta = [w, b]$
$\Delta\theta = [\Delta w, \Delta b]$

$\theta_{new} = \theta + \eta\Delta\theta$

— Let $\Delta\theta = u$, then from Taylor series,

$\mathcal{L}(\theta + \eta u) = \mathcal{L}(\theta) + \eta * u^T \nabla_\theta \mathcal{L}(\theta)$
$+ \frac{\eta^2}{2!} u^T \nabla^2 \mathcal{L}(\theta) u + \ldots$

$\approx \mathcal{L}(\theta) + \eta * u^T \nabla_\theta \mathcal{L}(\theta)$

$\Rightarrow \mathcal{L}(\theta + \eta\mu) - \mathcal{L}(\theta) = \eta u^T . \nabla^2 \mathcal{L}(\theta)$

→ To make it favourable,
$\mathcal{L}(\theta + \eta u) - \mathcal{L}(\theta) < 0$

$\Rightarrow u^T \nabla_\theta \mathcal{L}(\theta) < 0$

Let β be angle b/w u and $\nabla_\theta \mathcal{L}(\theta)$

$\Rightarrow \cos(\beta) = \frac{u^T \nabla_\theta \mathcal{L}(\theta)}{||u|| * ||\nabla_\theta \mathcal{L}(\theta)||}$

$\Rightarrow -1 \leq \frac{u^T \nabla_\theta \mathcal{L}(\theta)}{||u|| \times ||\nabla_\theta \mathcal{L}(\theta)||} \leq 1$

SURYA Gold

Let $k = ||u|| * ||\nabla_\theta \mathcal{L}(\theta)||$

$-k \leq k\cos(\beta) = u^T \nabla_\theta \mathcal{L}(\theta) \leq k$

→ $\cos(\beta)$ will be most negative when $\beta = 180$.

⇒ The direction u should move is supposed to be at 180°.

→ $w_{t+1} = w_t - \eta\Delta w_t$
$b_{t+1} = b_t - \eta\Delta b_t$

→ $\frac{\partial E}{\partial w_i} = \frac{\partial}{\partial w_i} \frac{1}{2} \sum_d (t_d - o_d)^2$

$= \frac{1}{2} \sum \frac{\partial}{\partial w_i} (t_d - o_d)^2$

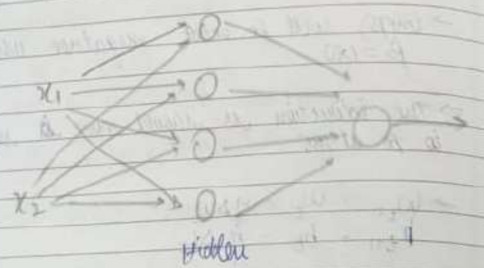
$= \frac{1}{2} \sum 2 (t_d - o_d) \frac{\partial}{\partial w_i} (t_d - o_d)$

$= \sum (t_d - o_d) \frac{\partial}{\partial w_i} (t_d - \vec{w} \cdot \vec{x}_d)$

$\frac{\partial E}{\partial w_i} = \sum (t_d - o_d)(-x_{i,d})$

$\boxed{\Delta w_i = \eta \sum (t_d - o_d) x_{id}}$

Boolean Function using network of perceptron :-



→ Bias of each perceptron is -2.
⇒ Threshold is 2.

Q) $f(x) = x^3 - 3x^2 + 2$. What is updated value of x after 3rd gradient descent. $x = 4, \eta = 0.01$.

Ans
$$y = x^3 - 3x^2 + 2$$
$$\frac{dy}{dx} = 3x^2 - 6x$$

$$x = x - \eta \frac{dy}{dx}$$

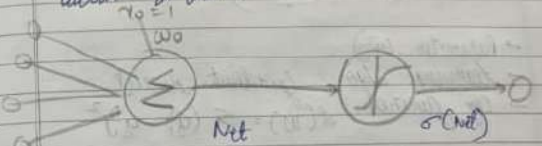
$$x = 4 - 0.01(24)$$
$$= 4 - 0.24$$
$$= 3.76$$

Multilayer :-

→ Input layer is where we feed the input.
→ Hidden layer
→ Output layer

Activation Function :-

→ Non-linear transformation that we apply on our input before propagating.
— Decide whether a neuron should be active or not.



→ Rectified Linear Unit (ReLU)

$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

→ Leaky ReLU : $f(x) = \begin{cases} y, & y > 0 \\ ay, & y \leq 0 \end{cases}$

→ Softmax : $\dfrac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$

— Parameters of ML are
· data
· model, itself

- learning algorithm
- objective function

→ eg: Machine learning setup for movie example

→ Data : $\{x_i\}$ = movie, y_i = like / dislike $\}_{i=1}^{n}$
→ Model : Our approximation of relation

$$\hat{y} = \left(\frac{1}{1 + e^{-(w^T x)}} \right)$$

→ Parameter : (w)
→ Learning algo : gradient descent.
→ Obj function : $\mathcal{L}(w) = \sum_i (\hat{y}_i - y_i)^2$

Forward Propagation :-

→ Consider :
2 inputs x_1 and x_2
4 hidden layer neurons (1 hidden layer)
1 Output neuron.

→ Initialise weight matrix :
→ Dimension of weight matrix :

No of units in prev layer
×
No of units in current layer.

→ Let the wt and bias b/w input and hidden layer be represented as W_{hx} and b_h respectively.

~~Z_1 = to h_{xx}~~

$$Z_1 = X W_{hx} + b_h$$

→ This is passed through activation f^n.

$$a_1 = \sigma(z_1).$$

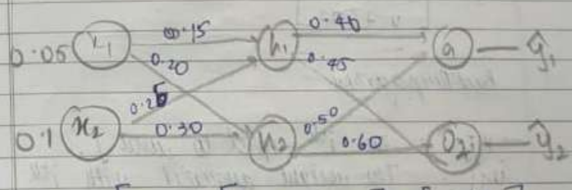
→ This is passed onto output layer

$$z_2 = a_1 W_{yh} + b_y$$

→ Final output

$$\hat{y} = \sigma(z_2)$$

Q) Apply forward propagation having input x_1, x_2 and layers h_1, $b_0 = 0.35$.



$$Z_1 = \begin{bmatrix} 0.05 \\ 0.1 \end{bmatrix} \cdot \begin{bmatrix} 0.15 & 0.20 \\ 0.25 & 0.30 \end{bmatrix} + \begin{bmatrix} 0.35 \\ 0.35 \end{bmatrix}$$

$$= \underset{1\times2}{\begin{bmatrix} 0.05 & 0.1 \end{bmatrix}} \underset{2\times2}{\begin{bmatrix} 0.15 & 0.20 \\ 0.25 & 0.30 \end{bmatrix}} + \begin{bmatrix} 0.35 \\ 0.35 \end{bmatrix}$$

$$= \begin{bmatrix} 0.0325 \\ 0.04 \end{bmatrix} + \begin{bmatrix} 0.35 \\ 0.35 \end{bmatrix}$$

$$= \begin{bmatrix} 0.3825 \\ 0.39 \end{bmatrix}$$

SURYA Gold

$$a_1 = \sigma(z_1) = \begin{bmatrix} \frac{1}{1+e^{-0.3825}} \\ \frac{1}{1+e^{-0.39}} \end{bmatrix} = \begin{bmatrix} 0.5944 \\ 0.5963 \end{bmatrix}$$

$$z_2 = a_1 \cdot w_y$$

$$= [0.5944, 0.5963] \begin{bmatrix} 0.40 & 0.45 \\ 0.50 & 0.55 \end{bmatrix}$$

$$= \begin{bmatrix} 0.5359 \\ 0.5954 \end{bmatrix} + \begin{bmatrix} 0.6 \\ 0.6 \end{bmatrix}$$

$$= \begin{bmatrix} 1.1359 \\ 1.1954 \end{bmatrix}$$

$$\hat{y} = \begin{bmatrix} 0.7069 \\ 0.7672 \end{bmatrix}$$

Backpropagation :-

- x_{ji} : The i^{th} input to unit j.
- w_{ji} : The weight associated with i^{th} input to j unit.
- net_j = the weighted sum $\sum w_{ji} x_{ji}$
- o_j = output of unit j.
- t_j = target output for unit j,
- σ : sigmoid f^n
- Downstream (j): Nodes whose immediate input is output of j.

SURYA Gold

$$\rightarrow w_{ji\,new} = w_{ji\,old} - \eta \Delta w_{ji}$$

$$\Delta w_{ji} = \frac{\partial E_d}{\partial w_{ji}}$$

$$= \frac{\partial E_d}{\partial net_j} \times \frac{\partial net_j}{\partial w_{ji}}$$

$$net_j = \frac{\partial w_{ji} \times x_{ji}}{\partial w_{ji}} = x_{ji}$$

$$\Delta w_{ji} = \frac{\partial E_d}{\partial net_j} \times x_{ji}$$

$$= \frac{\partial E_d}{\partial o_j} \times \frac{\partial o_j}{\partial net_j} \times x_{ji}$$

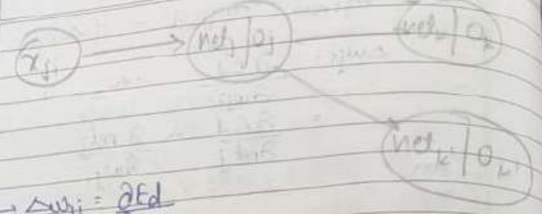
$$= \frac{\partial E_d}{\partial o_j} \times o_j(1-o_j) x_{ji}$$

$$= \frac{\partial \frac{1}{2} \sum_{k \in output} (t_k - o_k)^2}{\partial o_j} \times o_j(1-o_j) \times x_{ji}$$

$$= \frac{1}{2} \times 2(t_j - o_j) \times (-1) \times o_j(1-o_j) \times x_{ji}$$

$$\frac{\partial E_d}{\partial w_{ji}} = -(t_j - o_j)\, o_j(1-o_j)\, x_{ji}$$

$$\boxed{w_{ji\,new} = w_{ji\,old} + \eta \underbrace{(t_j - o_j)\, o_j(1-o_j)}_{\delta_j} x_{ji}}$$



$\rightarrow \Delta w_{ji} = \dfrac{\partial E_d}{\partial w_{ji}}$

$\rightarrow \dfrac{\partial E_d}{\partial w_{ji}} = \dfrac{\partial E_d}{\partial net_j} \times \dfrac{\partial net_j}{\partial w_{ji}}$

$= \dfrac{\partial E_d}{\partial net_j} \times x_{ji}$

$\dfrac{\partial E_d}{\partial net_j} = \sum\limits_{k \in Down(j)} -\delta_k \dfrac{\partial net_k}{\partial net_j}$

$= -\sum \delta_k \dfrac{\partial net_k}{\partial o_j} \times \dfrac{\partial o_j}{\partial net_j}$

$= -\sum \delta_k \dfrac{\partial net_k}{\partial o_j} \times o_j(1-o_j)$

$= -\sum\limits_{k \in (Dj)} \delta_k w_{kj}\, o_j (1-o_j)$

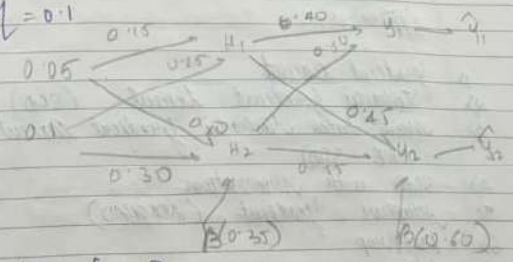
$\dfrac{\partial E_d}{\partial w_{ji}} = -\sum\limits_{k \in Down(j)} \delta_k w_{kj}\, o_j (1-o_j) \cdot x_{ji}$

$\Delta w_{ji} = x_{ji}\, o_j (1-o_j) \sum\limits_{k \in Down(j)} \delta_k w_{kj}$

$\boxed{w_{ji\,new} = w_{ji\,old} + \eta\, x_{ji}\, o_j (1-o_j) \sum \delta_k w_{kj}}$

Q) For the same question, update weights.

Ans $\eta = 0.1$



Actual $\begin{bmatrix} 0.1 \\ 0.99 \end{bmatrix}$

$w_5' = w_5 + (0.1)(0.01 - 0.7569)(0.7569)(1 - 0.7569)(0.5944)$

$w_5' = 0.40 - 0.00816$
$= \boxed{0.39184}$

$w_6' = 0.45 + (0.1)(0.99 - 0.7677)(0.7677)(1 - 0.7677)(0.5963)$

$= 0.45 + 0.00296$
$= \boxed{0.4523}$

$w_7' = 0.50 + (0.1)(0.01 - 0.7569)(0.7569)(1 - 0.7569)(0.5944)$

$= 0.50 - 0.00816$
$= \boxed{0.49184}$

$w_8' = 0.55 + 0.00236$
$= \boxed{0.55236}$

$\Delta w_1 = \eta\, O_j (1 - O_j)\, x_{ji} \sum (t_k - O_k) O_k (1 - O_k) w_{kj}$

Optimizers :-

1) Gradient Descent
2) Stochastic Gradient Descent (SGD)
3) Mini Batch Stochastic Gradient Descent (MB - SGD)
4) SGD with momentum
5) Adaptive Gradient (ADA GRAD)
6) RMS Prop
7) Adam

SGD with momentum :-

$$V_t = \gamma V_{t-1} + \eta \nabla w_t$$

$$V_0 = 0$$

$$V_1 = \gamma V_0 + \eta \nabla w_1$$

$$V_2 = \gamma V_1 + \eta \nabla w_2$$
$$= \gamma(\gamma V_0 + \eta \nabla w_1) + \eta \nabla w_2$$
$$= \gamma^2 V_0 + \gamma \eta \nabla w_1 + \eta \nabla w_2$$

$$V_3 = \eta \nabla w_3 + \gamma \eta \nabla w_2 + \gamma^2 \eta \nabla w_1 + \gamma^3 V_0$$

Adaptive Gradient :-

→ Key idea of Ada Grad is to have an adaptive learning rate for each of the weights.

gradient

→ $s_t \leftarrow s_{t-1} + g_t \odot g_t$

→ $w_t \leftarrow w_{t-1} - \dfrac{\eta_0}{\sqrt{s_t + \epsilon}} \odot g_t$

→ Adaptive learning rate wrt gradient.

→ Main weakness is accumulation of squared gradient in the denominator.

→ Changes learning rate aggressively.

→ Very old and stale data continue to affect learning rate

RMS Prop :-

→ Root mean square propagation

→ $s_t \leftarrow \gamma s_{t-1} + (1-\gamma)\, g_t \odot g_t$

$$s_t = (1-\gamma)\, g_t \odot g_t + \gamma s_{t-1}$$
$$= (1-\gamma)\, g_t \odot g_t + (1-\gamma)\gamma\, g_{t-1} \odot g_{t-1} + \gamma^2 s_{t-1}$$

Adam (Adaptive Moment Estimation) :-

→ ~~Maintain~~ $V_t = \beta_1 * V_{t-1} + (1-\beta_1) * g_t$

$$s_t = \beta_2 * s_{t-1} + (1-\beta_2) * g_t^2$$

→ $\Delta w = \dfrac{-\eta\, V_t}{\sqrt{s_t + \epsilon}} * g_t$

Deep Learning :-

→ Form of ML that enables computers to learn from experience and understand the world

Bias and Variance :-

- High variance / low bias : High model complexity
- → Low variance / high bias : Low model complexity

→ Ideally Low variance, low bias

Regularization :-

→ L1 and L2 regularization

L1 $\|w\|_1 = |w_1| + |w_2| + \dots |w_n|$ (LASSO)

L2 : $\|w\|_2 = w_1^2 + w_2^2 + \dots w_n^2$ (Ridge)

→ Loss = Error $(y, \hat{y})$

→ Loss function with L1 :
$$\text{Loss} = \text{Error}(y, \hat{y}) + \lambda \sum_{i=1}^{N} |w_i|$$

→ Loss function with L2 :
$$\text{Loss} = \text{Error}(y, \hat{y}) + \lambda \sum_{i=1}^{N} w_i^2$$

SURYA Gold
Date ______ Page ______

L :
$$w_{new} = w - \eta \frac{dL}{dw}$$
$$= w - \eta [2x(wx+b-y)]$$

L1 :
$$w_{new} = w - \eta \frac{\partial L_1}{\partial w}$$
$$= w - \eta \left[2x(wx+b-y) + \lambda \frac{d|w|}{dw} \right]$$
$$= \begin{cases} w - \eta [2x(wx+b-y) + \lambda] & w > 0 \\ w - \eta [2x(wx+b-y) - \lambda] & w < 0 \end{cases}$$

L2 :
$$w_{new} = w - \eta \frac{\partial L_2}{\partial w}$$
$$= w - \eta [2x(x+b-y) + 2\lambda w]$$

→ Data augmentation
→ Early stopping
→ Dropout

Convolution Neural Network :-

→ CNN typically includes two operations, which can be thought of as feature extractors : convolution and pooling.

→ Why not ANN :
- Images are too big
- Positions can change

→ Element wise multiplication happens with filter.

→ It is not feature reduction, it is feature mapping.

→ Apply convolution on the given matrix

$$M = \begin{bmatrix} 2 & 2 & 1 \\ 3 & 1 & -1 \\ 4 & 3 & 2 \end{bmatrix} \qquad F = \begin{bmatrix} -1 & 1 \\ -1 & 0 \end{bmatrix}$$

Stride = 1
Bias = 2

$$A = \begin{bmatrix} -3 & -2 \\ -6 & -5 \end{bmatrix} \Rightarrow A+B = \begin{bmatrix} -1 & 0 \\ -4 & -3 \end{bmatrix}$$

→ Stride: governs how many steps taken by filter.

→ After applying filter f, for an input image n, output size:

$I : (n \times n)$

$F : (f \times f)$ Stride = 1

$O : (n - f + 1) \times (n - f + 1)$

→ Image: (8×8)
Filter: (3×3)
Output: (6×6)

→ Padding p

Image: $(n+2p) \times (n+2p)$

Filter: $f \times f$

Output: $(n+2p-f+1) \times (n+2p-f+1)$

SURYA Gold

Date ______ Page ______

→ Stride s : $\left(\frac{n+2p-f}{s} + 1\right) \times \left(\frac{n+2p-f}{s} + 1\right)$

→ Pooling: feature reduction.

→ Five different layers in CNN:

- Input Layer
- Convo Layer (Convo + Relu)
- Pooling layer
-
-
-

→ Parameter sharing: A feature detector that is useful in one part of image is probably useful in another part.

→ Sparsity of connections: Convnets are able to have relatively few parameters.

→ $W_2 = W_1 - F + 2P + 1$
$H_2 = H_1 - F + 2P + 1$

Since $W_2 = W_1$ {Input width = Output width}

$\Rightarrow F = 2P + 1$

$P = \frac{F-1}{2}$

Q) Increasing the learning rate reduces the bias or reducing variance.

→ Calculating number of parameters :-

→ w_c = Number of weights of the Convolution layer.
→ B_c = Number base of conv layer
→ P_c = Number of parameter of conv
→ K = size(width) of Kernels
→ N = No of Kernels
→ C = No of channels

$W_c = K^2 \times C \times N$
$B_c = N$
$P_c = W_c + B_c$

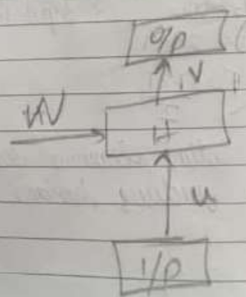
→ Batch normalisation : Between layers of Neural Networks.

→ $z = g(w, x)$

$z^N = \left(\frac{z - \mu_z}{s_z}\right)\gamma + \beta$

$a = f(z^N)$

__Recurrent Neural Network :-__



→ $V_{new} = V_{old} - \eta \frac{\partial L}{\partial V}$

$\frac{\partial L}{\partial V} = \sum_{t=0}^{T-1} \frac{\partial L_t}{\partial V}$

$\frac{\partial L_t}{\partial V} =$ $\quad Z_t = H_0 V_t$

$y_t = \text{sigmoid}(z_t)$

Loss f^n = Cross entropy

$= -y_t \log(\hat{y}_t) - (1-y_t)\log(1-\hat{y}_t)$

$\frac{\partial L_0}{\partial V} = \frac{\partial L}{\partial \hat{y}_t} \times \frac{\partial \hat{y}_t}{\partial z_t} \times \frac{\partial z_t}{\partial V}$

$= \frac{\hat{y}_t - y_t}{\hat{y}_t(1-\hat{y}_t)} \times y_t(1-y_t) \times h_0$

$\frac{\partial L_0}{\partial V} = (\hat{y}_t - y_t) h_0$

$\frac{\partial L}{\partial V} = \sum_{t=0}^{T-1} \frac{\partial L_T}{\partial V}$

$= \boxed{\sum_{t=0}^{T-1} (\hat{y} - y)_t \, h_t}$

→ For W, $\frac{\partial L_0}{\partial w} = \frac{\partial L_0}{\partial \hat{y}_t} \times \frac{\partial \hat{y}_t}{\partial h_0} \times \frac{\partial h_0}{\partial w} \times \frac{\partial h_k}{\partial w}$

$$\boxed{\frac{\partial L_j}{\partial w} = \sum_{k=0}^{j} \frac{\partial L_j}{\partial y_j} \times \frac{\partial y_j}{\partial h_j} \left(\prod_{m=k+1} \frac{\partial h_m}{\partial h_{m-1}} \right)}$$

SURYA Gold

Date ______ Page ______

$$\frac{\partial L}{\partial W} = \sum_{j=0}^{t+1} \left[\sum_{k=0}^{j} (\hat{y}_j - y_j) \prod_{m=k+1}^{j} W^T \cdot \right.$$

$$\left. \text{diag}\left[1 - \tanh^2 (W h_{m-1} + U x_m)\right] \odot h_{k-1} \right]$$

$$\frac{\partial}{\partial x} \tanh(x) = 1 - \tanh^2(x)$$

→ Represent the equation for $\frac{\partial L}{\partial W}$ / $\frac{\partial L}{\partial U}$

$\frac{\partial L}{\partial U}$ is same eq^n as $\frac{\partial L}{\partial W}$, where h_{k-1} is replaced by x_j

<u>Long-Short Term Memory (LSTM):-</u>

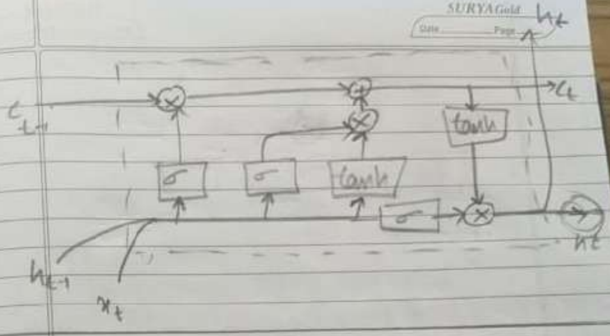
→ Forget gate
→ Input gate
→ Output gate

Q> A seller has a profit of 0 at day 1, at day 2, he has 0.25, 3 - 0.5, 4 - 0, 5 - ?

Q> At day 4, STM = -0.2, LTM = -0.3

SURYA Gold

Date ______ Page ______



→ Forget gate:

$$C_{t-1} = C_{t-1} \times sig(x_i U + h_{i-1} W + b)$$

→ Input gate:

$$C_t = C_{t-1} + T$$

$$T = sig(x_i U + h_{i-1} W + b) \times \tanh(x_i U + h_{i-1} W + b)$$

→ Output gate:

$$h_t = \tanh(C_t) \times sig(x_i U + h_{i-1} W + b)$$