Machine Learning (Module-1)

Machine Learning is the science of making computers learn and act like human by feeding data and information without being explicitly programmed.

Machine Learning

- Supervised (Labelled data)
  - Classification (With 2 or more classes (yes/no), (true/false)).
  - Regression (Output: Real or continous value).
    Ex:- Salary based on experience
    How much salary raise I will get?
- Unsupervised

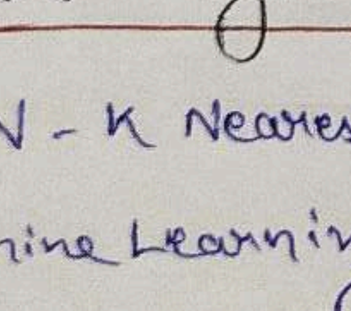
Will I get a salary raise or not.

. KNN - Algorithm ?

KNN - K Nearest Neighbors, is one of the simplest supervised Machine Learning algorithm most used for classification.

It classifies a data point based on how its neighbors are classified.

For K = 3 (Nearet neighbour)



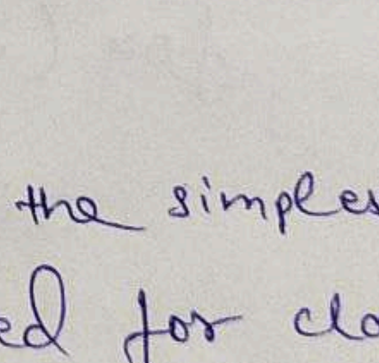
K=3

□ - 2

Δ - 1

(∴) ■ - ?

K=5



▲ - 3

* - 2

? - ▲

KNN-algorithm is also known as Lazy algorithm. and K value must be odd.

KNN-C (K Nearest Neighbourhood for classification)

*) Given the sample data set $D = \{x_i, y_i\}_{i=1}^{N}$

$x_i \in R^p$, $y_i \in \{1, 2, 3 \dots, m\}$ and

K must be an odd nearest neighbour for any $x \neq x_i$,

$\forall i =$ ~~1 to 0~~ 1 to N, determine the label of y to which it belongs.

Input :- $D = \{x_i, y_i\}_{i=1}^{N}$, K, $x \neq x_i \in R^p$

Output :- Class label of x ie determined y of x.

- KNN Algorithm for Classification (KNN-C)

1) Read the dataset $D = \{x_i, y_i\}_{i=1}^{N}$, K, $x \neq x_i \in R^p$

2) Determine the distances.

for (i to N)

$$d_i = \sqrt{\sum_{j=1}^{p} (x - x_{ij})^2}$$

3) Arrange the distances in non-decreasing order.

4) Select K-smallest distances and their associated n_j's.

5) Let $N_x = \{x_j\text{'s} \mid d_j$ is among the K-smallest$\}$

6) Determine the "n_j" whose number is maximum among those n_j's i.e; $l = \arg\max\{n_j\}$

7) Return l : The label class of that "n_j".

Q) Engineers results are out,

query → $\underset{\text{Student}}{X} = (\text{Math} = \underset{\text{CGPA}}{6}, \text{CS} = \underset{\text{CGPA}}{8}), \underline{K=3}$

Given dataset:-

	Math.	CS	Result
1	4	3	Fail
2	6	7	Pass ✓
3	7	8	Pass ✓
4	5	5	Fail
5	8	8	Pass ✓

Euclidean distance

$$d = \sqrt{|x_{O_1} - x_{A_1}|^2 + |x_{O_2} - x_{A_2}|^2}$$ or $$\boxed{d_i = \sqrt{\sum_{j=1}^{p} (x - x_{ij})^2}}$$

where, O → observed value

A → actual value

$$d_1 = \sqrt{(6-4)^2 + (8-3)^2} = \sqrt{4+25} = \sqrt{29} = 5.39$$

$$d_2 = \sqrt{(6-6)^2 + (8-7)^2} = \sqrt{0+1} = \sqrt{1} = \underline{1}$$

$$d_3 = \sqrt{(6-7)^2 + (8-8)^2} = \sqrt{1+0} = \sqrt{1} = \underline{1}$$

$$d_4 = \sqrt{(6-5)^2 + (8-5)^2} = \sqrt{1+9} = \sqrt{10} = 3.16$$

$$d_5 = \sqrt{(6-8)^2 + (8-8)^2} = \sqrt{4+0} = \sqrt{4} = \underline{2}$$

For K=3 (nearest neighbour).

d_2, d_3, d_5 are the nearest neighbour for K=3.

$d_2 = d_3 = d_5 =$ PASS.

(∴) Student x is PASS.

Q) Training dataset having six observation, three predictor X_1, X_2, X_3 and one qualitative response variable Y. Predict the value of Y when $X_1 = 2, X_2 = 1, X_3 = 3$ using K-KNN with K=3.

Observation	$X_1 = 2$	$X_2 = 1$	$X_3 = 3$	Y
1	0	2	0	Yellow
2	2	1	1	Yellow
3	2	2	0	Green
4	0	1	3	Yellow
5	-2	1	0	Green
6	1	1	2	Yellow

$$d_i = \sqrt{\sum_{j=1}^{p} (x - x_{ij})^2}$$

$$d_1 = \sqrt{(2-0)^2 + (1-2)^2 + (3-0)^2} = \sqrt{14} = 3.74$$

$$d_2 = \sqrt{(2-2)^2 + (1-1)^2 + (3-1)^2} = \sqrt{4} = 2$$

$$d_3 = \sqrt{(2-2)^2 + (1-2)^2 + (3-0)^2} = \sqrt{10} = 3.16$$

$$d_4 = \sqrt{(2-0)^2 + (1-1)^2 + (3-3)^2} = \sqrt{4} = 2$$

$$d_5 = \sqrt{(2+2)^2 + (1-1)^2 + (3-0)^2} = \sqrt{25} = 5$$

$$d_6 = \sqrt{(2-1)^2 + (1-1)^2 + (3-2)^2} = \sqrt{2} = 1.414$$

d_2, d_4 & d_6 are nearest neighbour.

$d_2 = d_4 = d_6 =$ Yellow.

$(\because)$ Y = Yellow

- **KNN - Regression :- (KNN - R)**

KNN - has higher prediction power as compare to Linear Regression.

Input :- Given $D = \{x_i, y_i\}_{i=1}^{N}$, $x_i \in R^p$

Output :- $y \in R$

Algorithm

1) Read the Dataset $D = \{x_i, y_i\}_{i=1}^{N}$, $y \in R$

2) Determine the distances,

for (i to N)

$$d_i = \sqrt{\sum_{j=1}^{p} (x - x_{ij})^2}$$

3) Arrange the distances in non-decreasing order.

4) There are K - no. of data points Y.

$$y = \frac{1}{K} \sum_{x_i \in N_x} x_i$$

5) This Y is the predictive value for new x.

Q) what is the age when Weight is 56 ? K = 3

Age	Weight
20	50
24	54 ✓
25	52 ✓
30	60 ✓

$$Age = \frac{24 + 25 + 30}{3}$$

$$= 26.33$$

Q	x_1 (Age)	x_2 (Height)	x_3 (weight)
1	19	155	65
2	20	161	72
3	25	163	59
4	21	170	66
5	40	171	67
	26	162	68

$K = 3$ Compute

$d_1 = \sqrt{(26-19)^2 + (68-65)^2} = \sqrt{58} = 7.6$

$d_2 = \sqrt{(26-20)^2 + (68-72)^2} = \sqrt{52} = 7.21$

$d_3 = \sqrt{(26-25)^2 + (68-59)^2} = \sqrt{82} = 9.05$

$d_4 = \sqrt{(26-21)^2 + (68-66)^2} = \sqrt{29} = 5.38$

$d_5 = \sqrt{(26-40)^2 + (68-67)^2} = \sqrt{197} = 14.03$

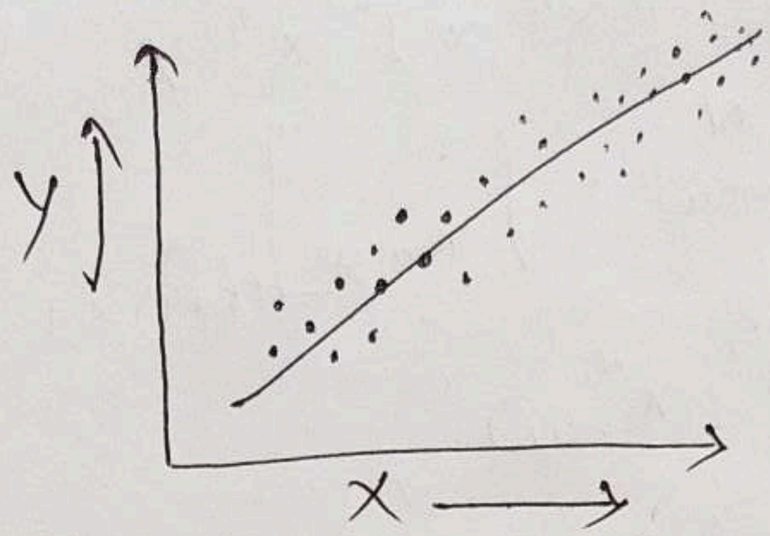
$d_4 < d_2 < d_1 < d_3 < d_5$

$\Rightarrow \frac{170 + 161 + 155}{3}$

$= 162$ ✓

- Linear Regression

Dependent variable is continous in nature.



Simple Linear Eqn

$$y = \alpha_0 + \alpha_1 x_1$$

$$y = mx + c$$

- Multiple linear Eqn

$$y = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \cdots \alpha_m x_m$$

α_i = Reg. Coeff

x_i = Independent Variable

y = Dependent Variable

Q) The table below provides a training data set containing four observation. Find the Linear Regression for Y.

X	Y	XY	X^2
1	3	3	1
2	4	8	4
3	5	15	9
4	7	28	16
10	19	54	30

$$y = bx + a \quad , b = \text{slope}$$

a = constant

$$y = 1.3x + 1.5$$

$$a = \frac{(\Sigma y)(\Sigma x^2) - (\Sigma x)(\Sigma xy)}{n(\Sigma x^2) - (\Sigma x)^2}$$

$$= \frac{19 \times 30 - 10 \times 54}{4 \times 30 - 100}$$

$$= \frac{570 - 540}{120 - 100} = \frac{30}{20} = 1.5$$

$$b = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{n(\Sigma x^2) - (\Sigma x)^2}$$

$$= \frac{4 \times 54 - 10 \times 19}{4 \times 30 - 100}$$

$$= \frac{216 - 190}{120 - 100} = \frac{26}{20} = 1.3$$

• Multiple Linear Regression

1) Read the Dataset $D: \{x_i, y_i\}_{i=1}^{N}$, $x_i \in \mathbb{R}^p$, $y_i \in$

2) Compute $\hat{\beta} = (X^T X)^{-1} X^T y$

3) Generate $x_{new}^{N} = \begin{bmatrix} 1 \\ x_{new} \end{bmatrix}_{(p+1)\times 1}$

4) $y_{new} = \hat{\beta}^T \cdot x_{new}^{N}$

$$y_{new} = [\beta_0 \ \beta_1 \ \beta_2 \ \cdots \ \beta_p] \begin{bmatrix} 1 \\ x_{new1} \\ x_{new2} \\ x_{new3} \\ \vdots \\ x_{newp} \end{bmatrix}$$

$$= \beta_0 + \beta_1 * x_{new1} + \beta_2 * x_{new2} + \cdots + \beta_p * x_{newp}$$

Q) Suppose we have a dataset with five predictors,
x_1 = GPA, x_2 = IQ, x_3 = Gender (1 for female & 0 for male)
x_4 = Interaction betwn GPA & IQ
x_5 = Interaction betwn GPA & Gender ...

The response is starting salary after graduation (in thousand dollars). Suppose we have the least squares to fit the models and get $\beta_0 = 50$, $\beta_1 = 20$, $\beta_2 = 0.07$, $\beta_3 = 35$, $\beta_4 = 0.01$, $\beta_5 = -10$.

1) "For a fixed value of IQ and GPA, male earn more on average than females provided that the GPA is high enough". Is this statement correct ? Justify your answer.

A) For female

$$\hat{y}_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5$$

$$= 50 + 20\,GPA + 0.07\,IQ + 35 \times 1 + 0.01 \times GPA \times IQ + -10 \times GPA \times 1$$

$$= 85 + 20\,GPA + 0.07\,IQ + 0.01 \times GPA \times IQ - 10\,GPA$$

$$= (85 + 10\,GPA + 0.07\,IQ + 0.01\,GPA \times IQ) \quad \text{—— (1)}$$

For male

$$\hat{y_2} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5$$

$$= \underline{50} + 20\,GPA + 0.07\,IQ + \underline{35 \times 0} + 0.01 \times GPA \times IQ + \underline{-10 \times GPA \times 0}$$

$$= 50 + 20\,GPA + 0.07\,IQ + 0.01\,GPA \times IQ \quad -(2)$$

When, Subtract both the eqn, we get;

$= 35 - 10^{*}$ mean (GPA) more than men,

Since we don't know the value of mean (GPA). we don't know whether men or women are average or not.

- Since, women are earning an average of $35 - 10^{*}$ mean (GPA) more than men, a higher GPA means that women earn less than men, so this is true.

B) Predict the salary of a female with IQ of 110 and a GPA of 4.0.

A)

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5$$

$$= 50 + 20 \times GPA + 0.07\,IQ + 35 \times 1 + 0.01\,GPA \times IQ + -10 \times GPA \times 1$$

$$= 50 + 20 \times 4.0 + 0.07 \times 110 + 35 + 0.01 \times 4.0 \times 110 + -10 \times 4.0 \times 1$$

$$= 50 + 80 + 7.7 + 35 + 4.4 - 40$$

$$= 137.1$$

- Multiple Linear Regression

Input : $D: \{x_i, y_i\}_{i=1}^{N}$, $x_i \in \mathbb{R}^p$, $y_i \in \mathbb{R}$

Output : $\hat{y} = ?$

Algorithm

1) Read the dataset $D: \{x_i, y_i\}_{i=1}^{N}$

⋮

Derivation

$$y = f(x)$$

$$y = f(x_1, x_2 x_3 \cdots x_p)$$

$$\Rightarrow \hat{y} = f(x_1, x_2 x_3 \cdots x_p) + E(\epsilon)$$

where, $E(\epsilon)$ is represented as Root Mean Square (RMS) need to be zero or near to zero.

i.e; we have to determine such that,

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \cdots + \beta_p x_p \quad -(1)$$

where, $(x_1, x_2 \cdots x_p)$ ~~or~~ are given and we have to determine $(\beta_0, \beta_1, \beta_3 \cdots \beta_p)$.

$$D: \begin{bmatrix} \hat{y_1} = x_{11}\beta_1 + x_{12}\beta_2 + \cdots x_p\beta_p \\ \hat{y_2} = x_{21}\beta_1 + x_{22}\beta_2 + \cdots x_{2p}\beta_p \\ \vdots \\ \hat{y_i} = x_{i1}\beta_1 + x_{i2}\beta_2 + \cdots x_{ip}\beta_p \\ \vdots \\ \hat{y_N} = x_{N1}\beta_1 + x_{N2}\beta_2 + x_{N3}\beta_3 + x_{Np}\beta_p \end{bmatrix}$$

$D: \{x_i, y_i\}_{i=1}^{N}$

where,

y - Actual value

$\hat{y}$ - predicted value

$$(\because) \quad RSS(\beta) = \sum_{i=1}^{N} (y_i - \hat{y_i})^2$$

$$\text{Min. } RSS(\beta) = \min_{\beta} \sum_{i=1}^{N} (y_i - \hat{y_i})^2, \quad \text{if } \nabla\beta\, RSS(\beta) = 0$$

The functn attains its minimum.

$$\nabla f(x_1, x_2, x_3 \cdots x_p) = 0$$

This is the necessary condition.

i.e; $\nabla_\beta (f \cdots) = 0 \Rightarrow \frac{\partial f}{\partial \beta_0} = 0$

$$\frac{\partial f}{\partial \beta_1} = 0$$

$$\vdots$$

$$\frac{\partial f}{\partial \beta_p} = 0$$

This condition is the sufficient condition,

$$H: \begin{bmatrix} \frac{\partial^2 f}{\partial \beta_0^2} & \frac{\partial^2 f}{\partial \beta_0 \partial \beta_1} & \frac{\partial^2 f}{\partial \beta_0 \beta_2} & \cdots \frac{\partial^2 f}{\partial \beta_0 \partial \beta_p} \\ \vdots & & & \\ \frac{\partial^2 f}{\partial \beta_r \partial \beta_0} & \frac{\partial^2 f}{\partial \beta_p \partial \beta_1} & \frac{\partial^2 f}{\partial \beta_p \partial \beta_2} & \cdots \frac{\partial^2 f}{\partial \beta_r^2} \end{bmatrix}$$

Necessary condition :

$$RSS(\beta) = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \quad \text{—— (1)}$$

where, Y - n×0 vector
β - column vector

$$= [Y - X\beta]^T [Y - X\beta] \quad \text{—— (2)}$$

$$RSS(\beta) = z^T . z$$

$$\frac{\partial}{\partial \beta}(RSS(\beta)) = \frac{\partial}{\partial \beta}(z^T . z)$$

$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial z} = \frac{\partial z}{\partial x}$$

$$\Rightarrow \frac{\partial}{\partial \beta}(Y - X\beta) . \partial z$$

$$= \frac{\partial Y}{\partial \beta} - \frac{\partial (X\beta)}{\partial \beta} . 22$$

$$= 0 - \frac{\partial}{\partial \beta}(X\beta) . 22$$

$$= X^T Y = X^T X \beta$$

$$\Rightarrow \boxed{\beta = (X^T X)^{-1} X^T Y}$$

Module 2

Q) Describe the K-Fold cross validation method for obtaining the test error of a regression model. How is cross validation used when dimensionality reduction is applied before the learning algorithm?

A) We can estimate the test error from training error.

By using :-

1) K-Fold Cross Validation.
2) Bootstrap method.

#1) K-Fold Cross Validation

Let T be the Training Data.

1) Divide the T into K subsets $T_1, T_2, T_3 \dots T_K$ such that $|T_1| = |T_2| = |T_3| \approx |T_K|$

2) For each 1, 2, 3 ... K take T_i as Test set and $T - T_i$ as the Training model.

3) Test a model using the data from T_i.

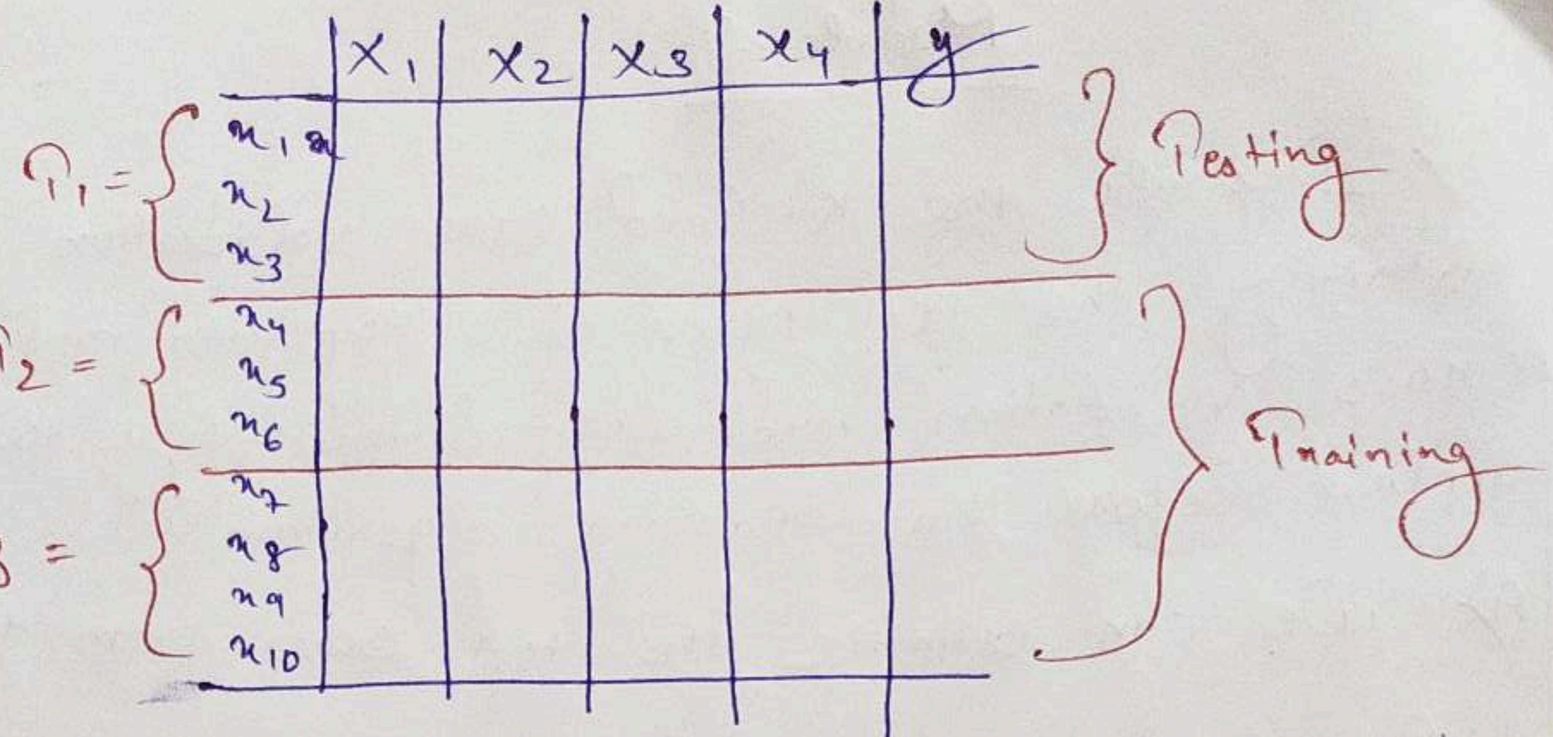
Compute the Cross-Validation Error.

$$CV_K = \frac{1}{K} \sum_{i=1}^{K} Err_i \quad \text{where } Err_i = \text{Mean Square Error}$$

$$\Rightarrow \boxed{MSE_i = \frac{1}{|T_i|} \sum_{(x_i, y_i) \in T_i} (y_j - \hat{f}(x_j))^2}$$

4) The CV_K computed is the estimate of the test error for K-subset.

Ex:-



When we have a multistep modules such as, using the learning algorithm after dimensionality reduction, then cross-validation is used as follows

1) K-CV normal

2) K-CV after reduction

1) Divide the dataset into K-subsets $T_1, T_2, T_3 \dots T_K$, $|T_1| = |T_2| = = \dots \approx |T_K|$

2) For $i = 1$ to K for ~~such~~ each subset determine the significant predictors using entire data set excluding the i^{th} subset.

3) Apply the learning algorithm with the set of predictors using the entire dataset excluding the i^{th} subset, as a training set and let the learn module be $\hat{f}_i$:

4) Compute the C_V as same, ~~C_V~~

$$C_V = \frac{1}{K} \sum_{i=1}^{K} Err_i, \text{ where}$$

CV_K - estimate of test error

Imp

(2)

Q) Describe the bootstrap method to estimate a statistic using a sample of N ~~as~~ observations. When does one use this method?

A) When the amount of dataset is less and we need to estimate the test error in that case, bootstrap method is applied, it's a flexible and powerful tool that can be used to :-

1) Quantify uncertainty associated with a given estimator or ML method.

2) Access the statistical accuracy of the model.

Ex:- Let we have a training data: $Z = (Z_1, Z_2, Z_3 \dots Z_N)$

where $Z_i = (x_i, y_i)$

	x_1	x_2	x_3	$\dots x_p$	y
Z_1					
Z_2					
Z_3					
$\vdots$					
Z_N					

We have a model which fits to the training data.

$$Z = \begin{bmatrix} x_{11} & x_{12} & x_{13} \dots x_{1p} & y_1 \\ x_{21} & x_{22} & x_{23} \dots x_{2p} & y_2 \\ \vdots & & & \vdots \\ x_{i1} & x_{i2} & x_{i3} \dots x_{ip} & y_i \\ \vdots & & & \vdots \\ x_{N1} & x_{N2} & x_{N3} \dots x_{Np} & y_N \end{bmatrix}$$

$Z =$ [z_i]

Replace this data with a new dataset.

$Z =$ [z_j]

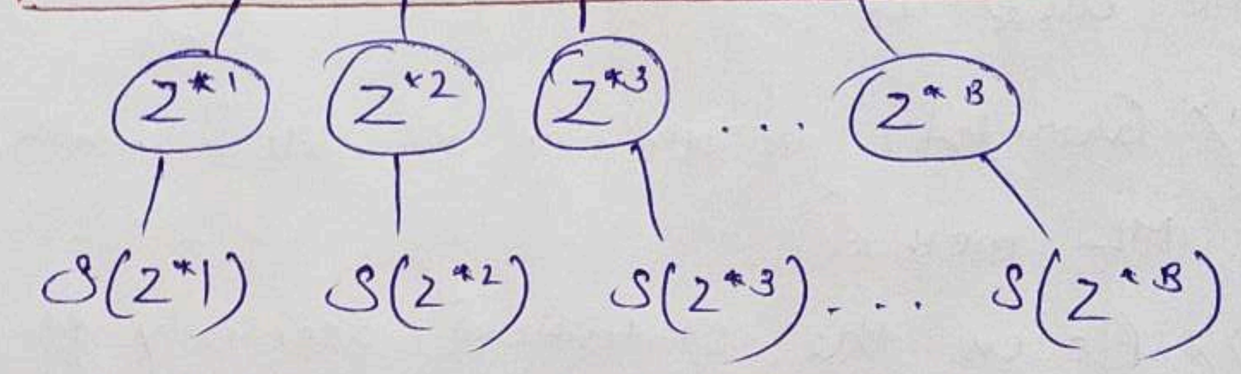
Replace this data with a new dataset.

Repeat the model to each of the dataset model and examine the behaviour of the model over B.

- **Bootstrap Algorithm**

1) Predicting / Estimating the test error from training error.

$$Z = (Z_1, Z_2, Z_3 \cdots, Z_n)$$



Here, $S(Z)$: Some quantity computed from the data set Z. The prediction at that point.

$$\overline{S^*} = \frac{1}{B} \sum_{b=1}^{B} S(Z^{*b})$$

$$\hat{Var}[S(Z)] = \frac{1}{B-1} \sum_{b=1}^{B} \left(S(Z^{*b}) - \overline{S^*}\right)^2$$

The bootstrap method is applied in :-

1) Financial investment (Ex :- Stock

We have a fixed sum of money that yield returns x and y invest α in x and invest $1-\alpha$ in y.

Here α is selected as :-

$$\alpha = \frac{\sigma_y^2 - \sigma_{xy}}{\sigma_x^2 + \sigma_y^2 - 2\sigma_{xy}}$$

where, $\sigma_x^2 = Var(x)$

$\sigma_y^2 = Var(y)$

$\sigma_{xy} = Cov(x, y)$

Here the $Var(x, y)$ and $Cov(x, y)$ are Unknown. So, we have estimate those values using previous dataset of x and y.

We can then estimate the value of α, that minimizes the risk.

i.e; $$\hat{\alpha} = \frac{\hat{\sigma}_y^2 - \hat{\sigma}_{xy}}{\hat{\sigma}_x^2 + \hat{\sigma}_y^2 - 2\hat{\sigma}_{xy}}$$

Conclusion:- The estimated value of the α should be such that, the subsequent error should be within the desire range.

Limitation : In the real world these procedures can't be applied because real data are replaced by manipulated data, so the machine learning algorithm mimic the process of obtaining new data. These process enables us to estimate the uncertainity of the model without generating additional samples. (Because we would predict the future and the future data is unknown).

Q) What is the need of Boosting ? State and explain Adaboost algorithm with Schematic Diagram ?

A) For any certain task if the models have slightly above 50% accuracy but not achieved desired accuracy in that case, the models will be combined ~~to~~ so as to yield higher accuracy.

i.e; Boosting is a method of ~~ach~~ combining weak models to achieve higher accuracy. ~~For ex~~

For example :- The task is :

Dataset : $\{x_i, y_i\}_{i=1}^{N}$, $x_i \in \mathbb{R}^p$: For Regression

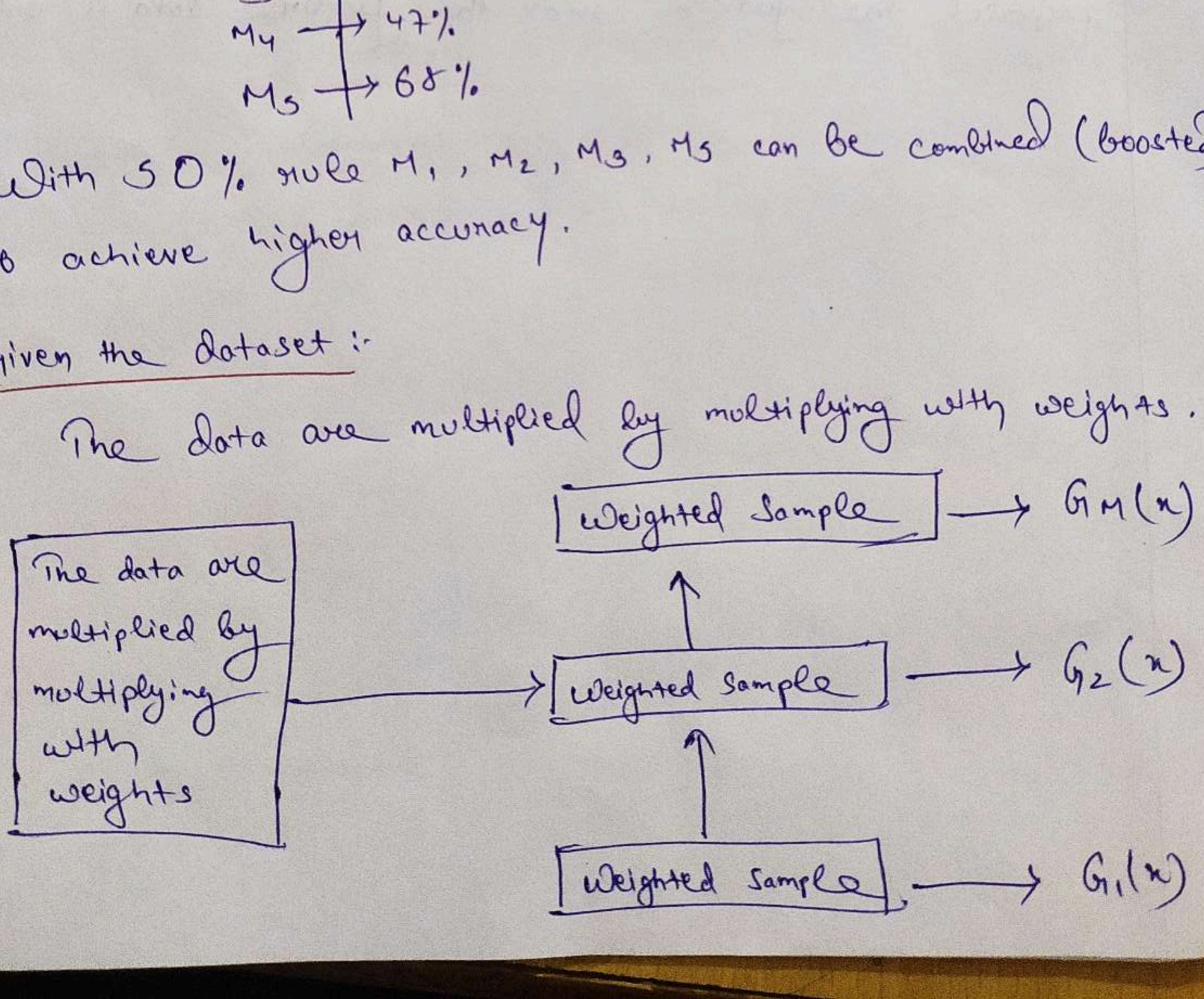
or $y_i \in \{0, 1\}$: For Classification

Model	Accuracy
M_1	61%
M_2	72%
M_3	54%
M_4	47%
M_5	68%

With 50% rule M_1, M_2, M_3, M_5 can be combined (boosted) to achieve higher accuracy.

Given the dataset :-

The data are multiplied by multiplying with weights.



- **Algorithm (AdaBoost)**

1) Initialize observation weights $w_i = \frac{1}{N}$, $i = 1, 2, 3 \cdots N$

2) for $m = 1$ to M

(a) Fit a classification ~~data~~ G_m to train the data using weight w_i

(b) Compute $err_m = \dfrac{\sum_{i=1}^{N} w_i \, I(y_i \neq G_m(x_i))}{\sum_{i=1}^{N} w_i}$

where, I: Indicator

$y_i = G_m(x_i)$ - Correct classification

$y_i \neq G_m(x_i)$ - Incorrect classification

$I = 1$, for misclassification

$I = 0$, for correctness

(c) Compute $\alpha_m = \log\left(\dfrac{1 - err_m}{err_m}\right)$

(d) Set $w_i \leftarrow w_i \exp\left(\alpha_m \cdot I(y_i \neq G_m(x_i))\right)$ $i = 1$ to N

Output: $G(x) = \text{sign} \sum_{m=1}^{M} \alpha_m * G_m(x)$

The AdaBoost algorithm is executed sequentially by considering the weak models one after another.

- When the weak models performance are combined through a weighted majority vote to produce the final prediction. i.e; $\alpha_1 G_1(x) + \alpha_2 G_2(x) + \alpha_3 G_3(x) + \alpha_m G_m(x)$, then predict through the voting which can be obtained by sign (signum) operator. Besides that which class it belongs to:

Model	Accuracy	Error
M_1	61%	.39
M_2	79%	.21
M_3	81%	.19
M_4	45%	X
M_5	62%	.38

Imp

Q) Write an algorithm for building the regression tree and explain the steps?

A) It is used for (i) classification & (ii) Regression

* Requirement

To determine the

i) Spitting Variable

ii) Spitting Point

Ex:- Let $D: \{x_i, y_i\}_{i=1}^{N}$, $x_i \in \mathbb{R}^2$, $y_i \in \{0, 1\}$

Construct a decision tree for the given data set;

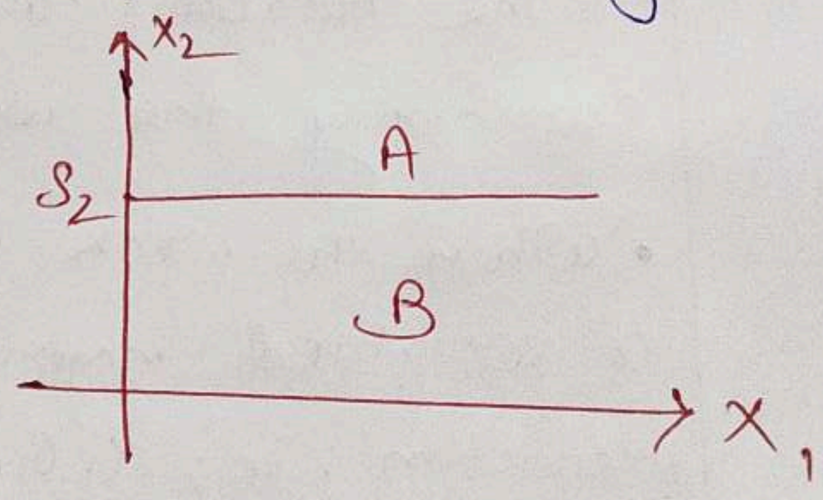
$D = \{(x_1, x_2) \mid x_1, x_2 \in \mathbb{R}\}$ Here $p = 2$

1st Split: Let us split X_2 at S_2.

$\Rightarrow$ S_2 partition the predictors space into 2 regions.

$A = \{x \mid x_2 > S_2\}$

$B = \{x \mid x_2 \leq S_2\}$

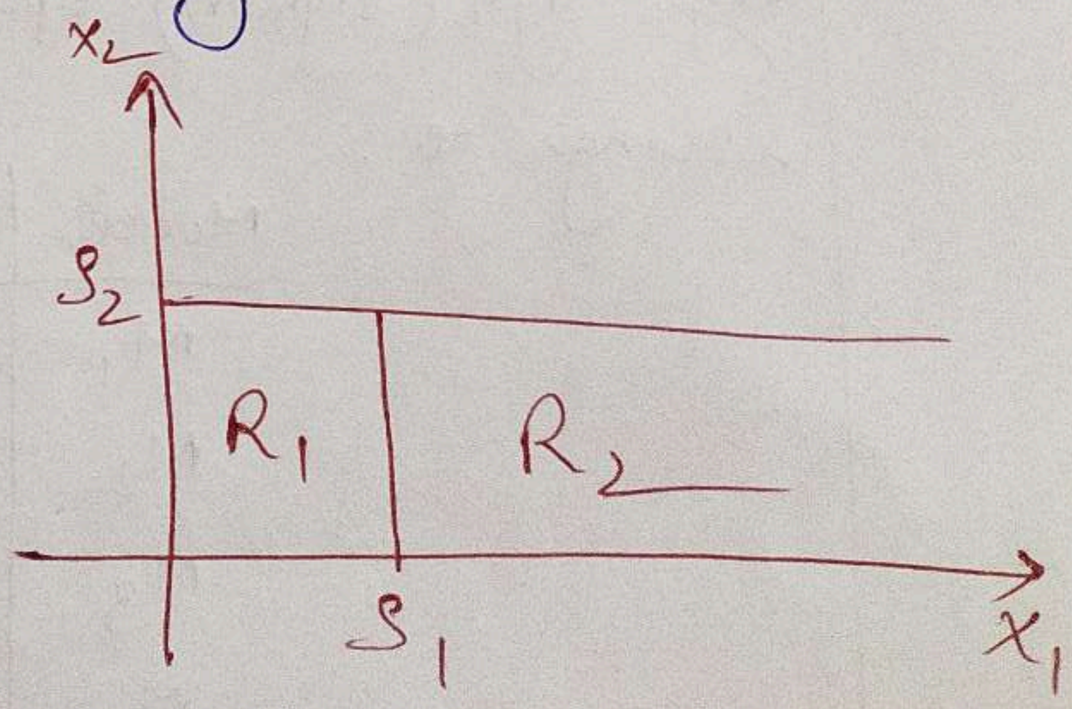


2nd Splitting:

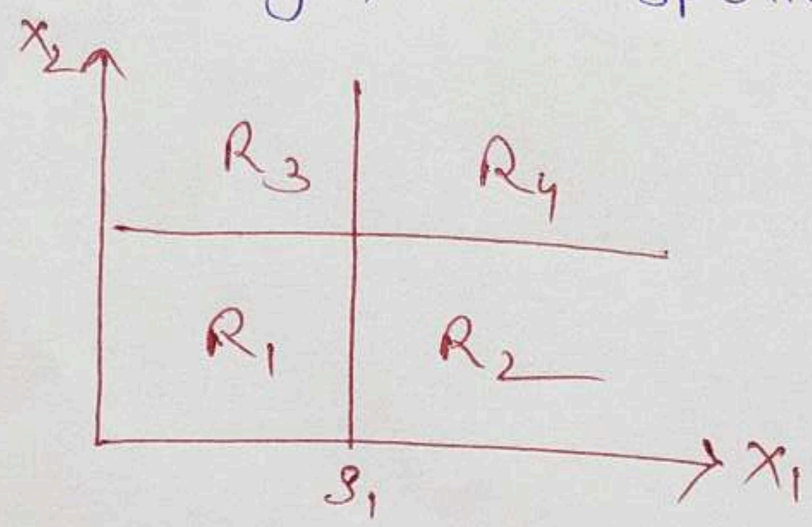
Let region B is spitted using X_1 at S_1.

$R_1 = \{x \mid x_1 < S_1\}$

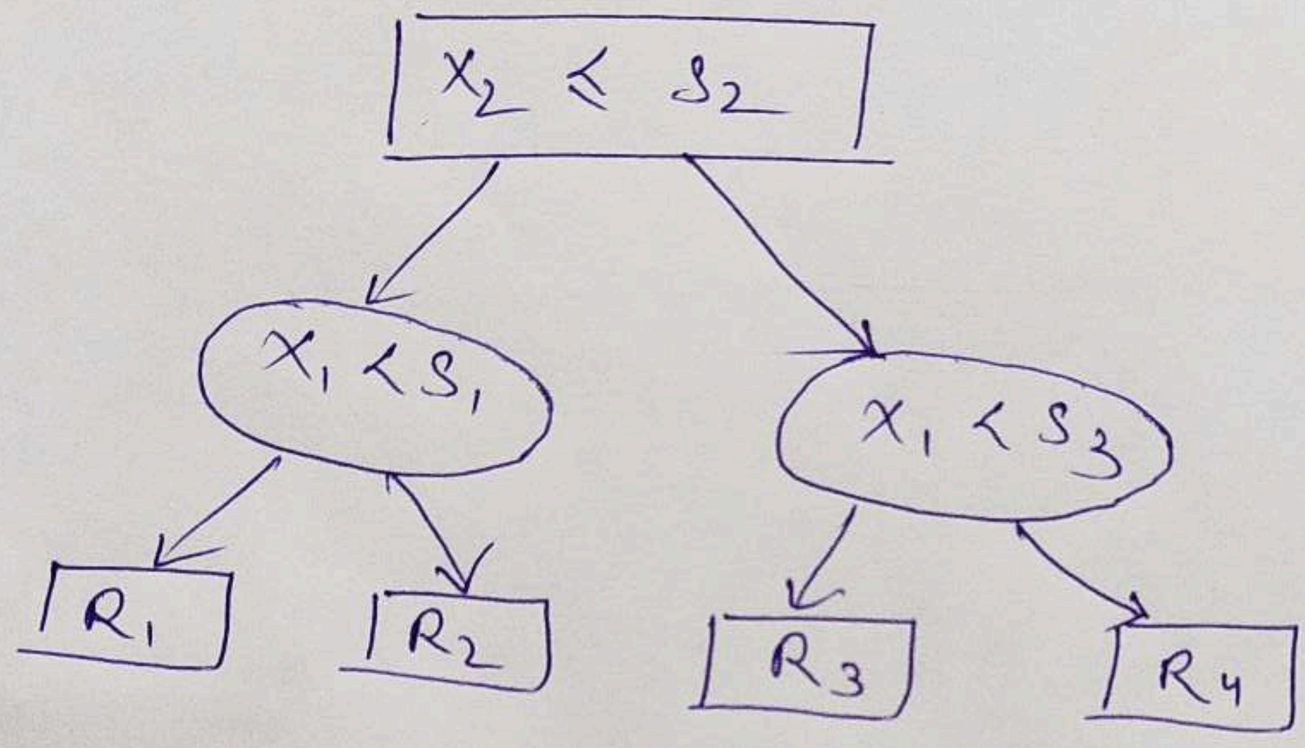
$R_2 = \{x \mid x_1 \geq S_1\}$



3rd Splitting : Let region A is splitted using X_1 at s_3.



The decision tree is,



So, giving the splitting variables and splitting points, decision tree can be constructed.

Module-3

Q)

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D15	overcast	mild	normal	weak	?

Using naive Bayes classifier

$P(Yes) = \frac{9}{14}$

$P(No) = \frac{5}{14}$

$P(\text{overcast} \mid Yes) = \frac{4}{9}$

$P(\text{overcast} \mid No) = \frac{0}{5}$

$P(mild \mid Yes) = \frac{4}{9}$

$P(mild \mid No) = \frac{2}{5}$

$P(normal \mid Yes) = \frac{6}{9}$

$P(normal \mid No) = \frac{1}{5}$

$P(weak \mid Yes) = \frac{6}{9}$

$P(weak \mid No) = \frac{2}{5}$

For Yes,

$P(Yes) \cdot P(overcast \mid Yes) \cdot P(mild \mid Yes) \cdot P(normal \mid Yes) \cdot P(weak \mid Yes)$

$= \frac{9}{14} \times \frac{4}{9} \times \frac{4}{9} \times \frac{6}{9} \times \frac{6}{9}$

$= \frac{5184}{91854} = 0.05$

For No,

$P(No) \cdot P(overcast \mid No) \cdot P(mild \mid No) \cdot P(normal \mid No) \cdot P(weak \mid No)$

$= \frac{5}{14} \times \frac{0}{5} \times \frac{2}{5} \times \frac{1}{5} \times \frac{2}{5} = 0.$

$= 0$

$P(overcast, mild, normal, weak)_{Yes} > P(overcast, mild, normal, weak)_{No}$

=> ~~Not~~ Yes, Match will be played.

- **Confusion Matrix** :-

Q) 1000 sample $[P + N = 1000]$

50 % of sample is -ve $\Rightarrow$ FP + TN = 500

50 % of sample is +ve $\Rightarrow$ TP + TN = 500

60 % of sample is sensitive.

$(\because)$ Sensitivity $= \frac{TP}{P} = \frac{TP}{TP+FN} \Rightarrow \frac{60}{100} = \frac{TP}{500} \Rightarrow \underline{TP = 300}$

70 % Accuracy.

$(\because)$ Accuracy $= \frac{TP + TN}{P + N} = \frac{300 + TN}{1000} \Rightarrow \frac{70}{100} = \frac{300 + TN}{1000}$

$\Rightarrow TN = \left(\frac{70}{100} \times 1000\right) - 300$

$\Rightarrow \underline{TN = 700 - 300 = 400}$

(i)

	1	0
1	TP 300	FP 100
0	FN 200	TN 400
	500	500

(ii) $\underline{\text{Specificity}} = \frac{TN}{TN + FP} = \frac{400}{500} = \frac{4}{5} = 0.8$

(iii) Precision $= \frac{TP}{TP + FP} = \frac{300}{300 + 100} = \frac{3}{4} = ~~0.6~~ \; 0.75$

Recall $= \frac{TP}{TP + FN} = \frac{300}{500} = \frac{3}{5} = 0.6$

(iV) F-Score $= \frac{2 * \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

$= \frac{2 * 0.75 \times 0.6}{0.75 + 0.6}$

$= \frac{0.9}{1.35}$

$= 0.67$

Q) Derive the formula for the Naive - Bayes classifier. What is its limitation ? How are the probabilities involved in the formula estimated ?

To classify certain event based on classified condition the Naive - Bayes is a suitable method.

$$P(a_1\ a_2\ a_3\ a_4 \dots a_N \mid v) = \pi_i\ P(a_i \mid v_j)$$

$$P(c|x) = \frac{P(x|c) \cdot P(c)}{P(x)}$$

Likelihood ↙

$P(c)$ ← class prior probability

$P(x)$ ← predictor for prior probability

↓

Posterior probability

Characteristics for Naive Bayes :-

1) It requires less memory.

2) It is very fast and robust.

3) It is robust to irrelevant features.

4) When there is little amount of data in that case the decision tree suffers from fragmentation. However Naive Bayes produces much better result.

5) It can be applied as email spam classifier.

Module-4 (Clustering)

Q) K-Mean Clustering , K = 3

Iteration-1

		x_1	x_2	d_1	d_2	d_3	Cluster
$m_1 \rightarrow$	A_1	2	9	0	3.6	2.2	1
	A_2	2	4	5	4.2	4.5	2
$m_2 \rightarrow$	B_1	5	7	3.6	0	1.4	2
	B_2	6	4	6.4	3.2	4.5	2
	C_1	1	2	7	6.4	6.7	2
$m_3 \rightarrow$	C_2	4	8	2.2	1.4	0	3

Cluster 1

$A_1 (2, 9)$

Cluster 2

$A_2 (2, 4)$

$B_1 (5, 7)$

$B_2 (6, 4)$

$C_1 (1, 2)$

Cluster 3

$C_2 (4, 8)$

New Centroid

$m_1 new = (2, 9)$

$m_2 new = \left(\frac{2+5+6+1}{4}, \frac{4+7+4+2}{4}\right) = (3.5, 4.25)$

$m_3 new = (4, 8)$

Iteration-2

	X_1	X_2	d_1	d_2	d_3	Cluster
A_1	2	9	0	~~5~~ 4.9	2.2	1
A_2	2	4	5	1.5	4.2	2
B_1	5	7	3.6	3	1.4	3
B_2	6	4	6.4	2.5	4.5	2
C_1	1	2	7	3.4	6.7	2
C_2	4	8	2.2	3.7	0	3

$m_1 =$
$m_2 = (3$
$m_3 = (4, 8)$

$m_1 = (2, 9)$
$m_2 = (3.5, 4.3)$
$m_3 = (4, 8)$

Cluster 1: $A_1(2,9)$

Cluster 2: $A_2(2,4)$, $B_2(6,4)$, $C_1(1,2)$

Cluster 3: $B_1(5,7)$, $C_2(4,8)$

New centroids

$m_1 = (2, 9)$, $m_2 = \left(\frac{2+6+1}{3}, \frac{4+4+2}{3}\right)$; $m_3 = \left(\frac{5+4}{2}, \frac{7+8}{2}\right)$

$m_2 = (3, 3.3)$ $m_3 = (4.5, 7.5)$

Iteration-3

	X_1	X_2	d_1	d_2	d_3	Cluster
A_1	2	9	0	5.8	2.9	1
A_2	2	4	5	1.2	4.3	2
B_1	5	7	3.6	4.2	0.7	3
B_2	6	4	6.4	3	3.8	2
C_1	1	2	7	2.4	6.5	2
C_2	4	8	2.2	4.8	0.7	3

Cluster 1: $A_1(2,9)$

Cluster 2: $A_2(2,4)$, $B_2(6,4)$, $C_1(1,2)$

Cluster 3: $B_1(5,7)$, $C_2(4,8)$

K-mean will stop as one of the data points doesn't change the clusters.

Q) Write down the K-means algorithm and explain the steps?

Algorithm :-

Input : $D = \{x_i\}_{i=1}^{N}$, K

1) Initialize the cluster center as $m_1, m_2 \cdots m_K$.

2) for $i = 1$ to N

3) compute $l = \underset{i \le j \le K}{\arg\min} |x_i - m_j|$ i.e; x_i assigned to l^{th} cluster.

4) $C_i = l$

5) for $j = 1$ to K

6) $$m_j = \frac{\sum_{i=1}^{N} I(C_i = j) x_i}{\sum_{i=1}^{N} I(C_i = j)}$$

7) Compute with cluster variance $\bar{J} = \sum_{i=1}^{N} ||x_i - m_{C_i}||^2$

where C_i is the index of the cluster center to which x_i is assigned.

8) if $(\bar{J} \le \epsilon)$ /* Here ϵ : Error tolerance */

Output the cluster & Return and STOP.

else

Go To Step-2.

Q) Write down the PCA algorithm and explain the steps.

Principal component analysis

Input : $\{x_i\}_{i=1}^{N}$, $x_i \in \mathbb{R}^p$

1) for $i = 1$ to N do

2) for $j = 1$ to p do

3) $$u_j = \frac{\sum_{k=1}^{N} x_{kj}}{N}$$

4) $x_{ij} = x_{ij} - u_j$

5) Compute the covariance matrix of the transformed samples.

$$C = \frac{1}{N} \sum_{i=1}^{N} x_i x_i^T$$ /* Here C is a symmetric matrix of order $p \times p$ */

6) Determine p Eigen values of C and arrange them in non-decreasing order

i.e; $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \ldots \geq \lambda_p > 0$

7) Choose the smallest positive integer "m" such that

$$\sum_{i=1}^{m} \lambda_i \geq t \sum_{i=1}^{p} \lambda_i \text{ , where } t = 0.9 \text{ or } 0.95$$

8) Compute the Eigen vectors $\alpha_1, \alpha_2, \alpha_3 \ldots \alpha_m$ of C associated with the Eigen values $\lambda_1, \lambda_2, \lambda_3 \ldots \lambda_m$ such that

$\|\alpha_i\| = 1$, $i = 1$ to m. /* α_i – principal component directions */

9) Let $A = [\alpha_1, \alpha_2 \ldots \alpha_m]$

10) Transform each sample values $x_i \in \mathbb{R}^p$ to the vector $y_i \in \mathbb{R}^m$ using

$$y_i = A^T x_i \text{ for } i = 1 \text{ to } N$$

$m \times 1$ $\quad$ $m \times p$ $\quad$ $p \times 1$

Module-5

Q) Define Reinforcement Learning. Differentiate it from Supervised Learning. Mention it's four components and explain their role during modelling.

A) Reinforcement Learning (RL) is a type of machine learning where an ~~angent~~ agent learns to make decisions by interacting with its environment and receiving feedback in the form of reward or penalties. Whereas, Supervised Learning involves learning from ~~dat~~ labelled data, where the objective is to predict the output given a set of inputs.

In Summary, while Supervised learning (SL) focuses on learning from labeled data to make predictions, whereas RL focuses on learning from trial-and-error to determine the best sequence of actions to take in a given situation.

Components of Reinforcement Learning algorithm :-

1) **Model** :- It is a mathematical description of transitions and rewards of the agents environment.

$$P(S_{t+1} = s' \mid S_t = s, a_t = a)$$

The reward $\gamma(S_t = s, a_t = a) = E(\gamma_t \mid S_t = s, a_t = a)$

2) **Policy (π)** :- The policy decides or determines what should be the action at a particular state.

It is a function $\pi : S \longrightarrow A$

Policy can be of two types :-

1) Deterministic Policy : $\pi(s) = a$.

2) Stochastic Policy : $\pi(a|s) = P(a_t = a \mid s_t = s)$

3) Value function (V^π)

A measure of the expected future reward given a state and a policy.

$$V^\pi(s_t = s) = E_\pi \left[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{3+3} \mid s_t = s \right]$$

4) Reward

Q) What are the various ways of computing the state value function of a Markov Reward Process? Write the Monte Carlo Simulation algorithm to calculate MRP value function.

A) There are 3 different ways to compute the state value function of a MRP:-

1) Simulation.
2) Analytic Solution.
3) Iterative Solution.

- Monte-Carlo Simulation Algorithm :-

* Initially generates a large number (N) of episodes starting with state 's' and time 't'.
* Compute $G_t, \forall t$.
* Mean $G_t = \frac{\sum G_t}{N}$
* For a MRP : M(S, P, R, Y) state s, t, N

~~1) Monte Carlo S~~

1) MCS (M, S, t, N)
2) i = 0
3) $G_t = 0$
4) while (i <= N) {
5) Generate episodes starting from 's' and time 't' using generated episodes.
6) $g = \sum_{i=t}^{H-1} y^{i-t} * R_i$
7) $G_t = G_t + g$.
8) i++
9) }
10) $V_t(S) = G_t / N$
11) return $V_t(S)$

Q) Write the Dynamic Programming algorithm to calculate finite MRP.

Iterative Solution for Finite Horizon :-

MRP : $M(S, P, R, \gamma)$

Alg:

1) Dynamic Programming Evaluation (M)
2) $\forall S \in S$, $V_H(S) = 0$
3) $t = H-1$
4) while $(t \geq 0)$ {
5) $V_t(S) = R(S) + \gamma \sum_{S' \in S} P(S'|S)\, V_{t+1}(S')$, $\forall S \in S$
6) $t = t-1$
7) }
8) return $V_t(S)$ $\forall S \in S$ and $t = 0, 1, \ldots H-1$

Computational Cost = $O(|S|^2)$

Iterative Solution for infinite horizon :-

→ Iterative Value function Evaluation

Alg:-

1) IVFE (M, E)
2) $\forall s \in S$, $V'(S) = 0$, $V(S) = \infty$
3) while $(\|V - V'\|_\infty > \epsilon)$ {
4) $V = V'$
5) $\forall s \in S$, $V'(S) = R(S) + \gamma \sum_{S' \in S} P(S'|S)\, V(S')$
6) }
7) return $V'(S)$, $\forall s \in S$.

Computational Cost = $O(|S|^2)$