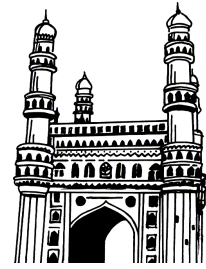


Rahul's ✓
Topper's Voice



M.C.A.

I Year II Sem

(Osmania University)

Latest 2023 Edition

MACHINE LEARNING

- ☞ Study Manual
- ☞ Important Questions
- ☞ Solved Model Papers
- ☞ Previous Question Papers

- by -

WELL EXPERIENCED LECTURER

Price
199-00



Rahul PublicationsTM
Hyderabad. Cell : 9391018098, 9505799122.

All disputes are subjects to Hyderabad Jurisdiction only

M.C.A.

I Year II Sem

MACHINE LEARNING

Inspite of many efforts taken to present this book without errors, some errors might have crept in. Therefore we do not take any legal responsibility for such errors and omissions. However, if they are brought to our notice, they will be corrected in the next edition.

© No part of this publication should be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording and/or otherwise without the prior written permission of the publisher

Price ` . 199/-

Sole Distributors :

Cell : 9391018098, 9505799122

VASU BOOK CENTRE

Shop No. 2, Beside Gokul Chat, Koti, Hyderabad.

Maternity Hospital Opp. Lane, Narayan Naik Complex, Koti, Hyderabad.

Near Andhra Bank, Subway, Sultan Bazar, Koti, Hyderabad -195.

MACHINE LEARNING

C O N T E N T S

STUDY MANUAL

Important Questions	III - VII
Unit - I	1 - 44
Unit - II	45 - 62
Unit - III	63 - 110
Unit - IV	111 - 142
Unit - V	143 - 184

SOLVED MODEL PAPERS

Model Paper - I	185 - 186
Model Paper - II	187 - 188
Model Paper - III	189 - 190

PREVIOUS QUESTION PAPERS

December - 2021	191 - 191
April - 2022	192 - 192

SYLLABUS

UNIT - I

Basic Maths: Probability, Linear Algebra, Convex Optimization **Background:** Statistical Decision Theory, Bayesian Learning (ML, MAP, Bayes estimates, Conjugate priors)

UNIT - II

Regression: Linear Regression, Ridge Regression, Lasso **Dimensionality Reduction:** Principal Component Analysis, Partial Least Squares

UNIT - III

Classification: Linear Classification, Logistic Regression, Linear Discriminant Analysis, Quadratic Discriminant Analysis, Perceptron, Support Vector Machines + Kernels, Artificial Neural Networks + Back Propagation, Decision Trees, Bayes Optimal Classifier, Naive Bayes.

UNIT - IV

Evaluation measures: Hypothesis testing, Ensemble Methods, Bagging, Adaboost Gradient Boosting, Clustering, K-means, K-medoids, Density-based Hierarchical, Spectral

UNIT - V

Miscellaneous topics: Expectation Maximization, GMMs, Learning theory, Introduction to Reinforcement Learning **Graphical Models:** Bayesian Networks. Use Cases of various ML Algorithms in Manufacturing, Retail, Transport, Healthcare, weather, insurance sectors.

Contents

Topic	Page No.
UNIT - I	
1.1 Basic Maths	1
1.1.1 Probability	1
1.1.2 Linear Algebra	21
1.2 Convex Optimization Background	36
1.3 Statistical Decision Theory	38
1.4 Bayesian Learning (ML, MAP, Bayes Estimates, Conjugate Priors)	42
UNIT - II	
2.1 Regression	45
2.1.1 Introduction	45
2.1.2 Ridge Regression	49
2.2 Lasso Dimensionality Reduction	51
2.2.1 Principal Component Analysis	55
2.2.2 Partial Least Squares	61
UNIT - III	
3.1 Classification	63
3.1.1 Linear classification	66
3.1.2 Logistic regression	69
3.1.3 Linear Discriminant Analysis	71
3.1.4 Quadratic Discriminant Analysis	73
3.1.5 Perceptron	74
3.1.6 Support Vector Machines + Kernels	76
3.1.7 Artificial Neural Networks + Back Propagation	80
3.1.8 Decision Trees	91
3.1.9 Bayes Optimal Classifier	106
3.10 Naive Bayes	108

Topic	Page No.
UNIT - IV	
4.1 Evaluation Measures	111
4.1.1 Hypothesis Testing	111
4.1.2 Ensemble Methods	113
4.1.3 Bagging	119
4.1.4 Adaboost/Gradient Boosting	120
4.1.5 Clustering	124
4.1.6 K-means	133
4.1.7 K-medoids	137
4.1.8 Density-based Hierarchical Spectral	140
UNIT - V	
5.1 Miscellaneous Topics	143
5.1.1 Expectation Maximization	143
5.1.2 Gmms	146
5.2 Learning Theory	151
5.3 Introduction to Reinforcement Learning Graphical Models	164
5.3.1 Bayesian Networks	173
5.4 Use Cases of Various ML Algorithms in Manufacturing, Retail, Transport, Healthcare, Weather, Insurance Sectors.	177

Important Questions

UNIT - I

1. Explain the types of probability with examples.

Ans :

Refer Unit-I, Q.No. 2

2. What are Independent Events in Probability? Explain, How to calculate the probability of independent events?

Ans :

Refer Unit-I, Q.No. 4

3. Define and prove Baye's theorem of probability.

Ans :

Refer Unit-I, Q.No. 6

4. What is the use of probability distribution? Explain its categories.

Ans :

Refer Unit-I, Q.No. 7

5. Write about Expected values and give an example for it.

Ans :

Refer Unit-I, Q.No.15

6. Explain about vectors and implementation of vectors with the help of some examples.

Ans :

Refer Unit-I, Q.No. 16

7. What is convex optimisation ? Explain how can we solve convex optimization problem.

Ans :

Refer Unit-I, Q.No. 21

8. Explain Bayesian Learning in terms of ML, MAP, Bayes Estimates and Conju-gate Priors.

Ans :

Refer Unit-I, Q.No. 42.

UNIT - II

1. **What is Regression Analysis? What are the various types of regressions used in regression analysis.**

Ans :

Refer Unit-II, Q.No. 1

2. **What is Regularization? How does Regularization Work?**

Ans :

Refer Unit-II, Q.No. 4

3. **Write about Dimensionality Reduction Technique.**

Ans :

Refer Unit-II, Q.No. 7

4. **What is PCA? Explain PCA algorithm.**

Ans :

Refer Unit-II, Q.No. 8

5. **How PLS techniques is used in linear decomposition? Expalin**

Ans :

Refer Unit-II, Q.No. 11

UNIT - III

1. **What is the Classification Algorithm in Machine Learning? Explain.**

Ans :

Refer Unit-III, Q.No. 1

2. **How can we use LDA technique for dimensionally Reduction.**

Ans :

Refer Unit-III, Q.No. 4

3. **Write about Quadratic Discriminant analysis.**

Ans :

Refer Unit-III, Q.No. 5

4. What is artificial neural network? Explain about ANN with its architecture.

Ans :

Refer Unit-III, Q.No. 9

5. What is Backpropagation? Explain about the working of backpropagation with example.

Ans :

Refer Unit-III, Q.No. 11

6. How does the Decision Tree algorithm Work?

Ans :

Refer Unit-III, Q.No. 13

7. What is the use of Bayes optimal classi-fier in the machine learning?

Ans :

Refer Unit-III, Q.No. 17

UNIT - IV

1. What is hypothesis testing? Explain about it with help of example.

Ans :

Refer Unit-IV, Q.No. 1

2. Write about different kinds of ensemble methods used in machine learning for predicting the data.

Ans :

Refer Unit-IV, Q.No. 2

3. What is Bagging technique? Why it is used for training data sets in ML.

Ans :

Refer Unit-IV, Q.No. 5

4. Explain Hierarchical Clustering in Machine Learning.

Ans :

Refer Unit-IV, Q.No. 10

5. What is K-Means Algorithm?How does the K-Means Algorithm Work?

Ans :

Refer Unit-IV, Q.No. 11

6. Explain DBSCAN Algorithm with example.

Ans :

Refer Unit-IV, Q.No. 13

UNIT - V

1. What is known as Expectation minimization. Explain EM algorithm.

Ans :

Refer Unit-V, Q.No. 1

2. Explain, How can we present EM algorithm using probabilistic model.

Ans :

Refer Unit-V, Q.No. 2

3. What Is Machine Learning? What are the various components in machine learning architecture?

Ans :

Refer Unit-V, Q.No. 5

4. What are Version spaces ? How can we use version spaces in representation of knowledge. Explain with example.

Ans :

Refer Unit-V, Q.No. 8

5. Give the brief introduction of Reinforcement Learning and working of Reinforcement Learning.

Ans :

Refer Unit-V, Q.No. 9

6. What are Bayesian Networks? How can we use Bayesian Networks in probability computations? Explain with an example.

Ans :

Refer Unit-V, Q.No. 12

7. Explain the machine learning algorithm in transport sector.

Ans :

Refer Unit-V, Q.No. 15

8. Explain the machine learning algorithm in insurance industry.

Ans :

Refer Unit-V, Q.No. 18

UNIT I

Basic Maths: Probability, Linear Algebra, Convex Optimization
Background: Statistical Decision Theory, Bayesian Learning (ML, MAP, Bayes estimates, Conjugate priors).

1.1 BASIC MATHS

1.1.1 Probability

Q1. Write about the probability and the terminology used for it.

Ans :

Probability

The meaning of probability is basically the extent to which something is likely to happen. This is the basic probability theory which is also used in the probability distribution, where you will learn the possibility of outcomes for a random experiment. To find the probability of a single event to occur, first we should know the total number of possible outcomes.

Definition - Experiment and Outcomes

In probabilities, an experiment is a process (could be "anything") in which there are one or more (usually more) possible outcomes each of which depends on chance.

Example

Here we show a couple of examples that show an *experiment* along with their *outcomes*.

1. For example rolling a dice is an experiment in which there are 6 possible outcomes:

1, 2, 3, 4, 5, 6

2. Another example of an experiment could be: picking a ball, at random, from a bag that contains 4 red balls and 6 green balls. In this experiment there would be two possible outcomes:

Red or Green

Definition - Sample Space

Given an experiment, the sample space consists of all the possible outcomes of the experiment.

The sample space is usually written, or illustrated, using one of the following:

- A list of all the possible outcomes written inside a set that we call U , or S .
- A sample space diagram, or
- A Venn Diagram.

Example

1. If we are flipping a coin the sample space would be:

$U = \{\text{Heads, Tails}\}$

2. If we're rolling a dice, then the *sample space* would be:

$U = \{1, 2, 3, 4, 5, 6\}$

Exercise 1

Write the sample space for each of the experiments, listed below.

1. Picking a ball at random from a bag that contains 3 green balls, 4 red balls and 3 yellow balls.
2. Spinning a wheel numbered 1, 2, 3, 4 and guessing the number on which it will stop spinning.
3. A class has 7 different nationalities in it: Swedish, American, English, French, Lebanese, Italian and Russian. A student is taken at random from the class and we have to guess his/her nationality.

Definition - Single Events

Given an experiment, along with its possible outcomes, an event is the name given to either one of the possible outcomes, or a group of outcomes.

Events are usually referred to using a capital letter, such as A, B, C,

The following show a couple of examples of single events.

Example 1

When we roll a single dice, we may define the event A as:

A: rolling an even number.

in which case there are three outcomes that correspond to event A:

2,4,6

Example 2

Another example could be when picking a ball at random from a bag containing 4 red balls and 6 green balls.

Definition - Combined Events

At times we'll be interested in the combination of two, or more, single events.

For instance, given two events A and B we may be interested in the event "A and B" or the event "A or B"

Example

When picking a card at random from a deck of 52 playing cards, we may be interested in the likelihood of picking a card that is both: even and red.

In this case we could define both events:

E: the number on the card is Even.

R: the card is Red.

We can then define the combined event:

"E and R": the number on the card is Even and the card is Red.

A probability is a number expressed as either:

- a Decimal
- a Fraction
- a Percentage

It's value is a measure of the likelihood of an event occurring.

Formula for Probability

The probability formula is defined as the possibility of an event to happen is equal to the ratio of the number of favourable outcomes and the total number of outcomes.

Probability of event to happen $P(E) = \frac{\text{Number of favourable outcomes}}{\text{Total Number of outcomes}}$

Sometimes students get mistaken for "favourable outcome" with "desirable outcome". This is the basic formula. But there are some more formulas for different situations or events.

Example

1. There is a container full of coloured bottles, red, blue, green and orange. Some of the bottles are picked out and displaced. Sumit did this 1000 times and got the following results:

- Number of blue bottles picked out: 300
- Number of red bottles : 200
- Number of green bottles : 450
- Number of orange bottles : 50

a) **What is the probability that Sumit will pick a green bottle?**

Ans:

For every 1000 bottles picked out, 450 are green.

Therefore, $P(\text{green}) = \frac{450}{1000} = 0.45$

b) **If there are 100 bottles in the container, how many of them are likely to be green?**

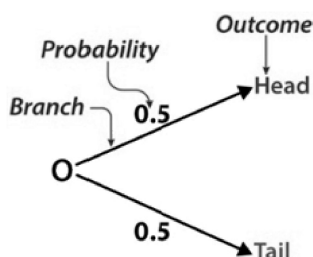
Ans:

The experiment implies that 450 out of 1000 bottles are green.

Therefore, out of 100 bottles, 45 are green.

Probability Tree

The tree diagram helps to organize and visualize the different possible outcomes. Branches and ends of the tree are two main positions. Probability of each branch is written on the branch, whereas the ends are containing the final outcome. Tree diagram used to figure out when to multiply and when to add. You can see below a tree diagram for the coin:



Notation

Given an event A, the probability of event A occurring is written: $p(A)$ Read: "the probability of event A"

The likelihood of an event A occurring is measured on a scale that goes from 0 to 1, where:

- 0 is the probability of something impossible.
- 1 is the probability of something certain.

All other events have a probability that lies somewhere in between these two values.

Q2. Explain the types of probability with examples.

Ans :

(Imp.)

Types of Probability

There are three major types of probabilities:

1. Theoretical Probability
2. Experimental Probability
3. Axiomatic Probability

1. Theoretical Probability

It is based on the possible chances of something to happen.

Theoretical probability is the theory behind probability. To find the probability of an event using theoretical probability, it is not required to conduct an experiment. Instead of that, we should know

about the situation to find the probability of an event occurring. The theoretical probability is defined as the ratio of the number of favourable outcomes to the number of possible outcomes.

For example, if a coin is tossed, the theoretical probability of getting head will be $\frac{1}{2}$.

Probability of Event $P(E) = \frac{\text{Number of Favourable outcomes}}{\text{Number of Possible outcomes}}$.

Example

Find the probability of rolling a 5 on a fair die

Sol:

To find the probability of getting 5 while rolling a die, an experiment is not needed. We know that, there are 6 possible outcomes when rolling a die. They are 1, 2, 3, 4, 5, 6.

Therefore, the probability is,

Probability of Event $P(E) = \frac{\text{No. of Favourable outcomes}}{\text{No. of Possible outcomes}}$.

$$P(E) = \frac{1}{6}$$

Hence, the probability of getting 5 while rolling a fair die is $\frac{1}{6}$.

2. Experimental Probability

It is based on the basis of the observations of an experiment. The experimental probability can be calculated based on the number of possible outcomes by the total number of trials. For example, if a coin is tossed 10 times and heads is recorded 6 times then, the experimental probability for heads is $\frac{6}{10}$ or, $\frac{3}{5}$.

The experimental Probability is defined as the ratio of the number of times that event occurs to the total number of trials.

Probability of Event $P(E) = \frac{\text{No. of. times that event occurs}}{\text{Total number of trials}}$

The basic difference between these two approaches is that in the experimental approach; the probability of an event is based on what has actually happened by conducting a series of actual experiments, while in theoretical approach; we attempt to predict what will occur without actually performing the experiments.

Example

If a coin is tossed 10 times, head appears 3 times. Find experimental probability of getting a head.

Sol.:

Experimental probability of getting a head = $3/10$

3. Axiomatic Probability

In axiomatic probability, a set of rules or axioms are set which applies to all types. These axioms are set by Kolmogorov and are known as Kolmogorov's three axioms. With the axiomatic approach to probability, the chances of occurrence or non-occurrence of the events can be quantified. The axiomatic probability lesson covers this concept in detail with Kolmogorov's three rules (axioms) along with various examples.

Conditional Probability is the likelihood of an event or outcome occurring based on the occurrence of a previous event or outcome.

Probability is a set function $P(E)$ that assigns to every event E a number called the "probability of E " such that:

1. The probability of an event is greater than or equal to zero

$$P(E) \geq 0$$

2. The probability of the sample space is one

$$P(\Omega) = 1$$

Axiomatic Probability Conditions

Let, S be the sample space of any random experiment and let P be the probability of occurrence of any event. Noting the characteristics of P , it should be a real valued function whose domain will be the power set of S and the range will lie in the interval $[0,1]$. This probability P will satisfy the following probability axioms:

1. For any event E , $P(E) \geq 0$
2. $P(S) = 1$
3. In case E and F are mutually exclusive events, then following equation will be valid:

$$P(E \cup F) = P(E) + P(F)$$

From point (3) it can be stated that $P(\phi) = 0$

If we need to prove this, let us take $F = \phi$ and make a note that

E and ϕ are disjoint events. Hence, from point (3) we can deduce that-

$$P(E \cup \phi) = P(E) + P(\phi) \text{ or}$$

$$P(E) = P(E) + P(\phi)$$

$$\text{i.e. } P(\phi) = 0$$

Let, the sample space of S contain the given outcomes $\delta_1, \delta_2, \delta_3, \dots, \delta_n$, then as per axiomatic definition of probability, we can deduce the following points:

1. $0 \leq P(\delta_i) \leq 1$ for each $\delta_i \in S$
2. $P(\delta_1) + P(\delta_2) + \dots + P(\delta_n) = 1$
3. For any event Q , $P(Q) = \sum P(\delta_i), \delta_i \in Q$.

This is to be noted that the singleton $\{\delta_i\}$ is known as elementary event and for the convenience of writing, we write $P(\delta_i)$ for $P(\{\delta_i\})$.

Example

On tossing a coin we say that the probability of occurrence of head and tail is $\frac{1}{2}$ each. Basically here we are assigning the probability value of $\frac{1}{2}$ for the occurrence of each event.

This condition basically satisfies both the conditions, i.e.

- Each value is neither less than zero nor greater than 1 and.
- Sum of the probabilities of occurrence of head and tail is 1.

Hence, for this case we can say that the probabilities of occurrence of head and tail are $\frac{1}{2}$ each.

Now, say $P(H) = \frac{1}{2}$ and $P(T) = \frac{1}{2}$

Does this probability value satisfy the conditions of axiomatic approach?

For this, let us again check the basic initial conditions of the axiomatic approach of probability.

- Each value is neither less than zero nor greater than 1 and
- Sum of the probabilities of occurrence of head and tail is 1.

Hence this sort of probability value assignment also satisfies the axiomatic approach of probability. Thus, we can conclude that there can be infinite ways to assign the probability to outcomes of an experiment.

Q3. Explain the probability rules with example problems.

Ans ;

Addition Rule 1

When two events, A and B, are mutually exclusive, the probability that A or B will occur is the sum of the probability of each event.

$$P(A \text{ or } B) = P(A) + P(B)$$

Let's use this addition rule to find the probability for Experiment 1.

Example 1

A single 6-sided die is rolled. What is the probability of rolling a 2 or a 5?

Probabilities

$$P(2) = \frac{1}{6}$$

$$P(5) = \frac{1}{6}$$

$$P(2 \text{ or } 5) = P(2) + P(5)$$

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

$$= \frac{2}{6}$$

$$= \frac{1}{3}$$

Example 2

A spinner has 4 equal sectors colored yellow, blue, green, and red. What is the probability of landing on red or blue after spinning this spinner?

Probabilities

$$P(\text{red}) = \frac{1}{4}$$

$$P(\text{blue}) = \frac{1}{4}$$

$$P(\text{red or blue}) = P(\text{red}) + P(\text{blue})$$

$$= \frac{1}{4} + \frac{1}{4}$$

$$= \frac{2}{4}$$

$$= \frac{1}{2}$$

Example 3

A glass jar contains 1 red, 3 green, 2 blue, and 4 yellow marbles. If a single marble is chosen at random from the jar, what is the probability that it is yellow or green?

Sol :

Probabilities

$$P(\text{red}) = \frac{4}{10}$$

$$P(\text{blue}) = \frac{3}{10}$$

$$P(\text{yellow or green}) = P(\text{yellow}) + P(\text{green})$$

$$= \frac{4}{10} + \frac{3}{10}$$

$$= \frac{7}{10}$$

Additional Rule 2

When two events, A and B, are non-mutually exclusive, the probability that A or B will occur is:

$$P(A \text{ or } B) = P(A) + P(B) - P(A \text{ and } B)$$

In the rule above, $P(A \text{ and } B)$ refers to the overlap of the two events. Let's apply this rule to some other experiments.

Example 4

In a math class of 30 students, 17 are boys and 13 are girls. On a unit test, 4 boys and 5 girls made an A grade. If a student is chosen at random from the class, what is the probability of choosing a girl or an A student?

Sol :

Probabilities

$$P(\text{girl or A}) = P(\text{girl}) + P(A) - P(\text{girl and A})$$

$$= \frac{13}{30} + \frac{9}{30} - \frac{5}{30}$$

$$= \frac{17}{30}$$

Rule of Subtraction

The probability that event A will occur is equal to 1 minus the probability that event A will not occur.

$$P(A) = 1 - P(A')$$

Example 5

The probability that Bill will graduate from college is 0.80. What is the probability that Bill will not graduate from college? Based on the rule of subtraction,

$$P(A) = 0.80$$

The probability that Bill will not graduate is $1.00 - 0.80$ or 0.20 .

Multiplication Rule

If A and B are two independent events in a probability experiment, then the probability that both events occur simultaneously is:

$$P(A \text{ and } B) = P(A) \cdot P(B)$$

In case of dependent events, the probability that both events occur simultaneously is:

$$P(A \text{ and } B) = P(A) \cdot P(B|A)$$

(The notation $P(B|A)$ means "the probability of B, given that A has happened.").

Example 6

Suppose you take out two cards from a standard pack of cards one after another, without replacing the first card. What is probability that the first card is the ace of spades, and the second card is a heart?

Sol :

The two events are dependent events because the first card is not replaced.

There is only one ace of spades in a deck of 52 cards. So:

$$P(\text{1st card is the ace of spades}) = \frac{1}{52}$$

If the ace of spaces is drawn first, then there are 51 cards left in the deck, of which 13 are hearts:

$$P(\text{2nd card is a heart} | \text{1st card is the ace of spades}) = \frac{13}{51}$$

So, by the multiplication rule of probability, we have:

$$P(\text{ace of spades, then a heart}) = \frac{1}{52} \cdot \frac{13}{51} = \frac{13}{52 \cdot 51} = \frac{1}{204}$$

Example 7

An urn contains 6 red marbles and 4 black marbles. Two marbles are drawn without replacement from the urn. What is the probability that both of the marbles are black?

Sol :

Let A = the event that the first marble is black; and let B = the event that the second marble is black. We know the following:

- In the beginning, there are 10 marbles in the urn, 4 of which are black. Therefore, $P(A) = 4/10$.
- After the first selection, there are 9 marbles in the urn, 3 of which are black. Therefore, $P(B|A) = 3/9$.

Therefore, based on the rule of multiplication:

$$P(A \cap B) = P(A) P(B|A)$$

$$P(A \cap B) = (4/10) * (3/9) \\ = 12/90 = 2/15 = 0.133$$

Q4. What are Independent Events in Probability? Explain, How to calculate the probability of independent events?

Ans : (Imp.)

Events are independent if the outcome of one event does not affect the outcome of another. For example, if you throw a die and a coin, the number on the die does not affect whether the result you get on the coin.

If A and B are independent events, then the probability of A happening AND the probability of B happening is $P(A) \times P(B)$.

The following gives the multiplication rule to find the probability of independent events occurring together. Scroll down the page for more examples and solutions of word problems that involve the probability of independent events.

Example 1

If a dice is thrown twice, find the probability of getting two 5's.

Sol :

$$P(\text{getting a 5 on the first throw}) = \frac{1}{6}$$

$$P(\text{getting a 5 on the second throw}) = \frac{1}{6}$$

$$P(\text{two 5's}) = \frac{1}{6} \times \frac{1}{6} = \frac{1}{36}$$

Example 2

Two sets of cards with a letter on each card as follows are placed into separate bags.

Sol :

Bag 1:

I	L	J	A	U
---	---	---	---	---

Bag 2:

L	R	H	E	C	A
---	---	---	---	---	---

Example

Sara randomly picked one card from each bag. Find the probability that:

- a) She picked the letters 'J' and 'R'.
- b) Both letters are 'L'.
- c) Both letters are vowels.

Sol :

$$\begin{aligned} \text{a) Probability that she picked J and R} \\ = \frac{1}{5} \times \frac{1}{6} = \frac{1}{30} \end{aligned}$$

$$\begin{aligned} \text{b) Probability that both letters are L} \\ = \frac{1}{5} \times \frac{1}{6} = \frac{1}{30} \end{aligned}$$

$$\begin{aligned} \text{c) Probability that both letters are vowels} \\ = \frac{3}{5} \times \frac{2}{6} = \frac{3}{15} = \frac{1}{5} \end{aligned}$$

Example 3

Two fair dice, one colored white and one colored red, are thrown. Find the probability that:

- the score on the red die is 2 and white die is 5.
- the score on the white die is 1 and red die is even.

Sol :

- Probability of the red die shows 2 and white die 5 = $\frac{1}{6} \times \frac{1}{6} = \frac{1}{36}$.
- Probability of the white die shows 1 and red die shows an even number = $\frac{3}{5} \times \frac{2}{6} = \frac{3}{15} = \frac{1}{5}$

Q5. What are known as dependent event. Explain with an example.

Ans :

Events are dependent if the outcome of one event affects the outcome of another. For example, if you draw two coloured balls from a bag and the first ball is not replaced before you draw the second ball then the outcome of the second draw will be affected by the outcome of the first draw.

If A and B are dependent events, then the probability of A happening AND the probability of B happening, given A, is $P(A) \times P(B \text{ after } A)$.

$$P(A \text{ and } B) = P(A) \times P(B \text{ after } A)$$

$P(B \text{ after } A)$ can also be written as $P(B|A)$ then $P(A \text{ and } B) = P(A) \times P(B|A)$.

Example 1

A purse contains four \$5 bills, five \$10 bills and three \$20 bills. Two bills are selected without the first selection being replaced. Find $P(\$5, \text{ then } \$5)$.

Sol :

There are four \$5 bills.

There are a total of twelve bills.

$$P(\$5) =$$

The result of the first draw affected the probability of the second draw.

There are three \$5 bills left.

There are a total of eleven bills left.

$$P(\$5 \text{ after } \$5) =$$

$$P(\$5, \text{ then } \$5) =$$

$$P(\$5) \cdot P(\$5 \text{ after } \$5) =$$

The probability of drawing a \$5 bill and then a \$5 bill is

Example 2

A bag contains 6 red, 5 blue and 4 yellow marbles. Two marbles are drawn, but the first marble drawn is not replaced.

- Find $P(\text{red, then blue})$
- Find $P(\text{blue, then blue})$

Sol :

- There are 6 red marbles.

There are a total of 15 marbles.

$$P(\text{red}) = \frac{6}{15}$$

The result of the first draw affected the probability of the second draw.

There are 5 blue marbles.

There are a total of 14 marbles left.

$$P(\text{blue after red}) = \frac{5}{14}$$

$$P(\text{red, then blue}) = P(\text{red}) \cdot P(\text{blue after red})$$

$$= \frac{6}{15} \times \frac{5}{14} = \frac{1}{7}$$

The probability of drawing a red marble and then a blue marble is $\frac{1}{7}$

- There are 5 blue marbles.

There are a total of 15 marbles.

$$P(\text{blue}) = \frac{5}{15}$$

The result of the first draw affected the probability of the second draw.

There are 4 blue marbles left.

There are a total of 14 marbles left.

$$P(\text{blue after blue}) = \frac{4}{14}$$

$$P(\text{blue, then blue}) = P(\text{blue}) \cdot P(\text{blue after blue}) = \frac{5}{15} \times \frac{4}{14} = \frac{2}{21}$$

The probability of drawing a red marble and then a blue marble is $\frac{2}{21}$.

Q6. Define and prove Baye's theorem of probability.

Ans :

(Imp.)

Bayes' theorem describes the probability of occurrence of an event related to any condition. It is also considered for the case of conditional probability. For example: if we have to calculate the probability of taking a blue ball from the second bag out of three different bags of balls, where each bag contains three different colour balls viz. red, blue, black. Such a case where the probability of occurrence of an event is calculated depending on other conditions is known as conditional probability.

The formula is:

$$P(A|B) = P(A) P(B|A)P(B)$$

How often A happens given that B happens, written **P(A|B)**,

How often B happens given that A happens, written **P(B|A)**

And how likely A is on its own, written **P(A)**

And how likely B is on its own, written **P(B)**

Bayes Theorem Proof

Statement

Let E_1, E_2, \dots, E_n be a set of events associated with a sample space S , where all the events E_i ,

E_2, \dots, E_n have nonzero probability of occurrence and they form a partition of S . Let A be any event associated with S , then according to Bayes theorem,

$$P(E_i | A) = \frac{P(E_i) P(A | E_i)}{\sum_{k=1}^n P(E_k) P(A | E_k)}$$

for any $k = 1, 2, 3, \dots, n$

Proof:

According to conditional probability formula,

$$P(E_i | A) = \frac{P(E_i \cap A)}{P(A)} \quad \dots (1)$$

Using multiplication rule of probability,

$$P(E_i \cap A) = P(E_i) P(A | E_i) \quad \dots (2)$$

Using total probability theorem,

$$P(A) = \sum_{k=1}^n P(E_k) P(A | E_k) \quad \dots (3)$$

Putting the values for equations (2) and (3) in equation 1, we get

$$P(E_i | A) = \frac{P(E_i) P(A | E_i)}{\sum_{k=1}^n P(E_k) P(A | E_k)}$$

Putting the values from equations (2) and (3) in equation 1, we get

$$P(E_i | A) = \frac{P(E_i) P(A | E_i)}{\sum_{k=1}^n P(E_k) P(A | E_k)}$$

Bayes Theorem Formula

If A and B are two events, then the formula for Bayes theorem is given by:

$$P(A|B) = N(A \cap B) / N(B); N(B) \neq 0$$

Where $P(A|B)$ is the probability of condition when event A is occurring while event B has already occurred.

$N(A \cap B)$ is the number of factors common to both A and B.

$N(B)$ is the number of factors in B

Examples and Solutions

Some illustrations will improve the understanding of the concept.

Example 1:

Bag I contains 4 white and 6 black balls while another Bag II contains 4 white and 3 black balls. One ball is drawn at random from one of the bags and it is found to be black. Find the probability that it was drawn from Bag I.

Sol:

Let E_1 be the event of choosing the bag I, E_2 the event of choosing the bag II and A be the event of drawing a black ball.

$$\text{Then, } P(E_1) = P(E_2) = 1/2$$

Also, $P(A|E_1) = P(\text{drawing a black ball from Bag I}) = 6/10 = 3/5$.

$P(A|E_2) = P(\text{drawing a black ball from Bag II}) = 3/7$.

By using Bayes' theorem, the probability of drawing a black ball from bag I out of two bags,

$$P(E_1|A) = \frac{P(E_1)P(A|E_1)}{P(E_1)P(A|E_1) + P(E_2)P(A|E_2)}$$

$$= \frac{\frac{1}{2} \times \frac{3}{5}}{\frac{1}{2} \times \frac{3}{5} + \frac{1}{2} \times \frac{3}{7}} = \frac{7}{12}$$

Example 2

A man is known to speak truth 2 out of 3 times. He throws a die and reports that number obtained is a four. Find the probability that the number obtained is actually a four.

Sol:

Let A be the event that the man reports that number four is obtained.

Let E_1 be the event that four is obtained and E_2 be its complementary event.

Then,

$$P(E_1) = \text{Probability that four occurs} = 1/6$$

$$P(E_2) = \text{Probability that four does not occurs}$$

$$= 1 - P(E_1)$$

$$= 1 - 1/6 = 5/6$$

Also, $P(A|E_1) = \text{Probability that man reports four and it is actually a four} = 2/3$.

$P(A|E_2) = \text{Probability that man reports four and it is not a four} = 1/3$.

By using Bayes' theorem, probability that number obtained is actually a four,

Q7. What is the use of probability distribution? Explain its categories.

Ans:

(Imp.)

The probability distribution gives the possibility of each outcome of a random experiment or events. It provides the probabilities of different possible occurrence. To recall, the probability is a measure of uncertainty of various phenomenon. Like, if you throw a dice, what is the possible outcomes of it, is defined by the probability.

Probability Distribution of Random Variables

A random variable has a probability distribution, which defines the probability of its unknown values. Random variables can be discrete (not constant) or continuous or both. That means it takes any of a designated finite or countable list of values, provided with a probability mass function feature of the random variable's probability distribution or can take any numerical value in an interval or set of intervals, through a probability density function that is representative of the random variable's probability distribution or it can be a combination of both discrete and continuous.

Two random variables with equal probability distribution can yet vary with respect to their relationships with other random variables or whether they are independent of these. The recognition of a random variable, which means, the outcomes of randomly choosing values as per the variable's probability distribution function, are called random variates.

Types of Probability Distribution

There are basically two types of probability distribution which are used for different purposes and various types of the data generation process.

1. Normal or Cumulative Probability Distribution
2. Binomial or Discrete Probability Distribution

1. Normal or Cumulative/Continuous Probability Distribution

This is also known as a continuous probability distribution or cumulative probability distribution. In this distribution, the set of possible outcomes can take on values on a continuous range.

For example, a set of real numbers, is a continuous or normal distribution, as it gives all the possible outcomes of real numbers. Similarly, set of complex numbers, set of a prime number, set of whole numbers etc are the examples of Normal Probability distribution. Also, in real-life scenarios, the temperature of the day is an example of continuous probability. Based on these outcomes we can create a distribution table. It is described by a probability density function. The formula for the normal distribution is;

$$P(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot e^{-(x-\mu)^2/2\sigma^2}$$

Where,

- μ = Mean Value
- σ = Standard Distribution of probability.
- If mean(μ) = 0 and standard deviation(σ) = 1, then this distribution is known to be normal distribution.
- x = Normal random variable

Normal Distribution Examples

Since the normal distribution statistics estimates many natural events so well, it has evolved into a standard of recommendation for many probability queries. Some of the examples are:

- Height of the Population of the world
- Rolling a dice (once or multiple times)

- To judge Intelligent Quotient Level of children in this competitive world
- Tossing a coin
- Income distribution in countries economy among poor and rich
- The sizes of females shoes
- Weight of newly born babies range
- Average report of Students based on their performance

2. Binomial/Discrete Probability Distribution

This distribution is also called a discrete probability distribution, where the set of outcomes are discrete in nature.

For example, if a dice is rolled, then all the possible outcomes are discrete and give a mass of outcomes. This is also known as probability mass functions.

So, the outcomes of binomial distribution consist of n repeated trials and the outcome may or may not occur. The formula for the binomial distribution is;

$$P(x) = \frac{n!}{r!(n-r)!} \cdot p^r (1-p)^{n-r}$$

$$P(x) = C(n, r) \cdot p^r (1-p)^{n-r}$$

Where,

- n = Total number of events
- r = Total number of successful events.
- p = Success on a single trial probability.
- ${}^nC_r = [n!/r!(n-r)!]$
- $1 - p$ = Failure Probability

Binomial Distribution Examples

As we already know, binomial distribution gives the possibility of a different set of outcomes. In the real-life the concept is used for:

- To find the number of used and unused materials while manufacturing a product
- To take a survey of positive and negative feedback from the people for anything.

- To check if a particular channel is watched by how many viewers by calculating the survey of YES/NO.
- Number of men and women working in a company
- To count the votes for a candidate in an election and many more.

Q8. What is Poisson Probability Distribution? Explain.

Ans :

The Poisson probability distribution is a discrete probability distribution that represents the probability of a given number of events happening in a fixed period of time or space if these cases occur with a known steady rate and individually of the time since the last event. Some of the real-life examples are:

- A number of patients arriving at a clinic between 10 to 11 AM.
- Number of emails received by a manager between the office hours
- The number of apples sold by a shopkeeper in the time period of 12 pm to 4 pm daily.

Probability Distribution Function

A function which is used to define the distribution of a probability is called a Probability distribution function. Depending upon the types, we can define these functions. Also, these functions are used in terms of probability density functions for any given random variable.

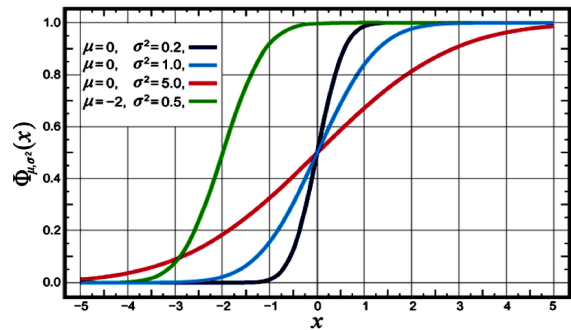
In the case of Normal distribution, the function of a real-valued random variable X is the function given by;

$$F_x(x) = P(X \leq x)$$

Where P shows the probability that the random variable X occurs on less than or equal to the value of x .

For a closed interval, $(a \rightarrow b)$, the cumulative probability function can be defined as;

$$P(a < X \leq b) = F_x(b) - F_x(a)$$



If we express, the cumulative probability function as integral of its probability density function f_x , then,

$$F_x(x) = \int_x^{-\infty} f_x(t) dt$$

In the case of a random variable $X=b$, we can define cumulative probability function as;

$$P(X=b) = F_x(b) - \lim_{x \rightarrow b} f_x(t)$$

In the case of Binomial distribution, as we know it is defined as the probability of mass or discrete random variable gives exactly some value. This distribution is also called probability mass distribution and the function associated with it is called a probability mass function.

Probability mass function is basically defined for scalar or multivariate random variables whose domain is variant or discrete. Let us discuss its formula:

Suppose a random variable X and sample space S is defined as;

$$X : S \rightarrow A$$

And $A \in R$, where R is a discrete random variable.

Then the probability mass function $f_x : A \rightarrow [0,1]$ for X can be defined as;

$$f_x(x) = P_r(X=x) = P(\{s \in S : X(s) = x\})$$

Probability Distribution Table

The table could be created on the basis of a random variable and possible outcomes. Say, a random variable X is a real-valued function whose domain is the sample space of a random experiment. The probability distribution $P(X)$ of a random variable X is the system of numbers.

X	X_1	X_2	X_3	X_n
P(X)	P_1	P_2	P_3	P_n

where $P_i > 0$, $i = 1$ to n and $P_1 + P_2 + P_3 + \dots + P_n = 1$.

Q9. Explain about Random variables, and how it is used in different context.

Ans :

A random variable is a rule that assigns a numerical value to each outcome in a sample space. Random variables may be either discrete or continuous. A random variable is said to be discrete if it assumes only specified values in an interval. Otherwise, it is continuous. When X takes values 1, 2, 3, ..., it is said to have a discrete random variable.

As a function, a random variable is needed to be measured, which allows probabilities to be assigned to a set of potential values. It is obvious that the results depend on some physical variables which are not predictable. Say, when we toss a fair coin, the final result of happening to be heads or tails will depend on the possible physical conditions.

As discussed there are two random variables, such as:

1. Discrete Random Variable
2. Continuous Random Variable

Variate

A variate can be defined as a generalization of the random variable. It has the same properties as that of the random variables without stressing to any particular type of probabilistic experiment. It always obeys a particular probabilistic law.

- A variate is called discrete variate when that variate is not capable of assuming all the values in the provided range.
- If the variate is able to assume all the numerical values provided in the whole range, then it is called continuous variate.

1. Discrete Random Variable

A discrete random variable can take only a finite number of distinct values such as 0, 1, 2, 3, 4, ... and so on. The probability distribution of a random variable has a list of probabilities compared with each of its possible values known as probability mass function.

In an analysis, let a person be chosen at random, and the person's height is demonstrated by a random variable. Logically the random variable is described as a function which relates the person to the person's height. Now in relation with the random variable, it is a probability distribution that enables the calculation of the probability that the height is in any subset of likely values, such as the likelihood that the height is between 175 and 185 cms, or the possibility that the height is either less than 45 or more than 180 cm. Now another random variable could be the person's age which could be either between 45 years to 50 years or less than 40 or more than 50.

2. Continuous Random Variable

A numerically valued variable is said to be continuous if, in any unit of measurement, whenever it can take on the values a and b . If the random variable X can assume an infinite and uncountable set of values, it is said to be a continuous random variable. When X takes any value in a given interval (a, b) , it is said to be a continuous random variable in that interval.

Formally, a continuous random variable is such whose cumulative distribution function is constant throughout. There are no "gaps" in between which would compare to numbers which have a limited probability of occurring. Alternately, these variables almost never take an accurately prescribed value c but there is a positive probability that its value will rest in particular intervals which can be very small.

Random Variable Formula

For a given set of data the mean and variance random variable is calculated by the formula. So, here we will define two major formulas:

- Mean of random variable
- Variance of random variable

Mean of random variable

If X is the random variable and P is the respective probabilities, the mean of a random variable is defined by:

$$\text{Mean } (\mu) = \sum XP$$

where variable X consist of all possible values and P consist of respective probabilities.

Variance of Random Variable

The variance tells how much is the spread of random variable X around the mean value. The formula for the variance of a random variable is given by;

$$\text{Var}(X) = \sigma^2 = E(X^2) - [E(X)]^2$$

where $E(X^2) = \sum X^2 P$ and $E(X) = \sum X P$

Functions of Random Variables

Let the random variable X assume the values x_1, x_2, \dots with corresponding probability $P(x_1), P(x_2), \dots$ then the expected value of the random variable is given by:

Expectation of X , $E(X) = \sum x P(x)$.

A new random variable Y can be stated by using a real Borel measurable function $g: R \rightarrow R$, to the results of a real-valued random variable X . That is, $Y = f(X)$. The cumulative distribution function of Y is then given by:

$$F_Y(y) = P(g(X) \leq y)$$

If function g is invertible (say $h = g^{-1}$) and is either increasing or decreasing, then the previous relationship can be extended to obtain:

$$F_Y(y) = P(g(X) \leq y) = \begin{cases} P(X \leq h(y)) = F_X(h(y)), & \text{if } h = g^{-1} \text{ increasing} \\ P(X \geq h(y)) = 1 - F_X(h(y)), & \text{if } h = g^{-1} \text{ decreasing} \end{cases}$$

Now if we differentiate both the sides of the above expressions with respect to y , then the relation between the probability density functions can be found:

$$f_Y(y) = f_X(h(y)) |dh(y)/dy|$$

Random Variable and Probability Distribution

The probability distribution of a random variable can be

- Theoretical listing of outcomes and probabilities of the outcomes.
- An experimental listing of outcomes associated with their observed relative frequencies.
- A subjective listing of outcomes associated with their subjective probabilities.

The probability of a random variable X which takes the values x is defined as a probability function of X is denoted by $f(x) = P(X = x)$

A probability distribution always satisfies two conditions:

- $f(x) \geq 0$
- $\sum f(x) = 1$

The important probability distributions are:

- Binomial distribution
- Poisson distribution
- Bernoulli's distribution
- Exponential distribution
- Normal distribution

Transformation of Random Variables

The transformation of a random variable means to reassign the value to another variable. The transformation is actually inserted to remap the number line from x to y , then the transformation function is $y = g(x)$.

Transformation of X or Expected Value of X for a Continuous Variable

Let the random variable X assume the values x_1, x_2, x_3, \dots with corresponding probability $P(x_1), P(x_2), P(x_3), \dots$ then the expected value of the random variable is given by

$$\text{Expectation of } X, E(X) = \int x P(x)$$

Random Variable Example

Find the mean value for the continuous random variable, $f(x) = x, 0 \leq x \leq 2$.

Sol:

Given: $f(x) = x, 0 \leq x \leq 2$.

The formula to find the mean value is

$$E(X) = \int_{-\infty}^{\infty} x f(x) dx \quad E(X) = \int_0^2 x f(x) dx \quad E(X) = \int_0^2 x \cdot x dx$$

$$E(X) = \int_{-\infty}^{\infty} x^2 dx \quad E(X) = \left(\frac{x^2}{3} \right)_0^2 \quad E(X) = \left(\frac{2^3}{3} \right) = \left(\frac{0^3}{3} \right) \quad E(X) = \left(\frac{8}{3} \right) - (0) \quad E(X) = \frac{8}{3}$$

Therefore, the mean of the continuous random variable, $E(X) = 8/3$.

Q10. Write about Expected values and give an example for it.

Ans:

(Imp.)

In a probability distribution, the weighted average of possible values of a random variable, with weights given by their respective theoretical probabilities, is known as the expected value, usually represented by $E(x)$.

The expected value informs about what to expect in an experiment "in the long run", after many trials. In most of the cases, there could be no such value in the sample space.

The weighted average formula for expected value is given by multiplying each possible value for the random variable by the probability that the random variable takes that value, and summing all these products. It can be written as

$$E(x) = \sum x_i P(x_i)$$

where x_i covers all possible values for the random variable, and $P(x_i)$ is the respective theoretical probability.

$E(x)$ is also called as mean of the probability distribution because it tells what to expect in the "long run" - that is, after many trials.

Example

When you roll a die, you will be paid \$1 for odd number and \$2 for even number. Find the expected value of money you get for one roll of the die.

The sample space of the experiment is $\{1,2,3,4,5,6\}$.

The table illustrates the probability distribution for a single roll of a die and the amount that will be paid for each outcome.

Roll (x)	1	2	3	4	5	6
Pr obability	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$
Amount (\$)	1	2	1	2	1	2

Use the weighted average formula.

$$\begin{aligned}
 E(x) &= 1\left(\frac{1}{6}\right) + 2\left(\frac{1}{6}\right) + 1\left(\frac{1}{6}\right) + 2\left(\frac{1}{6}\right) + 1\left(\frac{1}{6}\right) + 2\left(\frac{1}{6}\right) \\
 &= \frac{1}{6} + \frac{2}{6} + \frac{1}{6} + \frac{2}{6} + \frac{1}{6} + \frac{2}{6} \\
 &= \frac{9}{6} \\
 &= 1.5
 \end{aligned}$$

So, the expected value is \$1.50 . In other words, on average, you get \$1.50 per roll.

Q11. Write about Binomial Distribution with an example.

Ans :

A binomial distribution can be thought of as simply the probability of a SUCCESS or FAILURE outcome in an experiment or survey that is repeated multiple times. The binomial is a type of distribution that has two possible outcomes (the prefix “bi” means two, or twice). For example, a coin toss has only two possible outcomes: heads or tails and taking a test could have two possible outcomes: pass or fail.

The probability distribution becomes a binomial probability distribution when it meets the following requirements.

1. Each trail can have only two outcomes or the outcomes that can be reduced to two outcomes. These outcomes can be either a success or a failure.
2. The trails must be a fixed number.
3. The outcome of each trial must be independent of each other.
4. And the success of probability must remain the same for each trail.

Formula for Binomial Distribution in Probability

The formula for the binomial probability distribution is as stated below:

Binomial Distribution Formula	
Binomial Distribution	$P(x) = {}^nC_r \cdot p^r (1 - p)^{n-r}$
Or,	$P(x) = [n!/r!(n-r)!] \cdot p^r (1-p)^{n-r}$

Where,

- n = Total number of events
- r = Total number of successful events.
- p = Probability of success on a single trial.
- ${}^nC_r = [n!/(r!(n-r)!)]$
- $1 - p$ = Probability of failure.

Example

Toss a coin for 12 times. What is the probability of getting exactly 7 heads?

Sol:

Number of trials (n) = 12

Number of success (r) = 7

probability of single trail (p) = $\frac{1}{2} = 0.5$

$${}^nC_r = [n!/(r! \times (n-r)!)]$$

$$= 12! / 7!(12 - 7)!$$

$$= 12! / 7! 5!$$

$$= 95040120$$

$$= 792$$

$$p^r = 0.5^7 = 0.0078125$$

To Find $(1-p)^{n-r}$, calculate $(1-p)$ and $(n-r)$.

$$1 - p = 1 - 0.5 = 0.5$$

$$n - r = 12 - 7 = 5$$

$$(1-p)^{n-r} = 0.5^5 = 0.03125$$

$$\text{Solve } P(X = r) = {}^nC_r \cdot p^r \cdot (1-p)^{n-r}$$

$$= 792 \times 0.0078125 \times 0.03125$$

$$= 0.193359375$$

The probability of getting exactly 7 heads is 0.19.

Q12. What is binomial probability. Explain with example.

Ans:

Binomial Probability Formula

The binomial probability formula can calculate the probability of success for binomial

distributions. Binomial probability distribution along with normal probability distribution are the two probability distribution types. To recall, the binomial distribution is a type of distribution in statistics that has two possible outcomes. For instance, if you toss a coin and there are only two possible outcomes: heads or tails. In the same way, taking a test could have two possible outcomes: pass or fail. The Binomial Probability distribution is an experiment that possesses the following properties:

- There are a fixed number of trials which is denoted by n
- All the trials are independent.
- The outcome of each trial can either be a "success" or "failure".
- The probability of success remains constant and is denoted by p .

Binomial Probability Formula

The Binomial Probability distribution of exactly x successes from n number of trials is given by the below formula-

$$P(X) = {}^nC_x \cdot p^x \cdot q^{n-x}$$

Where,

- n = Total number of trials
- x = Total number of successful trials
- p = probability of success in a single trial
- q = probability of failure in a single trial = $1-p$

Example 1

A coin is flipped 6 times. What is the probability of getting exactly 2 tails?

Sol:

n = total number of trials = 6

x = total number of successful trials = 3

p = probability of success in one trial = $\frac{1}{2}$

q = probability of failure in one trial

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

$$P(X) = {}^nC_x \cdot p^x \cdot q^{n-x}$$

$$P(x) = [6!/(2! \times 4!)] \times (\frac{1}{2})^3 \times (\frac{1}{2})^{6-3}$$

$$= [6!/(2! \times 4!)] \times (\frac{1}{2})^3 \times (\frac{1}{2})^3$$

$$= 15/64$$

$$= 0.234$$

13. Explain normal distribution with an example.*Ans :*

The Normal Distribution is defined by the probability density function for a continuous random variable in a system. Let us say, $f(x)$ is the probability density function and X is the random variable. Hence, it defines a function which is integrated between the range or interval $(x \text{ to } x + dx)$, giving the probability of random variable X , by considering the values between x and $x + dx$.

$$f(x) \geq 0 \quad \forall \in (-\infty, +\infty)$$

$$\text{And } \int_{-\infty}^{+\infty} f(x) = 1$$

Normal Distribution Formula

The probability density function of normal or gaussian distribution is given by;

$$f(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Where,

- x is the variable
- μ is the mean
- σ is the standard deviation

Normal Distribution Curve

The random variables following the normal distribution are those whose values can find any unknown value in a given range. For example, finding the height of the students in the school. Here, the distribution can consider any value, but it will be bounded in the range say, 0 to 6ft. This limitation is forced physically in our query.

Whereas, the normal distribution doesn't even bother about the range. The range can also extend to $-\infty$ to $+\infty$ and still we can find a smooth curve. These random variables are called Continuous Variables, and the Normal Distribution then provides here probability of the value lying in a particular range for a given experiment. Also, use the normal distribution calculator to find the probability density function by just providing the mean and standard deviation value.

Normal Distribution Table

The table here shows the area from 0 to Z-value.

Z-Value	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.0000	0.0040	0.0080	0.0120	0.0160	0.0199	0.0239	0.0279	0.0319	0.0359
0.1	0.0398	0.0438	0.0478	0.0517	0.0557	0.0596	0.0636	0.0675	0.0714	0.0753
0.2	0.0793	0.0832	0.0871	0.0910	0.0948	0.0987	0.1026	0.1064	0.1103	0.1141
0.3	0.1179	0.1217	0.1255	0.1293	0.1331	0.1368	0.1406	0.1443	0.1480	0.1517
0.4	0.1554	0.1591	0.1628	0.1664	0.1700	0.1736	0.1772	0.1808	0.1844	0.1879
0.5	0.1915	0.1950	0.1985	0.2019	0.2054	0.2088	0.2123	0.2157	0.2190	0.2224

0.6	0.2257	0.2291	0.2324	0.2357	0.2389	0.2422	0.2454	0.2486	0.2517	0.2549
0.7	0.2580	0.2611	0.2642	0.2673	0.2704	0.2734	0.2764	0.2794	0.2823	0.2852
0.8	0.2881	0.2910	0.2939	0.2967	0.2995	0.3023	0.3051	0.3078	0.3106	0.3133
0.9	0.3159	0.3186	0.3212	0.3238	0.3264	0.3289	0.3315	0.3340	0.3365	0.3389
1.0	0.3413	0.3438	0.3461	0.3485	0.3508	0.3531	0.3554	0.3577	0.3599	0.3621
1.1	0.3643	0.3665	0.3686	0.3708	0.3729	0.3749	0.3770	0.3790	0.3810	0.3830
1.2	0.3849	0.3869	0.3888	0.3907	0.3925	0.3944	0.3962	0.3980	0.3997	0.4015
1.3	0.4032	0.4049	0.4066	0.4082	0.4099	0.4115	0.4131	0.4147	0.4162	0.4177
1.4	0.4192	0.4207	0.4222	0.4236	0.4251	0.4265	0.4279	0.4292	0.4306	0.4319
1.5	0.4332	0.4345	0.4357	0.4370	0.4382	0.4394	0.4406	0.4418	0.4429	0.4441
1.6	0.4452	0.4463	0.4474	0.4484	0.4495	0.4505	0.4515	0.4525	0.4535	0.4545
1.7	0.4554	0.4564	0.4573	0.4582	0.4591	0.4599	0.4608	0.4616	0.4625	0.4633
1.8	0.4641	0.4649	0.4656	0.4664	0.4671	0.4678	0.4686	0.4693	0.4699	0.4706
1.9	0.4713	0.4719	0.4726	0.4732	0.4738	0.4744	0.4750	0.4756	0.4761	0.4767
2.0	0.4772	0.4778	0.4783	0.4788	0.4793	0.4798	0.4803	0.4808	0.4812	0.4817
2.1	0.4821	0.4826	0.4830	0.4834	0.4838	0.4842	0.4846	0.4850	0.4854	0.4857
2.2	0.4861	0.4864	0.4868	0.4871	0.4875	0.4878	0.4881	0.4884	0.4887	0.4890
2.3	0.4893	0.4896	0.4898	0.4901	0.4904	0.4906	0.4909	0.4911	0.4913	0.4916
2.4	0.4918	0.4920	0.4922	0.4925	0.4927	0.4929	0.4931	0.4932	0.4934	0.4936
2.5	0.4938	0.4940	0.4941	0.4943	0.4945	0.4946	0.4948	0.4949	0.4951	0.4952
2.6	0.4953	0.4955	0.4956	0.4957	0.4959	0.4960	0.4961	0.4962	0.4963	0.4964
2.7	0.4965	0.4966	0.4967	0.4968	0.4969	0.4970	0.4971	0.4972	0.4973	0.4974
2.8	0.4974	0.4975	0.4976	0.4977	0.4977	0.4978	0.4979	0.4979	0.4980	0.4981
2.9	0.4981	0.4982	0.4982	0.4983	0.4984	0.4984	0.4985	0.4985	0.4986	0.4986
3.0	0.4987	0.4987	0.4987	0.4988	0.4988	0.4989	0.4989	0.4989	0.4990	0.4990

Example 1

Calculate the probability density function of normal distribution using the following data. $x = 3$, $\mu = 4$ and $\sigma = 2$.

Sol:

Given, variable, $x = 3$

Mean = 4 and

Standard deviation = 2

By the formula of the probability density of normal distribution, we can write;

$$f(3, 4, 2) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(3-4)^2}{2 \times 2^2}}$$

Hence, $f(3,4,2) = 1.106$.

Example 2

If the value of random variable is 2, mean is 5 and the standard deviation is 4, then find the probability density function of the gaussian distribution.

Sol.:

Given,

Variable, $x = 2$

Mean = 5 and

Standard deviation = 4

By the formula of the probability density of normal distribution, we can write;

$$f(3, 4, 2) = \frac{1}{4\sqrt{2\pi}} e^{\frac{-(2-5)^2}{2 \times 4^2}}$$

$$f(2, 2, 4) = 1/(4\sqrt{2\pi})e^0$$

$$f(2, 2, 4) = 0.0997.$$

Q14. Write about different types of random samplings.

Ans.:

Random sampling is a method of choosing a sample of observations from a population to make assumptions about the population. It is also called probability sampling. The counterpart of this sampling is Non-probability sampling or Non-random sampling. The primary types of this sampling are simple random sampling, stratified sampling, cluster sampling, and multistage sampling. In the sampling methods, samples which are not arbitrary are typically called convenience samples.

Type of Random Sampling

The random sampling method uses some manner of a random choice. In this method, all the suitable individuals have the possibility of choosing the sample from the whole sample space. It is a time consuming and expensive method. The advantage of using probability sampling is that it ensures the sample that should represent the population. There are four major types of this sampling method, they are;

1. Simple Random Sampling
2. Systematic Sampling
3. Stratified Sampling
4. Clustered Sampling

1. Simple random sampling

In this sampling method, each item in the population has an equal and likely possibility of getting selected in the sample (for example, each member in a group is marked with a specific number). Since the selection of item completely depends on the possibility, therefore this method is called "Method of chance Selection". Also, the sample size is large and the item is selected randomly, thus it is known as "Representative Sampling".

2. Systematic Random Sampling

In this method, the items are chosen from the destination population by choosing the random selecting point and picking the other methods after a fixed sample period. It is equal to the ratio of the total population size and the required population size.

3. Stratified Random Sampling

In this sampling method, a population is divided into subgroups to obtain a simple random sample from each group and complete the sampling process (for example, number of girls in a class of 50 strength). These small groups are called strata. The small group is created based on a few features in the population. After dividing the population into smaller groups, the researcher randomly selects the sample.

4. Clustered Sampling

Cluster sampling is similar to stratified sampling, besides the population is divided into a large number of subgroups (for example, hundreds of thousands of strata or subgroups). After that, some of these subgroups are chosen at random and simple random samples are then gathered within these subgroups. These subgroups are known as clusters. It is basically utilised to lessen the cost of data compilation.

Random Sampling Formula

If P is the probability, n is the sample size and N is the population. Then;

- The chance of getting a sample selected only once is given by;

$$P = 1 - (N-1/N) \cdot (N-2/N-1) \cdot \dots \cdot (N-n/N-(n-1))$$

$$\text{Cancelling} = 1 - (N-n/n)$$

$$P = n/N$$

- The chance of getting a sample selected more than once is given by;

$$P = 1 - (1 - (1/N))^n$$

Random Sampling Example

Suppose a firm has 1000 employees in which 100 of them have to be selected for onsite work. All their names will be put in a basket to pull 100 names out of those. Now, each employee has an equal chance of getting selected, so we can also easily calculate the probability (P) of a given employee being selected, since we know the sample size (n) and the population size (N).

Therefore, the chance of selection of an employee only once is;

$$P = n/N = 100/1000 = 10\%$$

And the chance of selection of an employee more than once is;

$$P = 1 - (1 - (1/N))^n$$

$$P = 1 - (999/1000)^{100}$$

$$P = 0.952$$

$$P \approx .5\%$$

1.1.2 Linear Algebra

Q15. What is Linear Algebra ? How many ways we can define the linear algebra functions. Explain some linear algebra functions with examples.

Ans :

It is a branch of mathematics that allows to define and perform operations on higher-dimensional coordinates and plane interactions in a concise way. Linear Algebra is an algebra extension to an undefined number of dimensions. Linear

Algebra concerns the focus on linear equation systems. It is a continuous type of mathematics and is applicable in science and engineering, as it helps one to model and efficiently simulate natural phenomena.

Vectors, Matrices, and Tensors

In machine learning, the majority of data is most often represented as vectors, matrices or tensors. Therefore, the machine learning heavily relies on the linear algebra.

- A vector is a 1D array. For instance, a point in space can be defined as a vector of three coordinates (x, y, z). Usually, it is defined in such a way that it has both the magnitude and the direction.
- A matrix is a two-dimensional array of numbers, that has a fixed number of rows and columns. It contains a number at the intersection of each row and each column. A matrix is usually denoted by square brackets [].
- A tensor is a generalization of vectors and matrices. For instance, a tensor of dimension one is a vector. In addition, we can also have a tensor of two dimensions which is a matrix. Then, we can have a three-dimensional tensor such as the image with RGB colors. This continues to expand to four-dimensional tensors and so on.

1
2
3
4
5
6
7

Vector

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

Matrix

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

Tensor

Before progressing to Linear Algebra concepts, we must understand the below properties:

- **Associative Property:** It is a property in Mathematics which states that if a, b and c are mathematical objects then $a + (b + c) = (a + b) + c$ in which + is a binary operation.

- **Commutative Property:** It is a property in Mathematics which states that if a and b are mathematical objects then $a + b = b + a$ in which $+$ is a binary operation.
- **Distributive Property:** It is a property in Mathematics which states that if a , b and c are mathematical objects then $a * (b + c) = (a * b) + (a * c)$ in which $*$ and $+$ are binary operators.

Linear Algebra Equations

The general linear equation is represented as $a_1x_1 + a_2x_2 + \dots + a_nx_n = b$

Here,

a 's – represents the coefficients

x 's – represents the unknowns

b – represents the constant

There exists a system of linear algebraic equations, which is the set of equations. The system of equations can be solved using the matrices.

It obeys the linear function such as $(x_1, \dots, x_n) \rightarrow a_1x_1 + \dots + a_nx_n$

To solve linear equations we will make heavy use of the following facts.

1. If $a=b$ then $a+c=b+c$ for any c . All this is saying is that we can add a number, c , to both sides of the equation and not change the equation.
2. If $a=b$ then $a - c = b - c$ for any c . As with the last property we can subtract a number, c , from both sides of an equation.
3. If $a=b$ then $ac=bc$ for any c . Like addition and subtraction, we can multiply both sides of an equation by a number, c , without changing the equation.
4. If $a=b$ then $ac=bc$ for any non-zero c . We can divide both sides of an equation by a non-zero number, c , without changing the equation

Example 1

Solve each of the following equations.

$$3(x+5) = 2(-6-x)-2x$$

Sol:

For this problem there are no fractions so we don't need to worry about the first step in the process. The next step tells to simplify both sides. So, we will clear out any parenthesis by multiplying the numbers through and then combine like terms.

$$3(x+5) = 2(-6-x)-2x$$

$$3x+15 = -12-2x-2x$$

$$3x+15 = -12-4x$$

The next step is to get all the xx 's on one side and all the numbers on the other side. Which side the xx 's go on is up to you and will probably vary with the problem. As a rule of thumb, we will usually put the variables on the side that will give a positive coefficient. This is done simply because it is often easy to lose track of the minus sign on the coefficient and so if we make sure it is positive we won't need to worry about it.

So, for our case this will mean adding $4xx$ to both sides and subtracting 15 from both sides. Note as well that while we will actually put those operations in this time we normally do these operations in our head.

$$3x+15 = -12-4x$$

$$3x+15-15+4x = -12-4x+4x-15$$

$$7x = -27$$

The next step says to get a coefficient of 1 in front of the xx . In this case we can do this by dividing both sides by a 7.

$$\frac{7x}{7} = \frac{-27}{7}$$

$$x = -\frac{27}{7}$$

Now, if we've done all of our work correct $x = -27/7$ is the solution to the equation.

The last and final step is to then check the solution. As pointed out in the process outline we need to check the solution in the **original** equation.

This is important, because we may have made a mistake in the very first step and if we did and then checked the answer in the results from that step it may seem to indicate that the solution is correct when the reality will be that we don't have the correct answer because of the mistake that we originally made.

The problem of course is that, with this solution, that checking might be a little messy. Let's do it anyway.

$$3\left(-\frac{27}{7} + 5\right) = 2\left(-6 - \left(-\frac{27}{7}\right)\right) - 2\left(-\frac{27}{7}\right)$$

$$3\left(\frac{8}{7}\right) = 2\left(-\frac{15}{7}\right) + \frac{54}{7}$$

$$\frac{24}{7} = \frac{24}{7} \quad \text{OK}$$

So, we did our work correctly and the solution to the equation is,

$$x = -27/7$$

Vector Spaces

As we know that linear algebra deals with the study of vector spaces and the linear transformations between them. By the definition of vector, it is a physical quantity that has both magnitude and direction. A vector space is defined as the collection of objects called vectors, which may be added together and multiplied (i.e. scaled) by numbers, called scalars. Generally, real numbers are taken as scalars, but there exist vector spaces with scalar multiplication by non-real numbers, i.e. complex numbers, or naturally any field.

The operations such as vector addition and scalar multiplication must satisfy specific requirements, called vector axioms. Generally, the terms real vector space and complex vector space are used to define that the scalars are real or complex numbers, respectively.

Suppose V be any vector space with elements a, b, c and scalars m, n over a field F , then the vector axioms are given by:

- Commutative of addition: $a + b = b + a$
- Associativity of addition: $a + (b + c) = (a + b) + c$

- Additive identity: $a + 0 = 0 + a = a$, where 0 is an element in V called zero vector.

- Additive inverse: $a + (-a) + (-a) + a = 0$, $a, -a$ belongs to V .

These four axioms define that the vector space V is an abelian group under addition.

Other axioms include distributivity of scalar multiplication with respect to vector addition and field addition, identity element of scalar multiplication and so on.

For example,

$$m(a) = ma; n(a + b) = na + nb$$

An element of a specific vector space may have different characteristics. For example, the elements can be a sequence, a function, a polynomial or a matrix. Linear algebra is affected by those properties of such things that are common or familiar to all vector spaces.

A linear map can be written for two given vector spaces namely V and W over a field F . This is sometimes referred to as linear transformation or mapping of vector spaces. Thus, it is given by:

$$T: V \rightarrow W$$

This allows us to write the addition of scalar multiplication of elements such as:

$$T(a + b) = T(a) + T(b)$$

$$T(ma) = mT(a)$$

Linear Function

A linear function is an algebraic equation in which each term is either a constant or the product of a constant and a single independent variable of power 1. In linear algebra, vectors are taken while forming linear functions. Some of the examples of the kinds of vectors that can be rephrased in terms of the function of vectors.

Mathematically, linear function is defined as:

A function $L: R^n \rightarrow R^m$ is linear if

$$(i) \quad L(x + y) = L(x) + L(y)$$

$$(ii) \quad L(\alpha x) = \alpha L(x)$$

for all $x, y \in R^n, \alpha \in R$

Example

Show that the function $L : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ given by $L(\mathbf{x}) = \begin{bmatrix} x_1 + 4x_2 \\ 3x_1 - x_2 \\ x_2 \end{bmatrix}$ is linear.

Sol.:

For any $x, y \in \mathbb{R}^2$, we have

$$\begin{aligned} L(x + y) &= L\left(\begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \end{bmatrix}\right) \\ &= \begin{bmatrix} (x_1 + y_1) + 4(x_2 + y_2) \\ 3(x_1 + y_1) - (x_2 + y_2) \\ (x_2 + y_2) \end{bmatrix} \\ &= \begin{bmatrix} (x_1 + 4x_2) + (y_1 + 4y_2) \\ (3x_1 - x_2) + (3y_1 - y_2) \\ (x_2) + (y_2) \end{bmatrix} \\ &= \begin{bmatrix} x_1 + 4x_2 \\ 3x_1 - x_2 \\ x_2 \end{bmatrix} + \begin{bmatrix} y_1 + 4y_2 \\ 3y_1 - y_2 \\ y_2 \end{bmatrix} \\ &= L(x) + L(y) \end{aligned}$$

For any $x \in \mathbb{R}^2$ and $\alpha \in \mathbb{R}$, we have

$$\begin{aligned} L(\alpha x) &= L\left(\begin{bmatrix} \alpha x_1 \\ \alpha x_2 \end{bmatrix}\right) \\ &= \begin{bmatrix} (\alpha x_1) + 4(\alpha x_2) \\ 3(\alpha x_1) - (\alpha x_2) \\ (\alpha x_2) \end{bmatrix} \\ &= \begin{bmatrix} \alpha(x_1 + 4x_2) \\ \alpha(3x_1 - x_2) \\ \alpha(x_2) \end{bmatrix} \\ &= \alpha \begin{bmatrix} x_1 + 4x_2 \\ 3x_1 - x_2 \\ x_2 \end{bmatrix} \\ &= \alpha L(x) \end{aligned}$$

Therefore, L is a linear function.

Linear Algebra Matrix

Matrices are linear functions of a certain kind. Matrix is the result of organizing information related to certain linear functions. Matrix almost appears in linear algebra because it is the central information of linear algebra.

Mathematically, this relation can be defined as follows.

A is an $m \times n$ matrix, then we get a linear function $L : R^n \rightarrow R^m$ by defining

$$L(x) = Ax \text{ or } Ax = B$$

Go through the example given below to understand this mapping in detail.

Example

A room contains x bags and y boxes of fruits and each bag contain 2 apples and 4 bananas and each box contains 6 apples and 8 bananas. There are a total of 20 apples and 28 bananas in the room. Find the value of x and y .

Sol.:

Write the simultaneous equation for the given information that the above condition becomes true.

$$2x + 6y = 20$$

$$4x + 8y = 28$$

Here the example given above shows the system of linear equations.

Now, write the above equation as equality between 2-vectors and using the rules, we get

$$(2x + 6y \quad 4x + 8y) = (20 \quad 28) \times (2 \quad 4) + y(6 \quad 8) = (20 \quad 28)$$

We denote the functions as an array of numbers is called a matrix.

Therefore, the function $(2 \quad 4 \quad 6 \quad 8)$ is defined by

$$(2 \quad 4 \quad 6 \quad 8)(xy) = x(2 \quad 4) + y(6 \quad 8)$$

Numerical Linear Algebra

Numerical linear algebra is also known as the applied linear algebra. Applied linear algebra deals with the study of how matrix operations can be used to create computer algorithms, which helps to solve the problems in continuous mathematics with efficiency and accuracy. In numerical linear algebra, many matrix decomposition methods are used to find the solutions for common linear algebraic problems like least-square optimization, locating Eigenvalues, and solving systems of linear equations. Some of the matrix decomposition methods in numerical linear algebra include Eigen decomposition, Single value decomposition, QR factorization and so on.

Linear Algebra Applications

Here, some of the linear algebra applications are given as:

- **Ranking in Search Engines:** One of the most important applications of linear algebra is in the creation of Google. The most complicated ranking algorithm is created with the help of linear algebra.
- **Signal Analysis:** It is massively used in encoding, analyzing and manipulating the signals that can be either audio, video or images etc.
- **Linear Programming:** Optimization is an important application of linear algebra which is widely used in the field of linear programming.

- **Error-Correcting Codes:** It is used in coding theory. If encoded data is tampered with a little bit and with the help of linear algebra it should be recovered. One such important error-correcting code is called hamming code
- **Prediction:** Predictions of some objects should be found using linear models which are developed using linear algebra.
- **Facial Recognition:** An automated facial recognition technology that uses linear algebraic expression is called principal component analysis.
- **Graphics:** An important part of graphics is projecting a 3-dimensional scene on a 2-dimensional screen which is handled only by linear maps which are explained by linear algebra.

Linear Algebra Problems

Linear algebra problems include matrices, spaces, vectors, determinants, and a system of linear equation concepts. Now, let us discuss how to solve linear algebra problems.

Example 1

Find the value of x, given that

$$= \begin{bmatrix} 3 & x \\ x & 1 \end{bmatrix} = \begin{bmatrix} 3 & 2 \\ 4 & 1 \end{bmatrix}$$

Sol.:

$$\text{Given that: } \begin{bmatrix} 3 & x \\ x & 1 \end{bmatrix} = \begin{bmatrix} 3 & 2 \\ 4 & 1 \end{bmatrix}$$

Thus, the above determinant can be equated as:

$$3 - x^2 = 3 - 8$$

$$3 - x^2 = -5$$

$$-x^2 = -5 - 3$$

$$-x^2 = -8,$$

Now, multiply both sides by -1, we get

$$x^2 = 8$$

$$x = \pm 2\sqrt{2}$$

Therefore, the value of x is $\pm 2\sqrt{2}$.

Q16. Explain about vectors and implementation of vectors with the help of some examples.

Ans.:

(Imp.)

Vectors

The vectors are defined as an object containing both magnitude and direction. Vector describes the movement of an object from one point to another. Vector math can be geometrically picturized by the directed line segment.

The length of the segment of the directed line is called the magnitude of a vector and the angle at which the vector is inclined shows the direction of the vector. The starting point of a vector is called "Tail" and the ending point (having an arrow) is called "Head."

A **vector** is defined as a mathematical structure. It has many applications in the field of physics and geometry. We know that the location of the points on the coordinate plane can be represented using the ordered pair such as (x, y). The usage of the vector is very useful in the simplification process of three-dimensional geometry.

Along with the term vector, we have heard the term scalar. A scalar actually represents the "real numbers". In simpler words, a vector of "n" dimensions is an ordered collection of n elements called "**components**".

Examples of Vectors

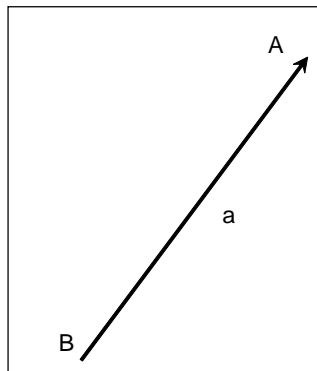
The most common examples of the vector are Velocity, Acceleration, Force, Increase/Decrease in Temperature etc. All these quantities have directions and magnitude both. Therefore, it is necessary to calculate them in their vector form.

Also, speed is a quantity that has magnitude but no direction. This is the basic difference between speed and velocity.

Vector Notation

As we know already, a vector has both magnitude and direction. In the above figure, the length of the line AB is the magnitude and head of the arrow points towards the direction.

Therefore, vectors between two points A and B is given as, \overline{AB} or vector a. The arrow over the head of the vector shows the direction of the vector.



Magnitude of a Vector

The magnitude of a vector is shown by vertical lines on both the sides of the given vector " $|a|$ ". It represents the length of the vector. Mathematically, the magnitude of a vector is calculated by the help of "Pythagoras Theorem," i.e.

$$|a| = \sqrt{x^2 + y^2}$$

Unit Vector

A unit vector has a length (or magnitude) equal to one, which is basically used to show the direction of any vector. A unit vector is equal to the ratio of a vector and its magnitude. Symbolically, it is represented by a cap or hat ($\hat{}$).

If a is vector of arbitrary length and its magnitude is $||a||$, then the unit vector is given by:

$$\hat{a} = \frac{a}{||a||}$$

It is also known as normalising a vector.

Zero Vector

A vector with zero magnitudes is called a zero vector. The coordinates of zero vector are given by (0,0,0) and it is usually represented by 0 with an arrow (\rightarrow) at the top or just 0.

The sum of any vector with zero vector is equal to the vector itself, i.e., if ' a ' is any vector, then;

$$0 + a = a$$

Note:

There is no unit vector for zero vector and it cannot be normalised.

Operations on Vectors

In maths, we have learned the different operations we perform on numbers. Let us learn here the vector operation such as Addition, Subtraction, Multiplication on vectors.

Addition of Vectors

The two vectors a and b can be added giving the sum to be $a + b$. This requires joining them head to tail.

We can translate the vector b till its tail meets the head of a . The line segment that is directed from the tail of vector a to the head of vector b is the vector " $a + b$ ".

Characteristics of Vector Math Addition

- Commutative Law- the order of addition does not matter, i.e, $a + b = b + a$
- Associative law- the sum of three vectors has nothing to do with which pair of the vectors are added at the beginning.

$$\text{i.e. } (a + b) + c = a + (b + c)$$

Subtraction of Vectors

Before going to the operation it is necessary to know about the reverse vector ($-a$).

A reverse vector ($-a$) which is opposite of ' a ' has a similar magnitude as ' a ' but pointed in the opposite direction.

First, we find the reverse vector.

Then add them as the usual addition.

Such as if we want to find vector $b - a$

$$\text{Then, } b - a = b + (-a)$$

Scalar Multiplication of Vectors

Multiplication of a vector by a scalar quantity is called "Scaling." In this type of multiplication, only the magnitude of a vector is changed not the direction.

- $S(a+b) = Sa + Sb$
- $(S+T)a = Sa + Ta$
- $a.1 = a$
- $a.0 = 0$
- $a.(-1) = -a$

Scalar Triple Product

The scalar triple product, also called a box product or mixed triple product, of three vectors, say a , b and c is given by $(a \times b) \cdot c$. Since it involves dot product and evaluates single value, therefore stated as the scalar product. It is also denoted by $(a \ b \ c)$.

The major application of the scalar triple product can be seen while determining the volume of a parallelepiped, which is equal to the absolute value of $|(a \times b) \cdot c|$, where a , b and c are the vectors denoting the sides of parallelepiped respectively. Hence,

Volume of parallelepiped

$$= ||a \times b|| \ ||c|| \ |\cos\phi| = |(a \times b) \cdot c|$$

Vector Multiplication

Basically, there are two types of vector multiplication:

- Cross product
- Dot product

Cross Product of Vectors

The cross product of two vectors results in a vector quantity. It is represented by a cross sign between two vectors.

$$a \times b$$

The mathematical value of a cross product-

$$a \times b = |a| \ |b| \ \sin \theta \ \hat{n}$$

where,

$|a|$ is the magnitude of vector a .

$|b|$ is the magnitude of vector b .

θ is the angle between two vectors a & b .

and \hat{n} is a unit vector showing the direction of the multiplication of two vectors.

Dot product of Vectors

The dot product of two vectors always results in scalar quantity, i.e. it has only magnitude and no direction. It is represented by a dot (.) in between two vectors.

$$a \cdot b = a \cdot b$$

The mathematical value of the dot product is given as

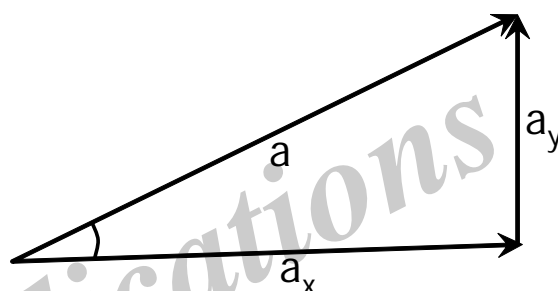
$$a \cdot b = |a| \ |b| \ \cos \theta$$

Components of Vectors (Horizontal & Vertical)

There are two components of a vector in the x-y plane.

1. Horizontal Component
2. Vertical Component

Breaking a vector into its x and y components in the vector space is the most common way for solving vectors.



A vector "a" is inclined with horizontal having an angle equal to θ .

This given vector "a" can be broken down into two components i.e. a_x and a_y .

The component a_x is called a "Horizontal component" whose value is $a \cos \theta$.

The component a_y is called a "Vertical component" whose value is $a \sin \theta$.

Applications of Vectors

Some of the important applications of vectors in real life are listed below:

- The direction in which the force is applied to move the object can be found using vectors.
- To understand how gravity uses a force of attraction on an object to work.
- The motion of a body which is confined to a plane can be obtained using vectors.
- Vectors help in defining the force applied on a body simultaneously in the three dimensions.
- Vectors are used in the field of Engineering, where the force is much stronger than the structure will sustain, else it will collapse.

- In various oscillators, vectors are used.
- Vectors also have its applications in 'Quantum Mechanics'.
- The velocity in a pipe can be determined in terms of the vector field - for example, fluid mechanics.
- We may also observe them everywhere in the general relativity.
- Vectors are used in various wave propagations such as vibration propagation, sound propagation, AC wave propagation, and so on.

PROBLEMS AND SOLUTIONS

Example 1

Given vector V, having a magnitude of 10 units and inclined at 60°. Break down the given vector into its two components.

Sol:

Given, Vector V having magnitude $|V| = 10$ units and $\theta = 60^\circ$

Horizontal component (V_x) = $V \cos \theta$

$$V_x = 10 \cos 60^\circ$$

$$V_x = 10 \times 0.5$$

$$V_x = 5 \text{ units}$$

Now, Vertical component (V_y) = $V \sin \theta$

$$V_y = 10 \sin 60^\circ$$

$$V_y = 10 \times \sqrt{3}/2$$

$$V_y = 10\sqrt{3} \text{ units}$$

Example 2

Find the magnitude of vector a (3, 4).

Sol:

Given Vector a = (3,4)

$$|a| = \sqrt{x^2 + y^2}$$

$$|a| = \sqrt{3^2 + 4^2}$$

$$|a| = \sqrt{9+16} = \sqrt{25}$$

Therefore, $|a| = 5$

Example 3

Find the scalar and vector multiplication of two vectors 'a' and 'b', given by $3\hat{i} - \hat{j} + 2\hat{k}$ and $\hat{i} + -2\hat{j} + 3\hat{k}$ respectively.

Sol:

Given vector a (3,-1,2) and vector b (1,-2,3)

Vector product (or Cross Product)

$$= \vec{a} \times \vec{b} = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ 3 & -1 & 2 \\ 1 & -2 & 3 \end{vmatrix}$$

$$\vec{a} \times \vec{b} = \begin{bmatrix} -1 & 2 \\ -2 & 3 \end{bmatrix} \hat{i} - \begin{bmatrix} 3 & 2 \\ 1 & 3 \end{bmatrix} \hat{j} + \begin{bmatrix} 3 & -1 \\ 1 & -2 \end{bmatrix} \hat{k}$$

$$\vec{a} \times \vec{b} = 1\hat{i} - 7\hat{j} + (-5)\hat{k}$$

$$\text{Now, } |\vec{a} \times \vec{b}| = \sqrt{(1)^2 + (-7)^2 + (-5)^2}$$

$$|\vec{a} \times \vec{b}| = \sqrt{75} = 5\sqrt{3}$$

Now finding magnitude of vectors a and b.

$$|\vec{a}| = \sqrt{(3)^2 + (-1)^2 + (2)^2}$$

$$|\vec{a}| = \sqrt{9+1+4} = \sqrt{14}$$

$$|\vec{b}| = \sqrt{(1)^2 + (-2)^2 + (3)^2}$$

$$|\vec{b}| = \sqrt{1+4+9} = \sqrt{14}$$

$$\sin \theta = \frac{|\vec{a} \times \vec{b}|}{|\vec{a}| \times |\vec{b}|}$$

$$\sin \theta = \frac{5\sqrt{3}}{\sqrt{14} \times \sqrt{14}}$$

$$\text{OR, } \theta = \sin^{-1} \left(\frac{5\sqrt{3}}{14} \right)$$

Thus the vector product is equal to

$$1\hat{i} - 7\hat{j} - 5\hat{k}$$

Scalar product (or Dot product)

$$= \vec{a} \cdot \vec{b} = |a| |b| \cos \theta$$

Where θ is the angle between the vectors. But we don't know the angle between the vectors thus another method of multiplication can be used.

$$a.b = (3i - 1j + 2k) \cdot (1i - 2j + 3k)$$

$$a.b = 3(i.i) + 2(j.j) + 6(k.k)$$

$$a.b = 3 + 2 + 6$$

$$a.b = 11$$

Q17. Define scalar.

Ans :

Scalars

Scalars are single numbers and are an example of a 0th-order tensor. In mathematics it is necessary to describe the set of values to which a scalar belongs. The notation $x \in R$ states that the (lowercase) scalar value x is an element of (or member of) the set of real-valued numbers, R .

There are various sets of numbers of interest within machine learning. N represents the set of positive integers (1,2,3,...). Z represents the integers, which include positive, negative and zero values. Q represents the set of *rational* numbers that may be expressed as a fraction of two integers.

Q18. What are the different forms of matrices? Explain.

Ans :

Matrix

A matrix is a two dimensional array of numbers. Generally matrices are represented by an uppercase bold letter such as A . Since a matrix is two dimensional, each element is represented by a small letter with two indices such as a_{ij} where i represents the row and j represents the column. A representation of $m \times n$ matrix is shown below,

Different Forms of Matrices

Rectangular matrices

The general form of rectangular matrices is shown below:

$$[A] = [a_{ij}] = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdot & \cdot & \cdot & \cdot & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdot & \cdot & \cdot & \cdot & a_{2n} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ a_{m1} & a_{m2} & a_{m3} & \cdot & \cdot & \cdot & \cdot & a_{mn} \end{bmatrix}$$

For example, below W is a matrix of dimensions 2×3 .

$$W = \begin{bmatrix} 3 & 7 & 8 \\ 6 & 5 & 2 \\ 9 & 4 & 1 \end{bmatrix}$$

with the "elements" of this matrix designated by a_{ij} with the first subscript i indicating the row number and the second subscript j indicating the column number.

The rectangular matrices have the number of rows $i \neq$ number of columns j .

Square matrices

This type of matrices with $i = j$, and are common in engineering analysis. Following is a typical square matrix of the size 3×3 :

$$[A] = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Row matrices

In this case, the total number of row $l = m = 1$ with the total number of columns $= n$:

$$\{A\} = \{a_{11} \ a_{12} \ a_{13} \ \cdot \ \cdot \ \cdot \ a_{1n}\}$$

Column matrices

These matrices have only one column, i.e. $n = j = 1$ but with m rows. Column matrices are used to express the components of vector quantities, such as the expression of a force vector

$$\{A\} = \begin{Bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ a_{m1} \end{Bmatrix}$$

Upper triangular matrices

We realize that all SQUARE matrices have a "diagonal" line across the elements drawn from those

at the first row and column. An upper triangular matrix has all the elements in this matrix to be zero below its diagonal line, such as

Illustrated in the form in the right for an upper triangular matrix of 3×3 with elements below the diagonal line: $a_{21} = a_{31} = a_{32} = 0$:

Diagonal of a square matrix

$$[A] = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Lower triangular matrices

This is an opposite case to the upper triangular matrix, in which all elements above the diagonal lines are zero as shown below for a 3×3 square matrix:

$$[A] = \begin{bmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Diagonal matrices

In these matrices, the only non-zero elements are those on the diagonals. Example of a diagonal matrix of the size of 4×4 is shown below:

$$[A] = \begin{bmatrix} a_{11} & 0 & 0 & 0 \\ 0 & a_{22} & 0 & 0 \\ 0 & 0 & a_{33} & 0 \\ 0 & 0 & 0 & a_{44} \end{bmatrix}$$

This type of matrices is similar to that of diagonal matrices, except that the non-zero elements on the diagonal lines have a value of unity, i.e. "1.0". A 4×4 unity matrix is shown in the right::

$$[I] = \begin{bmatrix} 1.0 & 0 & \dots & 0 \\ \vdots & & & \vdots \\ 0 & \dots & & 1.0 \end{bmatrix}$$

Unity matrices have the following useful properties.

$$\alpha[I] = \alpha \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & \alpha \end{bmatrix} = \alpha$$

diagonal matrix and $[A][I] = [I][A]$

Transposition of Matrices

Transposition of a matrix $[A]$ often is required in engineering analysis. It is designated by $[A]^T$.

Transposition of matrix $[A]$ is carried out by interchanging the subscripts that define the locations of the elements in matrix $[A]$.

Mathematical operations of matrix transposition will be followed by letting $[a_{ij}]^T = [a_{ji}]$.

Following are three such examples:

Case 1: Transposing a column matrix into a row matrix:

$$\{A\}^T = \begin{Bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ \vdots \\ a_{m1} \end{Bmatrix} = \{a_{11} \ a_{21} \ a_{31} \ \dots \ a_{m1}\}$$

Case 2: Transposing a rectangular matrix into another rectangular matrix:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}^T = \begin{bmatrix} a_{11} & a_{21} \\ a_{12} & a_{22} \\ a_{13} & a_{23} \end{bmatrix}$$

Case 3: Transposing a square matrix into another square matrix:

Diagonal of a square matrix

$$[A] = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

(a) Original Matrix

$$[A] = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

(b) Transposed Matrix

Q19. Write about various operations performed on matrices.

Ans :

Matrix Algebra

Addition and subtraction of matrices

We are made aware of the fact that matrices are expressions of arrays of numbers or variables – but not single numbers. As such, addition/subtraction and multiplications of matrices need to follow certain rules.

Addition or subtraction of two matrices requires that both matrices having the same size, i.e., with equal number of rows and columns.

$$[A] \pm [B] = [C] \text{ with } C_{ij} = a_{ij} \pm b_{ij}$$

in which a_{ij} , b_{ij} and c_{ij} are the elements of the matrices $[A]$, $[B]$ and $[C]$ respectively.

Multiplication of matrices by a scalar quantity

$$\alpha [C] = [\alpha C_{ij}]$$

Multiplication of two matrices

Multiplication of two matrices requires the satisfaction of the following condition:

The total number of column in the first matrix = *the total number of rows in the second matrix*

$$[C]_{(m \times p)} = [A]_{(m \times n)} \times [B]_{(n \times p)}$$

in which the notations shown in the parentheses below the matrices denotes the number of rows and columns in each of these matrices.

The following recurrence relationship may be used to determine the elements in the product matrix $[C]$ with $c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \dots + a_{in}b_{nj}$ where $i=1,2,\dots,m$ and $j=1,2,3,\dots,n$.

Following are four (4) examples on multiplications of matrices

Example 1

Multiply two matrices $[A]$ and $[B]$ defined as:

$$[A] = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \text{ and } [B] = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix}$$

Sol :

$$\begin{aligned} [C] = [A] [B] &= \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} \\ &= \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} + a_{13}b_{31} & a_{11}b_{12} + a_{12}b_{22} + a_{13}b_{32} & a_{11}b_{13} + a_{12}b_{23} + a_{13}b_{33} \\ a_{21}b_{11} + a_{22}b_{21} + a_{23}b_{31} & \bullet & \bullet \\ \bullet & \bullet & \bullet \end{bmatrix} \end{aligned}$$

Example 2

Multiply the following rectangular matrix and a column matrix

Sol:

We checked the number of column of the first matrix equals the number of rows of the second Matrix. We may thus have the flowing multiplication:

$$\{y\} = [C] \{x\} = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} C_{11}x_1 + C_{12}x_2 + C_{13}x_3 \\ C_{21}x_1 + C_{22}x_2 + C_{23}x_3 \end{Bmatrix} = \begin{Bmatrix} y_1 \\ y_2 \end{Bmatrix}$$

Example 3

This example will show the difference in the results of multiplication of two matrices in the different OEDER of the matrices.

Case A: Multiply a Row matrix by a Column matrix, resulting in a scalar quantity.

$$\{a_{11} \quad a_{12} \quad a_{13}\} \begin{Bmatrix} b_{11} \\ b_{21} \\ b_{31} \end{Bmatrix} = a_{11}b_{11} + a_{12}b_{21} + a_{13}b_{31}$$

Case B: Multiply a Column matrix by a Row matrix, resulting in a Square matrix!

$$\begin{Bmatrix} a_{11} \\ a_{21} \\ a_{31} \end{Bmatrix} \{b_{11} \quad b_{12} \quad b_{13}\} = \begin{bmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{11}b_{13} \\ a_{21}b_{11} & a_{21}b_{12} & a_{21}b_{13} \\ a_{31}b_{11} & a_{31}b_{12} & a_{31}b_{13} \end{bmatrix} \quad (\text{a square matrix})$$

Additional rules on multiplication of matrices

- **Distributive law:** $[A]([B] + [C]) = [A][B] + [A][C]$
- **Associative law:** $[A]([B][C]) = [A][B][C]$
- **Caution:** Different order of multiplications of two matrices will result indifferent forms, i.e. the order of matrices in multiplication is very important. Always Remember the following relations:

$$[A][B] \neq [B][A]$$

- Product of two transposed matrices: $([A][B])^T = [B]^T[A]^T$

Matrix Representation of Simultaneous Equations

Matrix operations are powerful tools in modern-day engineering analysis. They are widely used in solving large numbers of simultaneous equations using digital computers. Following are the expressions on how matrices may be used to develop algorithms for the solution of large number of n-simultaneous equations:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdot & \cdot & \cdot & \cdot & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdot & \cdot & \cdot & \cdot & a_{2n} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & a_{n3} & \cdot & \cdot & \cdot & \cdot & a_{nm} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ x_n \end{Bmatrix} = \begin{Bmatrix} r_1 \\ r_2 \\ \cdot \\ \cdot \\ r_n \end{Bmatrix}$$

$[A] \qquad \qquad \qquad \{x\} = \{r\}$

From which, we may conveniently express these simultaneous linear equations in the following simplified form: where matrix $[A]$ is usually referred to as the "coefficient matrix," $\{x\}$ is the "unknown matrix," and $\{r\}$ is the "resultant matrix."

Example 4

The matrix equation:

$$\begin{bmatrix} 8 & 4 & 1 \\ 2 & 6 & -1 \\ 1 & -2 & 1 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 12 \\ 3 \\ 2 \end{Bmatrix}$$

represents the 3 simultaneous Equations:

$$8x_1 + 4x_2 + x_3 = 12$$

$$2x_1 + 6x_2 - x_3 = 3$$

$$x_1 - 2x_2 + x_3 = 2$$

where x_1 , x_2 and x_3 are the unknowns to be solved by these 3 simultaneous equations.

Ans :

Matrix Inversion $[A]^{-1}$

Since matrices are used to represent ARRAYS of numbers or variables in engineering analysis (but not single numbers or variables), there is no such thing as the division of two matrices. The closest to "divisions" in matrix algebra is matrix inversion. We define the inversion of matrix $[A]$, i.e. $[A]^{-1}$ to be: $[A][A]^{-1} = [A]^{-1}[A] = [I]$ where $[I]$ is a unity matrix defined by Equation.

One must note a fact that inversion of a matrix $[A]$ is possible only if the equivalent determinant of $[A]$, i.e.

The matrix $[A]$ is called "singular matrix" if $|A| \neq 0$.

Following are the 4 steps to invert the matrix $[A]$:

Step 1

Evaluate the equivalent determinant of the matrix $[A]$, and make sure that $|A| \neq 0$.

Step 2

If the elements of matrix $[A]$ are a_{ij} , we may determine the elements of the co-factor matrix $[C]$ to be:

in which $|A'|$ is the equivalent determinant of a matrix $[A']$ that has all elements of $[A]$ excluding those in the i^{th} row and j^{th} column.

Step 3

Transpose the co-factor matrix from $[C]$ to $[C]^T$.

Step 4

The inverse matrix $[A]^{-1}$ for matrix $[A]$ may be established by the following expression.

Step 5

Transpose the co-factor matrix from $[C]$ to $[C]^T$.

Step 6

The inverse matrix $[A]^{-1}$ for matrix $[A]$ may be established by the following expression:

$$[A]^{-1} = \frac{1}{|A|} [C]^T$$

Example 5

We will invert the following 3×3 matrix $[A]$ following the 4 steps specified in the proceeding slide:

Ans :

$$[A] = \begin{bmatrix} 1 & 2 & 3 \\ 0 & -1 & 4 \\ -2 & 5 & -3 \end{bmatrix}$$

Step 1

Evaluate the equivalent determinant of $[A]$:

$$\begin{aligned} |A| &= \begin{vmatrix} 1 & 2 & 3 \\ 0 & -1 & 4 \\ -2 & 5 & -3 \end{vmatrix} \\ &= 1 \begin{vmatrix} -1 & 4 \\ 5 & -3 \end{vmatrix} - 2 \begin{vmatrix} 0 & 4 \\ -2 & -3 \end{vmatrix} + 3 \begin{vmatrix} 0 & -1 \\ -2 & -3 \end{vmatrix} \\ &= -39 (\neq 0) \end{aligned}$$

Step 2

Determine the elements of the co-factor matrix, $[C]$:

$$C_{11} = (-1)^{1+1} [(-1)(-3) - (4)(5)] = -17$$

$$C_{12} = (-1)^{1+2} [(0)(-3) - (4)(-2)] = -18$$

$$C_{13} = (-1)^{1+3} [(0)(5) - (1)(-2)] = -2$$

$$C_{21} = (-1)^{2+1} [(2)(-3) - (3)(5)] = 21$$

$$C_{22} = (-1)^{2+2} [(1)(-3) - (3)(-2)] = 3$$

$$C_{23} = (-1)^{2+3} [(1)(5) - (2)(-2)] = -9$$

$$C_{31} = (-1)^{3+1} [(2)(4) - (3)(-1)] = 11$$

$$C_{32} = (-1)^{3+2} [(1)(4) - (3)(0)] = -4$$

$$C_{33} = (-1)^{3+3} [(1)(-1) - (2)(0)] = -1$$

We thus have the co-factor matrix, [C] in the form:

$$[C] = \begin{bmatrix} -17 & -8 & -2 \\ 21 & 3 & -9 \\ 11 & -4 & -1 \end{bmatrix}$$

Step 3

Transpose the [C] matrix is:

$$[C]^T = \begin{bmatrix} -17 & 21 & 11 \\ -8 & 3 & -4 \\ -2 & -9 & -1 \end{bmatrix}$$

which leads to the inverted matrix [A] to be:

Step 4

Determine the inverse matrix, $[A]^{-1}$ following Equation

$$[A]^{-1} = \frac{[C]^T}{|A|} = \frac{1}{-39} \begin{bmatrix} -17 & 21 & 11 \\ -8 & 3 & -4 \\ -2 & -9 & -1 \end{bmatrix}$$

$$= \frac{1}{39} \begin{bmatrix} 17 & -21 & -11 \\ 8 & -3 & 4 \\ 2 & 9 & 1 \end{bmatrix}$$

One may verify the correct inversion of matrix [A] by:

$[A][A]^{-1} = [I]$ where [I] is a unit matrix defined in Equation.

Q20. Write about tensors and the use of tensors in machine language.

Ans :

Tensor

- An tensor is an array of data (numbers, functions, etc.) which is expanded in any number (0 and greater) of dimensions. The number of dimensions is called rank of tensor
- A tensor is a vector or matrix of n-dimensions that represents all types of data. All values in a tensor hold identical data type with a known (or partially known) shape.
- The shape of the data is the dimensionality of the matrix or array.
- The graph is a set of computation that takes place successively. Each operation is called an op node and are connected to each other.

Rank 0 tensor

A tensor that has no dimensions (0).

A

[5]

A is a 0 dimensional tensor

Rank 1 tensor

A tensor that is expanded in only one dimension.

$$\begin{bmatrix} 2 \\ 31 \\ 1 \end{bmatrix} \downarrow \text{Dimension \#1} \begin{bmatrix} 21 & 8 & 11 \end{bmatrix} \xrightarrow{\text{Dimension \#1}}$$

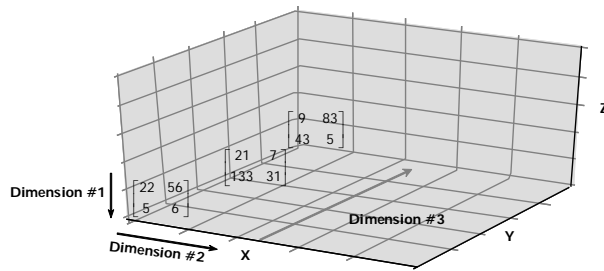
Examples of 1 dimensional tensors

Rank 2 tensor

$$\begin{matrix} & \xrightarrow{\text{Dimension \#2}} \\ \text{Dimension \#1} \downarrow & \begin{bmatrix} 2 & 333 & 5 \\ 11 & 54 & 90 \\ 21 & 5 & 51 \end{bmatrix} \end{matrix}$$

A two dimensional tensor

Rank 3 tensor



Types of Tensor

In TensorFlow, all the computations pass through one or more tensors. A `tf.tensor` is an object with three properties:

- A unique label (name)
- A dimension (shape)
- A data type (dtype)

Each operation you will do with TensorFlow involves the manipulation of a tensor. There are four main tensor type you can create:

- `tf.Variable`
- `tf.constant`
- `tf.placeholder`
- `tf.SparseTensor`

Tensors in Machine Learning

While the above is all true, there is nuance in what tensors technically are and what we refer to as tensors as relates to machine learning practice. If we temporarily consider them simply to be data structures, below is an overview of where tensors fit in with scalars, vectors, and matrices, and some simple code demonstrating how Numpy can be used to create each of these data types.

1.2 CONVEX OPTIMIZATION BACKGROUND

Q21. What is convex optimisation ? Explain how can we solve convex optimization problem.

Ans : (Imp.)

1. Many situations arise in machine learning where we would like to optimize the value of some function.

- That is, given a function $f: \mathbb{R}^n \rightarrow \mathbb{R}$, we want to find $x \in \mathbb{R}^n$ that minimizes (or maximizes) $f(x)$.
- In the general case, finding the global optimum of a function can be a very difficult task.
- However, for a special class of optimization problems, known as convex optimization problems, we can efficiently find the global solution in many cases.
- "Efficiently" has both practical and theoretical connotations:
- It means that we can solve many real-world problems in a reasonable amount of time.

2. And it means that theoretically we can solve problems in time that depends only polynomially on the problem size.

Convex Sets Definition

A set C is convex if, for any $x, y \in C$ and with $\lambda \in [0, 1]$, the point $\lambda x + (1 - \lambda)y$ is also in C . Intuitively, this means that if we take any two elements in C , and draw a line segment between these two elements, then every point on that line segment also belongs to C .

Figure 1 shows an example of one convex and one non-convex set. The point is called a convex combination of the points x and y .

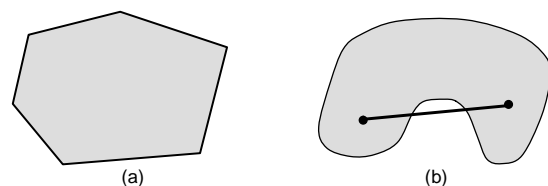


Fig.: Examples of a convex set (a) and a non-convex set

Optimization problem Standard form

Minimize $f_0(x)$

Subject to $f_i(x) \leq 0, \quad i = 1, \dots, m$

$h_i(x) = 0, \quad i = 1, \dots, p$

- $x \in \mathbb{R}^n$ is the optimization variable
- $f_0: \mathbb{R}^n \rightarrow \mathbb{R}$ is the objective or cost function
- $f_i: \mathbb{R}^n \rightarrow \mathbb{R}, \quad i = 1, \dots, m$, are the inequality constraint functions.

- $h_i : \mathbb{R}^n \rightarrow \mathbb{R}$ are the equality constraint functions

Domain and implicit constraints

$$x \in D = \bigcap_{i=0}^m \text{dom } f_i \cap \bigcap_{i=1}^p \text{dom } h_i,$$

Feasible

Variable x is feasible: and x satisfies All constraints.

Problem is feasible

At least exists one feasible variable

Optimal value

$$p^* = \inf\{f_0(x) \mid f_i(x) \leq 0, i = 1, \dots, m, h_i(x) = 0, i = 1, \dots, p\}$$

$p^* = \infty$ if problem is infeasible (no x satisfies the constraints).

$p^* = -\infty$ if problem is unbounded below.

Optimal solution

A feasible x is optimal if $f_0(x) = p^*$;

Local optimal point

x is locally optimal if there is an $R > 0$ such that x is optimal for

$$\begin{aligned} &\text{Minimize (over } z) f_0(z) \\ &\text{subject to } f_i(z) \leq 0, i = 1, \dots, m, h_i(z) = 0, \\ &\quad i = 1, \dots, p \mid \|z - x\|_2 \leq R. \end{aligned}$$

- $f_0(x) = x^3 - 3x$, $p^* = -\infty$, local optimum at $x = 1$.

Results for one optimal problem

- Not feasible
 $f_1(x) = \varepsilon x$
- Unbounded below
 $f_0(x) = -\log x$, $\text{dom } f_0 = \mathbb{R}_{++}$: $p^* = -\infty$
- Has optimal value, but no optimal point
 $f_0(x) = 1/x$, $\text{dom } f_0 = \mathbb{R}_{++}$: $p^* = 0$, no optimal point
- Has optimal value, and also optimal point
 $f_0(x) = x \log x$, $\text{dom } f_0 = \mathbb{R}_{++}$: $p^* = -1/e$, $x = 1/e$ is optimal.

Convex optimization

Convex optimization has applications in a wide range of disciplines, such as automatic control systems, estimation and signal processing, communications and networks, electronic circuit design, data analysis and modeling, finance, statistics (optimal experimental design), and structural optimization, where the approximation concept has proven to be efficient. With recent advancements in computing and optimization algorithms, convex programming is nearly as straightforward as linear programming.

Stanford form

$$\text{Minimize } f_0(x)$$

$$\text{subject to } f_i(x) \leq 0, i = 1, \dots, m$$

$$a_i^T x = b_i, i = 1, \dots, p$$

f_0, f_1, \dots, f_m are convex; equality constraints are affine.

Feasible set of a convex optimization problem is convex set

- Domain of convex function is convex set
- Sublevel set of convex function is convex set
- Intersection of convex sets are convex set.

Abstract convex optimization problem

- **Convex optimization:** Minimize a convex function in convex set

$$\text{Minimize } f_0(x) = x_1^2 + x_2^2$$

$$\text{subject to } f_1(x) = x_1/(1 + x_2^2) \leq 0$$

$$h_1(x) = (x_1 + x_2)^2 = 0$$

- f_0 is convex; feasible set $\{(x_1, x_2) \mid x_1 = -x_2 \leq 0\}$ is convex.
- not a convex problem (according to our definition): f_1 is not convex, h_1 is not affine.
- Equivalent (but not identical) to the convex problem.

$$\text{Minimize } x_1^2 + x_2^2$$

$$\text{Subject to } x_1 \leq 0$$

$$x_1 + x_2 = 0$$

Convert to a equivalent convex problem may help solve

Local optima is global optima

Suppose x is locally optimal and y is optimal with $f_0(y) < f_0(x)$.

Consider: $z = \theta y + (1 - \theta)x$

We have: $f_0(z) \leq \theta f_0(x) + (1 - \theta) f_0(y) < f_0(x)$

We can easily choose a small θ , which contradicts our assumption that x is local optimal

Ex: $\theta = R/(2 \|y-x\|)$

Equivalent convex problem

Examples

- Two problems are equivalent if the solution of one is readily obtained from the other one
- Some transformations preserve convexity, which is nice.

Example: eliminating equality constraints

Minimize $f_0(x)$

Subject to $f_i(x) \leq 0, i = 1, \dots, m$

$Ax = b$

is equivalent to

minimize (over z) $f_0(Fz + x_0)$

subject of $f_i(Fz + x_0) \leq 0, i = 1, \dots, m$

where F and x_0 are such that

Introducing equality constraints

Minimize $f_0(Fz + x_0)$

Subject to $f_i(A_i x + b_i) \leq 0, i = 1, \dots, m$

$Ax = b$

is equivalent to

minimize (over x, y) $f_0(y)$

subject of $f_i(y) \leq 0, i = 1, \dots, m$

$y_i = A_i x + b_i, i = 0, 1, \dots, m$

Introducing slack variables for linear inequalities

Minimize $f_0(x)$

Subject to $a_i^T x \leq b_i$

is equivalent to

minimize (over x, s) $f_0(x)$

subject of $a_i^T x + s_i = b_i, i = 1, \dots, m$

$s_i \geq 0, i = 1, \dots, m$

- **Epigraph form:** standard form convex problem is equivalent to,

minimize (over x, t) t

Subject to $f_0(x) - t \leq 0$

$f_i(x) \leq 0, i = 1, \dots, m$

- Minimizing over some variables

minimize $f_0(x_1, x_2)$

Subject to $f_i(x_1) \leq 0, i = 1, \dots, m$

is equivalent to

minimize $f_0(x_1)$

subject to $f_i(x_1) \leq 0, i = 1, \dots, m$

Where $f_0(x_1) = \inf_{x_2} f_0(x_1, x_2)$.

1.3 STATISTICAL DECISION THEORY

Q22. What is statistical decision theory? Explain its framework.

Ans :

This is mathematical theory in the field of Machine Learning that allows us to make optimal decisions in situations involving uncertainty.

So from the figure we can see that the goal of the physician would be to get the highest score possible which is 100% and that is the objective of Decision Theory, to make the most optimal decision.

Very Good Score: 100%	Very Bad Score: 0%
Not Good 40%	Good 85%

- Minimise wrong decisions
- Reduce expected loss

Application of Decision Theory – Cancer Diagnosis

Scenario 1

There is presence of cancer and the physician decides to perform a surgery. That is 100% because it's the best decision to take.

Scenario 2

There is presence of cancer but the physician decides not to perform a surgery. That is a score of 0 as it is the worst case scenario and of course the consequences would be very serious.

Scenario 3

Cancer is absent but the physician decides to perform a surgery anyway. This is a low score but does not result in any serious consequence

Scenario 4

Cancer is absent and the physician decides not to perform a surgery. This is a good decision as well.

	Surgery Performed	Surgery Not Performed
Cancer Present (C1)	Very Good Score: 100%	Very Bad Score: 0%
Cancer Absent (C2)	Not Good 40%	Good 85%

Statistical decision theory is concerned with the problem of making decisions.

It combines the sampling information (data) with a knowledge of the consequences of our decisions.

Three major types of inference:

- point estimator (educated guess")
- confidence interval,
- hypotheses testing,

Framework for a Decision Problem

Decision maker has available K possible courses of action: a_1, a_2, \dots, a_K . Actions are sometime called alternatives. There are H possible uncertain states of nature: s_1, s_2, \dots, s_H . States of nature are the possible outcomes over which the

decision maker has no control. Sometimes states of nature are called events. For each possible action-state of nature combination, there is an associated outcome representing either profit or loss, called the monetary payoff, M_{ij} , that corresponds to action a_i and state of nature s_j . The table of all such outcomes for a decision problem is called a payoff table.

Payoff Table for a Decision Problem with K Possible Actions and H Possible States of Nature

ACTIONS		STATES OF NATURE			
a_i	s_j	s_1	s_2	\dots	s_H
a_1		M_{11}	M_{12}	\dots	M_{1H}
a_2		M_{21}	M_{22}	\dots	M_{2H}
\vdots		\vdots	\vdots	\vdots	\vdots
\vdots		\vdots	\vdots	\vdots	\vdots
\vdots		\vdots	\vdots	\vdots	\vdots
a_K		M_{K1}	M_{K2}	\dots	M_{KH}

Decision Rule Based on Maximin Criterion

Suppose that a decision maker has to choose from K admissible actions a_1, a_2, \dots, a_K , given H possible states of nature s_1, s_2, \dots, s_H . Let M_{ij} denote the payoff corresponding to the i th action and j th state of nature. For each action, seek the smallest possible payoff. For action a_1 , for example, this is the smallest of $M_{11}, M_{12}, \dots, M_{1H}$. Let us denote the minimum M_1^* where

$$M_1^* = \min(M_{11}, M_{12}, \dots, M_{1H})$$

More generally, the smallest possible payoff for action a_i is given by

$$M_i^* = \min(M_{i1}, M_{i2}, \dots, M_{iH})$$

The maximin criterion then selects the action a_i for which the corresponding M_i^* is largest (that is, the action for which the minimum payoff is highest).

Regret or Opportunity Loss Table

Suppose that a payoff table is arranged as a rectangular array, with rows corresponding to actions and columns to states of nature. If each payoff in the table is subtracted from the largest payoff in its column, the resulting array is called a regret table, or opportunity loss table.

Decision Rule Based on the Minimax Criterion

Given the regret table, the action dictated by the minimax regret criterion is found as follows: For each row (action), find the maximum regret. Choose the action corresponding to the minimum of these maximum regrets. The minimax criterion selects the action for which the maximum regret is smallest; that is, the minimax regret criterion produces the smallest possible opportunity loss that can be guaranteed.

Payoffs with State-of-Nature Probabilities

ACTIONS		STATES OF NATURE			
a_i	s_i	s_1 (π_1)	s_2 (π_2)	...	s_H (π_H)
a_1		M_{11}	M_{12}	...	M_{1H}
a_2		M_{21}	M_{22}	...	M_{2H}
.
.
.
a_K		M_{K1}	M_{K2}	...	M_{KH}

Expected Monetary Value (EMV) Criterion

Suppose that a decision maker has K possible actions, a_1, a_2, \dots, a_K and is faced with H states of nature. Let M_{ij} denote the payoff corresponding to the i th action and j th state and p_j the probability of occurrence of the j th state of nature with

The expected monetary value of action a_i , $EMV(a_i)$, is

$$EMV(a_i) = \pi_1 M_{i1} + \pi_2 M_{i2} + \dots + \pi_H M_{iH}$$

$$= \sum_{j=1}^H \pi_j M_{ij}$$

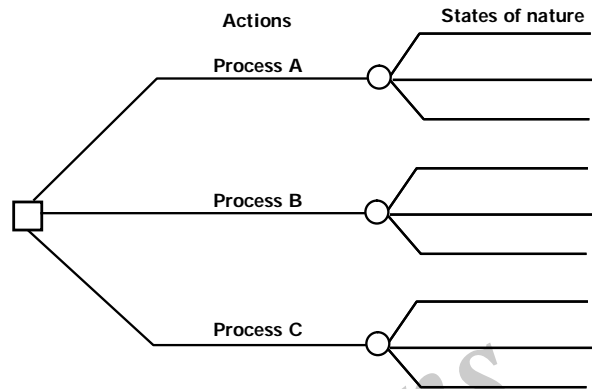
The Expected Monetary Value Criterion adopts the action with the largest expected monetary value; that is, given a choice among alternation actions, the EMV criterion dictates the choice of the action for which the EMV is highest.

Decision Trees

The tree diagram is a graphical device that forces the decision-maker to examine all possible outcomes, including unfavorable ones

All decision trees contain:

- Decision (or action) nodes
- Event (or state-of-nature) nodes
- Terminal nodes



Bayes' Theorem

Let s_1, s_2, \dots, s_H be H mutually exclusive and collectively exhaustive events, corresponding to the H states of nature of a decision problem. Let A be some other event. Denote the conditional probability that s_i will occur, given that A occurs, by $P(s_i | A)$, and the probability of A , given s_i , by $P(A | s_i)$. Bayes' Theorem states that the conditional probability of s_i , given A , can be expressed as,

$$P(s_i | A) = \frac{P(A | s_i)P(s_i)}{P(A)}$$

$$P(s_i | A) = \frac{P(A | s_i)P(s_i)}{P(A | s_1)P(s_1) + P(A | s_2)P(s_2) + \dots + P(A | s_H)P(s_H)}$$

In the terminology of this section, $P(s_i)$ is the prior probability of s_i and is modified to the posterior probability, $P(s_i | A)$, given the sample information that event A has occurred.

Expected Value of Perfect Information, EVPI

Suppose that a decision maker has to choose from among K possible actions, in the face of H states of nature, s_1, s_2, \dots, s_H . Perfect information corresponds to knowledge of which state of nature will arise. The expected value of perfect information is obtained as follows:

- i. Determine which action will be chosen if only the prior probabilities $P(s_1), P(s_2), \dots, P(s_H)$ are used

- ii. For each possible state of nature, s_i , find the difference, W_i , between the payoff for the best choice of action, if it were known that state would arise, and the payoff for the action chosen if only prior probabilities are used. This is the value of perfect information, when it is known that s_i will occur.

- iii. The expected value of perfect information, EVPI, is then

$$EVPI = P(s_1)W_1 + P(s_2)W_2 + \dots + P(s_H)W_H$$

Expected Value of Sample Information, EVSI

Suppose that a decision maker has to choose from among K possible actions, in the face of H states of nature, s_1, s_2, \dots, s_H . The decision-maker may obtain sample information. Let there be M possible sample results, A_1, A_2, \dots, A_M . The expected value of sample information is obtained as follows:

- i) Determine which action will be chosen if only the prior probabilities were used.

- ii) Determine the probabilities of obtaining each sample result:

$$P(A_i) = P(A_i | s_1)P(s_1) + P(A_i | s_2) + \dots + P(A_i | s_H)P(s_H)$$

- iii) For each possible sample result, A_i , find the difference, V_i , between the expected monetary value for the optimal action and that for the action chosen if only the prior probabilities are used. This is the value of the sample information, given that A_i was observed.

- iv) The expected value of sample information, EVSI, is then:

$$EVPI = P(s_1)W_1 + P(s_2)W_2 + \dots + P(s_H)W_H$$

Obtaining a Utility Function

Suppose that a decision maker may receive several alternative payoffs. The transformation from payoffs to utilities is obtained as follows:

- The units in which utility is measured are arbitrary. Accordingly, a scale can be fixed in any convenient fashion.
- Let L be the lowest and H the highest of all the payoffs. Assign utility 0 to payoff L and utility 100 to payoff H . Let I be any payoff

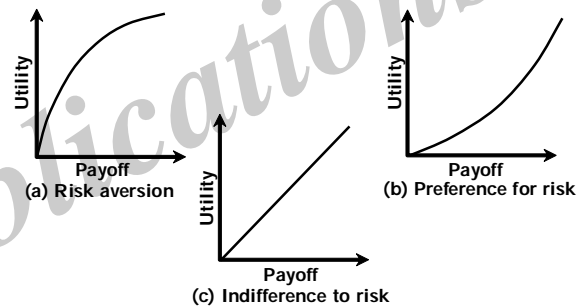
between L and H . Determine the probability p such that the decision-maker is indifferent between the following alternatives:

- Receive payoff I with certainty
- Receive payoff H with probability π and payoff L with probability $(1 - \pi)$

- iii. The utility to the decision-maker of payoff I is then 100π . The curve relating utility to payoff is called a utility function.

Utility Functions

- Risk Aversion;
- Preference for Risk;
- Indifference to Risk



The Expected Utility Criterion

Suppose that a decision maker has K possible actions, a_1, a_2, \dots, a_K and is faced with H states of nature. Let U_{ij} denote the utility corresponding to the i th action and j th state and p_j the probability of occurrence of the j th state of nature. Then the expected utility, $EU(a_i)$, of the action a_i is,

$$EU(a_i) = \pi_1 U_{i1} + \pi_2 U_{i2} + \dots + \pi_h U_{ih} = \sum_{j=1}^h \pi_j U_{ij}$$

Given a choice between alternative actions, the expected utility criterion dictates the choice of the action for which expected utility is highest. Under generally reasonable assumptions, it can be shown that the rational decision-maker should adopt this criterion

If the decision-maker is indifferent to risk, the expected utility criterion and expected monetary value criterion are equivalent.

1.4 BAYESIAN LEARNING (ML, MAP, BAYES ESTIMATES, CONJUGATE PRIORS)

Q23. Explain Bayesian Learning in terms of ML, MAP, Bayes Estimates and Conjugate Priors.

Ans :

(Imp.)

- Bayesian machine learning is a particular set of approaches to probabilistic machine learning.
- Bayesian learning treats model parameters as random variables – in Bayesian learning, parameter estimation amounts to computing posterior distributions for these random variables based on the observed data.
- Bayesian probability basically talks about the interpretation of "partial beliefs".
- Bayesian Estimation calculates the validity of the proposition.

Validity of the Proposition depends on two things:

- i) Prior Estimate.
- ii) New Relevant evidence.

Focusing First on the Prior Estimation of the Parameters

We can interpret the Bayes' Rule

$$\text{prob}(\Theta | X) = \frac{\text{prob}(X | \Theta) \cdot \text{prob}(\Theta)}{\text{prob}(X)}$$

$$\text{posterior} = \frac{\text{like lihood} \cdot \text{prior}}{\text{evidence}}$$

As

Making explicit the formula for likelihood as used above, we can write

$$\text{likelihood} = \text{prob}(X | \Theta)$$

Maximum Likelihood (ML) Estimation of Θ

Maximum likelihood estimation is a method that determines values for the parameters of a model. The parameter values are found such that they maximize the likelihood that the process described by the model produced the data that were actually observed.

We seek that value for Θ which maximizes the likelihood shown on the previous slide. That is, we seek that value for Θ which gives largest value to $\text{prob}(X | \Theta)$

We denote such a value of Θ by Θ_{ML}

We know that the joint probability of a collection of independent random variables is a product of the probabilities associated with the individual random variables in the collection.

Recognizing that the evidence X consists of the independent observations $\{x_1, x_2, \dots\}$, we seek that value Θ which maximizes

$$\prod_{x_i \in X} \text{prob}(x_i | \Theta)$$

Because of the product in the expression at the bottom of the previous slide, it is simpler to use its logarithm instead.

Using the symbol L to denote the logarithm:

$$L = \sum_{x_i \in X} \log \text{prob}(x_i | \Theta)$$

we can now write for the ML solution:

$$\hat{\Theta}_{ML} = \arg \max_{\Theta} L$$

That is, we seek those values for the parameters in Θ which maximize L . The ML solution is usually obtained by setting.

$$\frac{\partial L}{\partial \theta_i} = 0 \quad \forall \theta_i \in \Theta$$

MAP Hypothesis

In Bayesian statistics, a maximum a posterior probability (MAP) estimate is an estimate of an unknown quantity, that equals the mode of the posterior distribution. The MAP can be used to obtain a point estimate of an unobserved quantity on the basis of empirical data.

Maximum a Posteriori (MAP) Estimation of Θ For constructing the maximum a posteriori estimate for the parameter set Θ , we first go back to the Bayes' Rule.

$$\text{prob}(\Theta | X) = \frac{\text{prob}(X | \Theta) \cdot \text{prob}(\Theta)}{\text{prob}(X)}$$

We now seek that value for Θ which maximizes the posterior $\text{prob}(\Theta | X)$.

We denote such a value of Θ by $\hat{\Theta}_{\text{MAP}}$

$$\begin{aligned}\hat{\Theta}_{\text{MAP}} &= \arg \max_{\Theta} \text{prob}(\Theta | X) \\ &= \arg \max_{\Theta} \frac{\text{prob}(X | \Theta) \cdot \text{prob}(\Theta)}{\text{prob}(X)} \\ &= \arg \max_{\Theta} \text{prob}(X | \Theta) \cdot \text{prob}(\Theta) \\ &= \arg \max_{\Theta} \prod_{x_i \in X} \text{prob}(x_i | \Theta) \cdot \text{prob}(\Theta)\end{aligned}$$

As to why we dropped the denominator in the third re-write on the right, that's because it has no direct functional dependence on the parameters with respect to which we want the right-hand side to be maximized.

As with the ML estimate, we can make this problem easier if we first take the logarithm of the posteriors. We can then write

$$\hat{\Theta}_{\text{MAP}} = \arg \max_{\Theta} \left(\sum_{x_i \in X} \log \text{prob}(x_i | \Theta) + \log \text{prob}(\Theta) \right)$$

Bayesian Estimation

Given the evidence X , ML considers the parameter vector Θ to be a constant and seeks out that value for the constant that provides maximum support for the evidence. ML does not allow us to inject our prior beliefs about the likely values for Θ in the estimation calculations.

MAP allows for the fact that the parameter vector Θ can take values from a distribution that expresses our prior beliefs regarding the parameters. MAP returns that value for Θ where the probability $\text{prob}(\Theta | X)$ is a maximum.

Both ML and MAP return only single and specific values for the parameter Θ .

Bayesian estimation, by contrast, calculates fully the posterior distribution $\text{prob}(\Theta | X)$.

Of all the Θ values made possible by the estimated posterior distribution, it is our job to select a value that we consider best in some sense. For example, we may choose the expected value of Θ assuming its variance is small enough.

The variance that we can calculate for the parameter from its posterior distribution allows us to express our confidence in any specific value we may use as an estimate. If the variance is too large, we may declare that there does not exist a good estimate for Θ .

What makes Bayesian Estimation Complicated?

Bayesian estimation is made complex by the fact that now the denominator in the Bayes' Rule cannot be ignored. The denominator, known as the probability of evidence, is related to the other probabilities that make their appearance in the Bayes' Rule by

$$\text{prob}(\Theta | \chi) = \frac{\text{prob}(\chi | \Theta) \cdot \text{prob}(\Theta)}{\text{prob}(\chi)}$$

$$\text{prob}(\chi) = \int_{\Theta} \text{prob}(\chi | \Theta) \cdot \text{prob}(\Theta) d\Theta$$

Conjugate Priors

As you saw, Bayesian estimation requires us to compute the full posterior distribution for the parameters of interest, as opposed to, say, just the value where the posterior acquires its maximum value. As shown already, the posterior is given by

The most challenging part of the calculation here is the derivation of a closed form for the marginal in the denominator on the right.

For a given algebraic form for the likelihood, the different forms for the prior $\text{prob}(\Theta)$ pose different levels of difficulty for the determination of the marginal in the denominator and, therefore, for the determination of the posterior.

For a given likelihood function $\text{prob}(X | \Theta)$, a prior $\text{prob}(\Theta)$ is called a conjugate prior if the posterior $\text{prob}(\Theta | X)$ has the same algebraic form as the prior. Obviously, Bayesian estimation and prediction becomes much easier should the engineering assumptions allow a conjugate prior to be chosen for the applicable likelihood function. When the likelihood can be assumed to be Gaussian, a Gaussian prior would constitute a conjugate prior because in this case the posterior would also be Gaussian.

UNIT II

Regression: Linear Regression, Ridge Regression, Lasso.

Dimensionality Reduction: Principal Component Analysis, Partial Least Squares

2.1 REGRESSION

2.1.1. Introduction

Q1. What is Regression Analysis? What are the various types of regressions used in regression analysis.

Ans : (Imp.)

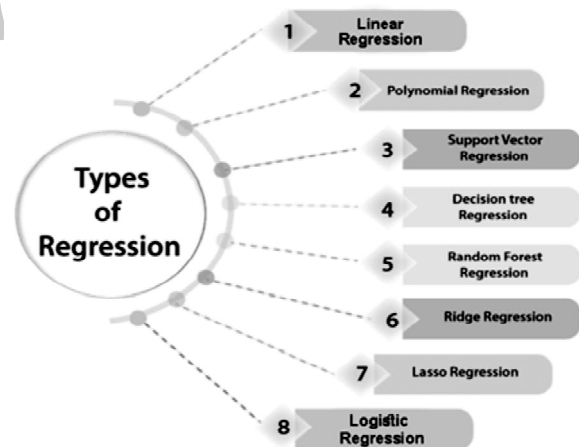
As mentioned above, Regression analysis helps in the prediction of a continuous variable. There are various scenarios in the real world where we need some future predictions such as weather condition, sales prediction, marketing trends, etc., for such case we need some technology which can make predictions more accurately. So for such case we need Regression analysis which is a statistical method and used in machine learning and data science. Below are some other reasons for using Regression analysis:

- Regression estimates the relationship between the target and the independent variable.
- It is used to find the trends in data.
- It helps to predict real/continuous values.
- By performing the regression, we can confidently determine the most important factor, the least important factor, and how each factor is affecting the other factors.

Types of Regression

There are various types of regressions which are used in data science and machine learning. Each type has its own importance on different scenarios, but at the core, all the regression methods analyze the effect of the independent variable on dependent variables. There are some important types of regression which are given below:

- Linear Regression
- Logistic Regression
- Polynomial Regression
- Support Vector Regression
- Decision Tree Regression
- Random Forest Regression
- Ridge Regression
- Lasso Regression



Q2. Explain about Linear Regression model

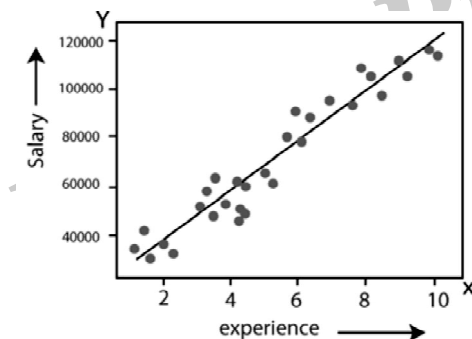
Ans :

Linear Regression

- Linear regression is a statistical regression method which is used for predictive analysis.
- It is one of the very simple and easy algorithms which works on regression and shows the relation ship between the continuous variables.

- It is used for solving the regression problem in machine learning.
- Linear regression shows the linear relationship between the independent variable (X-axis) and the dependent variable (Y-axis), hence called linear regression.
- If there is only one input variable (x), then such linear regression is called simple linear regression. And if there is more than one input variable, then such linear regression is called multiple linear regression.
- The relationship between variables in the linear regression model can be explained using the below image.

Here we are predicting the salary of an employee on the basis of the year of experience. The linear regression model provides a sloped straight line representing the relationship between the variables. Consider the below image:



Mathematically, we can represent a linear regression as:

$$y = a_0 + a_1x + \varepsilon$$

Here,

Y = Dependent Variable (Target Variable)

X = Independent Variable (predictor Variable)

a_0 = intercept of the line (Gives an additional degree of freedom)

a_1 = Linear regression coefficient (scale factor to each input value).

ε = random error

The values for x and y variables are training data sets for Linear Regression model representation.

Types of Linear Regression

Linear regression can be further divided into two types of the algorithm:

➤ Simple Linear Regression

If a single independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Simple Linear Regression.

➤ Multiple Linear regression

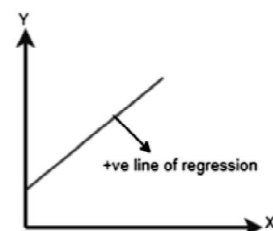
If more than one independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Multiple Linear Regression.

Linear Regression Line

A linear line showing the relationship between the dependent and independent variables is called a regression line. A regression line can show two types of relationship:

➤ Positive Linear Relationship

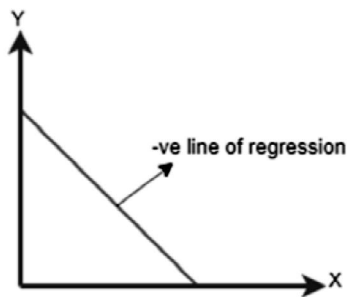
If the dependent variable increases on the Y-axis and independent variable increases on X-axis, then such a relationship is termed as a Positive linear relationship.



The line equation will be: $Y = a_0 + a_1X$

➤ Negative Linear Relationship

If the dependent variable decreases on the Y-axis and independent variable increases on the X-axis, then such a relationship is called a negative linear relationship.



The line of equation will be: $Y = -a_0 + a_1X$

Finding the best fit line

When working with linear regression, our main goal is to find the best fit line that means the error between predicted values and actual values should be minimized. The best fit line will have the least error.

The different values for weights or the coefficient of lines (a_0 , a_1) gives a different line of regression, so we need to calculate the best values for a_0 and a_1 to find the best fit line, so to calculate this we use cost function.

Cost function

- The different values for weights or coefficient of lines (a_0 , a_1) gives the different line of regression, and the cost function is used to estimate the values of the coefficient for the best fit line.
- Cost function optimizes the regression coefficients or weights. It measures how a linear regression model is performing.
- We can use the cost function to find the accuracy of the mapping function, which maps the input variable to the output variable. This mapping function is also known as Hypothesis function.

For Linear Regression, we use the Mean Squared Error (MSE) cost function, which is the average of squared error occurred between the predicted values and actual values. It can be written as:

For the above linear equation, MSE can be calculated as:

$$MSE = \frac{1}{N} \sum_{i=1}^n (y_i - (a_1x_i + a_0))^2$$

Where,

N = Total number of observation

Y_i = Actual value

$(a_1x_i + a_0)$ = Predicted value.

Residuals

The distance between the actual value and predicted values is called residual. If the observed points are far from the regression line, then the residual will be high, and so cost function will high. If the scatter points are close to the regression line, then the residual will be small and hence the cost function.

Gradient Descent

- Gradient descent is used to minimize the MSE by calculating the gradient of the cost function.
- A regression model uses gradient descent to update the coefficients of the line by reducing the cost function.
- It is done by a random selection of values of coefficient and then iteratively update the values to reach the minimum cost function.

Model Performance

The Goodness of fit determines how the line of regression fits the set of observations. The process of finding the best model out of various models is called optimization. It can be achieved by below method:

R-squared method

- R-squared is a statistical method that determines the goodness of fit.
- It measures the strength of the relationship between the dependent and independent variables on a scale of 0-100%.
- The high value of R-square determines the less difference between the predicted values and actual values and hence represents a good model.
- It is also called a coefficient of determination, or coefficient of multiple determination for multiple regression.

- It can be calculated from the below formula:

$$R\text{-squared} = \frac{\text{Explained variation}}{\text{Total Variation}}$$

Assumptions of Linear Regression

Below are some important assumptions of Linear Regression. These are some formal checks while building a Linear Regression model, which ensures to get the best possible result from the given data set.

- **Linear relationship between the features and target**

Linear regression assumes the linear relationship between the dependent and independent variables.

- **Small or no multicollinearity between the features**

Multicollinearity means high-correlation between the independent variables. Due to multicollinearity, it may difficult to find the true relationship between the predictors and target variables. Or we can say, it is difficult to determine which predictor variable is affecting the target variable and which is not. So, the model assumes either little or no multicollinearity between the features or independent variables.

- **Homoscedasticity Assumption**

Homoscedasticity is a situation when the error term is the same for all the values of independent variables. With homoscedasticity, there should be no clear pattern distribution of data in the scatter plot.

- **Normal distribution of error terms**

Linear regression assumes that the error term should follow the normal distribution pattern. If error terms are not normally distributed, then confidence intervals will become either too wide or too narrow, which may cause difficulties in finding coefficients.

It can be checked using the q-q plot. If the plot shows a straight line without any deviation, which means the error is normally distributed.

- **No autocorrelations:**

The linear regression model assumes no autocorrelation in error terms. If there will be any correlation in the error term, then it will drastically reduce the accuracy of the model. Autocorrelation usually occurs if there is a dependency between residual errors.

Q3. Elaborate Simple and Multiple Linear Regression Models.

Ans :

Simple Linear Regression

Simple Linear Regression is a type of Regression algorithms that models the relationship between a dependent variable and a single independent variable. The relationship shown by a Simple Linear Regression model is linear or a sloped straight line, hence it is called Simple Linear Regression.

The key point in Simple Linear Regression is that the dependent variable must be a continuous/real value. However, the independent variable can be measured on continuous or categorical values.

Simple Linear regression algorithm has mainly two objectives:

- Model the relationship between the two variables. Such as the relationship between Income and expenditure, experience and Salary, etc.
- Forecasting new observations. Such as Weather forecasting according to temperature, Revenue of a company according to the investments in a year, etc.

Simple Linear Regression Model

The Simple Linear Regression model can be represented using the below equation:

$$y = a_0 + a_1x + \epsilon$$

Where,

a_0 = It is the intercept of the Regression line (can be obtained putting $x=0$)

a_1 = It is the slope of the regression line, which tells whether the line is increasing or decreasing.

ε = The error term. (For a good model it will be negligible)

Multiple Linear Regression

In the previous topic, we have learned about Simple Linear Regression, where a single Independent/Predictor (X) variable is used to model the response variable (Y). But there may be various cases in which the response variable is affected by more than one predictor variable; for such cases, the Multiple Linear Regression algorithm is used.

Moreover, Multiple Linear Regression is an extension of Simple Linear regression as it takes more than one predictor variable to predict the response variable. We can define it as:

Example:

Prediction of CO₂ emission based on engine size and number of cylinders in a car.

Some key points about MLR

- For MLR, the dependent or target variable (Y) must be the continuous/real, but the predictor or independent variable may be of continuous or categorical form.
- Each feature variable must model the linear relationship with the dependent variable.
- MLR tries to fit a regression line through a multidimensional space of data-points.

MLR equation

In Multiple Linear Regression, the target variable(Y) is a linear combination of multiple predictor variables $x_1, x_2, x_3, \dots, x_n$. Since it is an enhancement of Simple Linear Regression, so the same is applied for the multiple linear regression equation, the equation becomes:

$$Y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n \dots (a)$$

Where,

Y = Output/Response variable

$b_0, b_1, b_2, b_3, \dots, b_n$ = Coefficients of the model.

$x_1, x_2, x_3, x_4, \dots$ = Various Independent/feature variable

Assumptions for Multiple Linear Regression

- A linear relationship should exist between the Target and predictor variables.
- The regression residuals must be normally distributed.
- MLR assumes little or no multi-collinearity (correlation between the independent variable) in data.

Applications of Multiple Linear Regression

There are mainly two applications of Multiple Linear Regression

- Effectiveness of Independent variable on prediction
- Predicting the impact of changes

2.1.2 Ridge Regression

Q4. What is Regularization? How does Regularization Work?

Ans : (Imp.)

- Regularization is one of the most important concepts of machine learning. It is a technique to prevent the model from overfitting by adding extra information to it.
- Sometimes the machine learning model performs well with the training data but does not perform well with the test data. It means the model is not able to predict the output when deals with unseen data by introducing noise in the output, and hence the model is called over fitted. This problem can be deal with the help of a regularization technique.
- This technique can be used in such a way that it will allow to maintain all variables or features in the model by reducing the magnitude of the variables. Hence, it maintains accuracy as well as a generalization of the model.
- It mainly regularizes or reduces the coefficient of features toward zero. In simple words, "In regularization technique, we reduce the magnitude of the features by keeping the same number of features."

Working of Regularization

Regularization works by adding a penalty or complexity term to the complex model. Let's consider the simple linear regression equation:

$$y = \beta_0 + \beta_1 \times 1 + \beta_2 \times 2 + \beta_3 \times 3 + \dots + \beta_n \times n + b$$

In the above equation, Y represents the value to be predicted

X1, X2, ...Xn are the features for Y.

$\beta_0, \beta_1, \dots, \beta_n$ are the weights or magnitude attached to the features, respectively. Here represents the bias of the model, and b represents the intercept.

Linear regression models try to optimize the β_0 and b to minimize the cost function. The equation for the cost function for the linear model is given below:

$$\sum_{i=1}^M (y_i - y'_i)^2 = \sum_{i=1}^M \left(y_i - \sum_{j=0}^n \beta_j * X_{ij} \right)^2$$

Now, we will add a loss function and optimize parameter to make the model that can predict the accurate value of Y. The loss function for the linear regression is called as RSS or Residual sum of squares.

Q5. What is Regularization? How does Regularization Work?

Ans :

- Regularization is one of the most important concepts of machine learning. It is a technique to prevent the model from over fitting by adding extra information to it.
- Sometimes the machine learning model performs well with the training data but does not perform well with the test data. It means the model is not able to predict the output when deals with unseen data by introducing noise in the output, and hence the model is called over fitted. This problem can be deal with the help of a regularization technique.
- This technique can be used in such a way that it will allow to maintain all variables or features in the model by reducing the magnitude of the variables. Hence, it maintains accuracy as well as a generalization of the model.

- It mainly regularizes or reduces the coefficient of features toward zero. In simple words, "In regularization technique, we reduce the magnitude of the features by keeping the same number of features."

Working of Regularization

Regularization works by adding a penalty or complexity term to the complex model. Let's consider the simple linear regression equation:

$$y = \beta_0 + \beta_1 \times 1 + \beta_2 \times 2 + \beta_3 \times 3 + \dots + \beta_n \times n + b$$

In the above equation, Y represents the value to be predicted X1, X2, ...Xn are the features for Y.

$\beta_0, \beta_1, \dots, \beta_n$ are the weights or magnitude attached to the features, respectively. Here represents the bias of the model, and b represents the intercept.

Linear regression models try to optimize the β_0 and b to minimize the cost function. The equation for the cost function for the linear model is given below:

$$\sum_{i=1}^M (y_i - y'_i)^2 = \sum_{i=1}^M \left(y_i - \sum_{j=0}^n \beta_j * X_{ij} \right)^2$$

Q6. What are the various techniques used for regularization ? Write about them

Or

Write about Ridge and Lasso Regressions

Ans :

Techniques of Regularization

There are mainly two types of regularization techniques, which are given below:

- Ridge Regression
- Lasso Regression

Ridge Regression

- Ridge regression is one of the types of linear regression in which a small amount of bias is introduced so that we can get better long-term predictions.

- Ridge regression is a regularization technique, which is used to reduce the complexity of the model. It is also called as L2 regularization.
- In this technique, the cost function is altered by adding the penalty term to it. The amount of bias added to the model is called Ridge Regression penalty. We can calculate it by multiplying with the lambda to the squared weight of each individual feature.
- The equation for the cost function in ridge regression will be:

$$\sum_{i=1}^M (y_i - y'_i)^2 = \sum_{i=1}^M \left(y_i - \sum_{j=0}^n \beta_j * x_{ij} \right)^2 + \lambda \sum_{j=0}^n \beta_j^2$$

- In the above equation, the penalty term regularizes the coefficients of the model, and hence ridge regression reduces the amplitudes of the coefficients that decreases the complexity of the model.
- As we can see from the above equation, if the values of λ tend to zero, the equation becomes the cost function of the linear regression model. Hence, for the minimum value of λ , the model will resemble the linear regression model.
- A general linear or polynomial regression will fail if there is high collinearity between the independent variables, so to solve such problems, Ridge regression can be used.
- It helps to solve the problems if we have more parameters than samples.

Lasso Regression

- Lasso regression is another regularization technique to reduce the complexity of the model. It stands for Least Absolute and Selection Operator.
- It is similar to the Ridge Regression except that the penalty term contains only the absolute weights instead of a square of weights.
- Since it takes absolute values, hence, it can shrink the slope to 0, whereas Ridge Regression can only shrink it near to 0.

- It is also called as L1 regularization. The equation for the cost function of Lasso regression will be:

$$\sum_{i=1}^M (y_i - y'_i)^2 = \sum_{i=1}^M \left(y_i - \sum_{j=0}^n \beta_j * x_{ij} \right)^2 + \lambda \sum_{j=0}^n |\beta_j|$$

- Some of the features in this technique are completely neglected for model evaluation.
- Hence, the Lasso regression can help us to reduce the overfitting in the model as well as the feature selection.

Key Difference between Ridge Regression and Lasso Regression

- Ridge regression is mostly used to reduce the overfitting in the model, and it includes all the features present in the model. It reduces the complexity of the model by shrinking the coefficients.
- Lasso regression helps to reduce the overfitting in the model as well as feature selection

2.2 LASSO DIMENSIONALITY REDUCTION

Q7. Write about Dimensionality Reduction Technique.

Ans :

(Imp.)

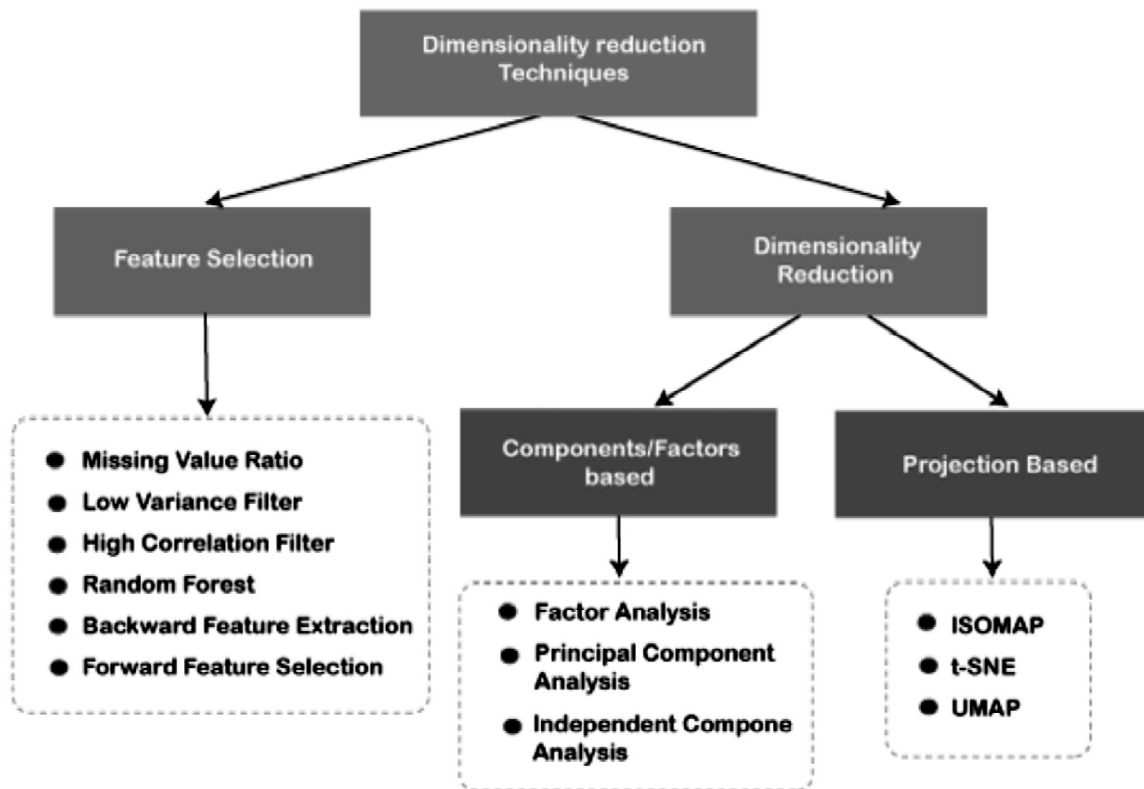
The number of input features, variables, or columns present in a given data set is known as dimensionality, and the process to reduce these features is called dimensionality reduction.

A dataset contains a huge number of input features in various cases, which makes the predictive modeling task more complicated. Because it is very difficult to visualize or make predictions for the training dataset with a high number of features, for such cases, dimensionality reduction techniques are required to use.

Dimensionality reduction technique can be defined as, "It is a way of converting the higher dimensions dataset into lesser dimensions dataset ensuring that it provides similar information." These

techniques are widely used in machine learning for obtaining a better fit predictive model while solving the classification and regression problems.

It is commonly used in the fields that deal with high-dimensional data, such as speech recognition, signal processing, bioinformatics, etc. It can also be used for data visualization, noise reduction, cluster analysis, etc.



The Curse of Dimensionality

Handling the high-dimensional data is very difficult in practice, commonly known as the curse of dimensionality. If the dimensionality of the input dataset increases, any machine learning algorithm and model becomes more complex. As the number of features increases, the number of samples also gets increased proportionally, and the chance of overfitting also increases. If the machine learning model is trained on high-dimensional data, it becomes overfitted and results in poor performance.

Hence, it is often required to reduce the number of features, which can be done with dimensionality reduction.

Benefits of applying Dimensionality Reduction

Some benefits of applying dimensionality reduction technique to the given dataset are given below:

- By reducing the dimensions of the features, the space required to store the dataset also gets reduced.
- Less Computation training time is required for reduced dimensions of features.
- Reduced dimensions of features of the dataset help in visualizing the data quickly.
- It removes the redundant features (if present) by taking care of multicollinearity.

Disadvantages of dimensionality Reduction

There are also some disadvantages of applying the dimensionality reduction, which are given below:

- Some data may be lost due to dimensionality reduction.
- In the PCA dimensionality reduction technique, sometimes the principal components required to consider are unknown.

Approaches of Dimension Reduction

There are two ways to apply the dimension reduction technique, which are given below:

I. Feature Selection

Feature selection is the process of selecting the subset of the relevant features and leaving out the irrelevant features present in a dataset to build a model of high accuracy. In other words, it is a way of selecting the optimal features from the input dataset.

Three methods are used for the feature selection:

1. Filters Methods

In this method, the dataset is filtered, and a subset that contains only the relevant features is taken. Some common techniques of filters method are:

- Correlation
- Chi-Square Test
- ANOVA
- Information Gain, etc.

2. Wrappers Methods

The wrapper method has the same goal as the filter method, but it takes a machine learning model for its evaluation. In this method, some features are fed to the ML model, and evaluate the performance. The performance decides whether to add those features or remove to increase the accuracy of the model. This method is

more accurate than the filtering method but complex to work. Some common techniques of wrapper methods are:

- Forward Selection
- Backward Selection
- Bi-directional Elimination

3. Embedded Methods

Embedded methods check the different training iterations of the machine learning model and evaluate the importance of each feature. Some common techniques of Embedded methods are:

- LASSO
- Elastic Net
- Ridge Regression, etc.

II. Feature Extraction

Feature extraction is the process of transforming the space containing many dimensions into space with fewer dimensions. This approach is useful when we want to keep the whole information but use fewer resources while processing the information.

Some common feature extraction techniques are

- (a) Principal Component Analysis
- (b) Linear Discriminant Analysis
- (c) Kernel PCA
- (d) Quadratic Discriminant Analysis

Common techniques of Dimensionality Reduction

- (a) Principal Component Analysis
- (b) Backward Elimination
- (c) Forward Selection
- (d) Score comparison
- (e) Missing Value Ratio
- (f) Low Variance Filter
- (g) High Correlation Filter

- (h) Random Forest
- (i) Factor Analysis
- (j) Auto-Encoder

(a) Principal Component Analysis (PCA)

Principal Component Analysis is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation. These new transformed features are called the Principal Components. It is one of the popular tools that is used for exploratory data analysis and predictive modeling.

PCA works by considering the variance of each attribute because the high attribute shows the good split between the classes, and hence it reduces the dimensionality. Some real-world applications of PCA are image processing, movie recommendation system, optimizing the power allocation in various communication channels.

(b) Backward Feature Elimination

The backward feature elimination technique is mainly used while developing Linear Regression or Logistic Regression model. Below steps are performed in this technique to reduce the dimensionality or in feature selection:

- In this technique, firstly, all the n variables of the given dataset are taken to train the model.
- The performance of the model is checked.
- Now we will remove one feature each time and train the model on $n-1$ features for n times, and will compute the performance of the model.
- We will check the variable that has made the smallest or no change in the performance of the model, and then we will drop that variable or features; after that, we will be left with $n-1$ features.

- Repeat the complete process until no feature can be dropped.

In this technique, by selecting the optimum performance of the model and maximum tolerable error rate, we can define the optimal number of features require for the machine learning algorithms.

(c) Forward Feature Selection

Forward feature selection follows the inverse process of the backward elimination process. It means, in this technique, we don't eliminate the feature; instead, we will find the best features that can produce the highest increase in the performance of the model. Below steps are performed in this technique:

- We start with a single feature only, and progressively we will add each feature at a time.
- Here we will train the model on each feature separately.
- The feature with the best performance is selected.
- The process will be repeated until we get a significant increase in the performance of the model.

(d) Missing Value Ratio

If a dataset has too many missing values, then we drop those variables as they do not carry much useful information. To perform this, we can set a threshold level, and if a variable has missing values more than that threshold, we will drop that variable. The higher the threshold value, the more efficient the reduction.

(e) Low Variance Filter

As same as missing value ratio technique, data columns with some changes in the data have less information. Therefore, we need to calculate the variance of each variable, and all data columns with variance lower than a given threshold are dropped because low variance features will not affect the target variable.

(f) High Correlation Filter

High Correlation refers to the case when two variables carry approximately similar information. Due to this factor, the performance of the model can be degraded. This correlation between the independent numerical variable gives the calculated value of the correlation coefficient. If this value is higher than the threshold value, we can remove one of the variables from the dataset. We can consider those variables or features that show a high correlation with the target variable.

(g) Random Forest

Random Forest is a popular and very useful feature selection algorithm in machine learning. This algorithm contains an in-built feature importance package, so we do not need to program it separately. In this technique, we need to generate a large set of trees against the target variable, and with the help of usage statistics of each attribute, we need to find the subset of features.

Random forest algorithm takes only numerical variables, so we need to convert the input data into numeric data using hot encoding.

(h) Factor Analysis

Factor analysis is a technique in which each variable is kept within a group according to the correlation with other variables, it means variables within a group can have a high correlation between themselves, but they have a low correlation with variables of other groups.

We can understand it by an example, such as if we have two variables Income and spend. These two variables have a high correlation, which means people with high income spends more, and vice versa. So, such variables are put into a group, and that group is known as the factor. The number of these factors will be reduced as compared to the original dimension of the dataset.

(i) Auto-encoders

One of the popular methods of dimensionality reduction is auto-encoder, which is a type of

ANN or artificial neural network, and its main aim is to copy the inputs to their outputs. In this, the input is compressed into latent-space representation, and output is occurred using this representation. It has mainly two parts:

- **Encoder :** The function of the encoder is to compress the input to form the latent-space representation.
- **Decoder:** The function of the decoder is to recreate the output from the latent-space representation.

2.2.1. Principal Component Analysis

Q8. What is PCA? Explain PCA algorithm.

Ans : **(Imp.)**

Principal Component Analysis is an unsupervised learning algorithm that is used for the dimensionality reduction in machine learning. It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation. These new transformed features are called the Principal Components. It is one of the popular tools that is used for exploratory data analysis and predictive modeling. It is a technique to draw strong patterns from the given dataset by reducing the variances.

PCA generally tries to find the lower-dimensional surface to project the high-dimensional data.

PCA works by considering the variance of each attribute because the high attribute shows the good split between the classes, and hence it reduces the dimensionality. Some real-world applications of PCA are image processing, movie recommendation system, optimizing the power allocation in various communication channels. It is a feature extraction technique, so it contains the important variables and drops the least important variable.

The PCA algorithm is based on some mathematical concepts such as:

- Variance and Covariance
- Eigenvalues and Eigen factors

Some common terms used in PCA algorithm:**➤ Dimensionality**

It is the number of features or variables present in the given dataset. More easily, it is the number of columns present in the dataset.

➤ Correlation

It signifies that how strongly two variables are related to each other. Such as if one changes, the other variable also gets changed. The correlation value ranges from -1 to +1. Here, -1 occurs if variables are inversely proportional to each other, and +1 indicates that variables are directly proportional to each other.

➤ Orthogonal

It defines that variables are not correlated to each other, and hence the correlation between the pair of variables is zero.

➤ Eigenvectors

If there is a square matrix M , and a non-zero vector v is given. Then v will be eigenvector if Av is the scalar multiple of v .

➤ Covariance Matrix

A matrix containing the covariance between the pair of variables is called the Covariance Matrix.

Principal Components in PCA

As described above, the transformed new features or the output of PCA are the Principal Components. The number of these PCs are either equal to or less than the original features present in the dataset. Some properties of these principal components are given below:

- The principal component must be the linear combination of the original features.
- These components are orthogonal, i.e., the correlation between a pair of variables is zero.
- The importance of each component decreases when going to 1 to n , it means the 1 PC has the most importance, and n PC will have the least importance.

Steps for PCA algorithm**1. Getting the dataset**

Firstly, we need to take the input dataset and divide it into two subparts X and Y , where X is the training set, and Y is the validation set.

2. Representing data into a structure

Now we will represent our dataset into a structure. Such as we will represent the two-dimensional matrix of independent variable X . Here each row corresponds to the data items, and the column corresponds to the Features. The number of columns is the dimensions of the dataset.

3. Standardizing the data

In this step, we will standardize our dataset. Such as in a particular column, the features with high variance are more important compared to the features with lower variance.

If the importance of features is independent of the variance of the feature, then we will divide each data item in a column with the standard deviation of the column. Here we will name the matrix as Z .

4. Calculating the Covariance of Z

To calculate the covariance of Z , we will take the matrix Z , and will transpose it. After transpose, we will multiply it by Z . The output matrix will be the Covariance matrix of Z .

5. Calculating the Eigen Values and Eigen Vectors

Now we need to calculate the eigenvalues and eigenvectors for the resultant covariance matrix Z . Eigenvectors or the covariance matrix are the directions of the axes with high information. And the coefficients of these eigenvectors are defined as the eigenvalues.

6. Sorting the Eigen Vectors

In this step, we will take all the eigenvalues and will sort them in decreasing order, which means from largest to smallest. And simultaneously sort the eigenvectors accordingly in matrix P of eigenvalues. The resultant matrix will be named as P^* .

7. Calculating the new features Or Principal Components

Here we will calculate the new features. To do this, we will multiply the P^* matrix to the Z . In the resultant matrix Z^* , each observation is the linear combination of original features. Each column of the Z^* matrix is independent of each other.

8. Remove less or unimportant features from the new dataset.

The new feature set has occurred, so we will decide here what to keep and what to remove. It means, we will only keep the relevant or important features in the new dataset, and unimportant features will be removed out.

Applications of Principal Component Analysis

- PCA is mainly used as the dimensionality reduction technique in various AI applications such as computer vision, image compression, etc.
- It can also be used for finding hidden patterns if data has high dimensions. Some fields where PCA is used are Finance, data mining, Psychology, etc.

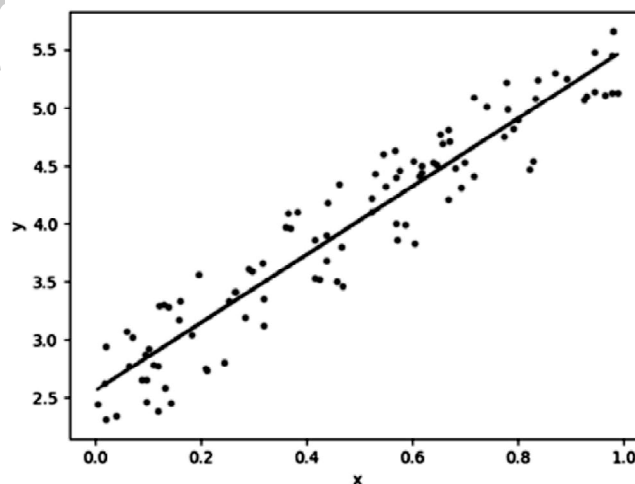
Q9. How does principal component analysis help in dimension reduction?

Ans :

Principal Component Analysis(PCA) is one of the most popular linear dimension reduction algorithms. It is a projection based method that transforms the data by projecting it onto a set of orthogonal(perpendicular) axes.

"PCA works on a condition that while the data in a higher-dimensional space is mapped to data in a lower dimension space, the variance or spread of the data in the lower dimensional space should be maximum."

In the below figure the data has maximum variance along the red line in two-dimensional space.



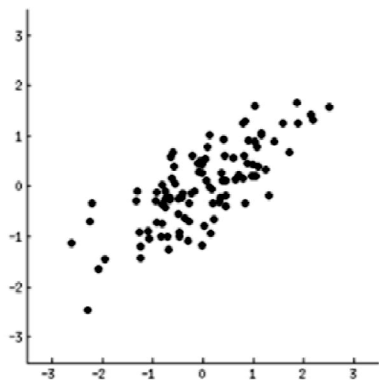
Intuition

Let's develop an intuitive understanding of PCA. Suppose, you wish to distinguish between different food items based on their nutritional content. Which variable will be a good choice to differentiate food items? If you choose a variable that varies a lot from one food item to another, you will be able to detach them properly

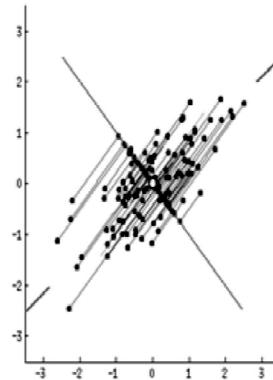
We can create an artificial variable through a linear combination of original variables like $\text{New_Var} = 4 \cdot \text{Var1} - 4 \cdot \text{Var2} + 5 \cdot \text{Var3}$.

This is what essentially PCA does, it finds the best linear combinations of the original variables so that the variance or spread along the new variable is maximum.

Suppose we have to transform a 2-dimensional representation of data points to a one-dimensional representation. So we will try to find a straight line and project data points on them. (A straight line is one dimensional). There are many possibilities to select a straight line.



Dataset



PCA in act on

Say the magenta line will be our new dimension.

If you see the red lines (connecting the projection of blue points on a magenta line) i.e. the perpendicular distance of each data point from the straight line is the projection error. The sum of the error of all data points will be the total projection error.

Our new data points will be the projections (red points) of those original blue data points. As we can see we have transformed 2-dimensional data points to one-dimensional data points by projection them on 1-dimensional space i.e. a straight line. That magenta straight line is called the principal axis. Since we are projecting to a single dimension, we have only one principal axis. We apply the same procedure to find the next principal axis from the residual variance. Apart from being the direction of maximum variance, the next principal axis must be orthogonal(perpendicular or Uncorrelated to each other,) to the other principal axes.

Once, we get all the principal axes, the dataset is projected onto these axes. The columns in the projected or transformed dataset are called principal components.

The principal components are essentially the linear combinations of the original variables, the weights vector in this combination is actually the eigenvector found which in turn satisfies the principle of least squares.

Linear algebra there are the two main procedures used in PCA to reduce dimensionality.

1. Eigenvalue Decomposition (EVD)
2. Singular Value Decomposition (SVD)

1. Eigenvalue Decomposition [EVD]

Matrix Decomposition is a process in which a matrix is reduced to its constituent parts to simplify a range of more complex operations. Eigenvalue Decomposition is the most used matrix decomposition method which involves decomposing a square matrix ($n \times n$) into a set of eigenvectors and eigenvalues.

Eigenvectors are unit vectors, which means that their length or magnitude is equal to 1.0. They are often referred to as right vectors, which simply means a column vector (as opposed to a row vector or a left vector).

Eigenvalues are coefficients applied to eigenvectors that give the vectors their length or magnitude. For example, a negative eigenvalue may reverse the direction of the eigenvector as part of scaling it.

Mathematically, A vector is an eigenvector of a matrix any $n \times n$ square matrix A if it satisfies the following equation:

$$A \cdot v = \lambda \cdot v$$

This is called the eigenvalue equation, where A is an $n \times n$ parent square matrix that we are decomposing, v is the eigenvector of the matrix, and λ represents the eigenvalue scalar.

In simpler words, the linear transformation of a vector v by A has the same effect of scaling the vector by factor λ . Note that for $m \times n$ non-square matrix A with $m \neq n$, $A \cdot v$ an m -D vector but $\lambda \cdot v$ is an n -D vector, i.e., no eigenvalues and eigenvectors are defined. If you wanna diver deeper into mathematics.

$$\begin{array}{c} \text{Original} \\ \text{Matrix} \end{array} \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} = \begin{array}{c} \text{Eigenvectors} \\ \text{Matrix} \end{array} \begin{bmatrix} -1 & -1 \\ 2 & 1 \end{bmatrix} \begin{array}{c} \text{Eigenvalues} \\ \text{Matrix} \end{array} \begin{bmatrix} -2 & 0 \\ 0 & -1 \end{bmatrix} \begin{array}{c} \text{Inverse of} \\ \text{Eigenvectors} \\ \text{Matrix} \end{array} \begin{bmatrix} 1 & 1 \\ -2 & -1 \end{bmatrix}$$

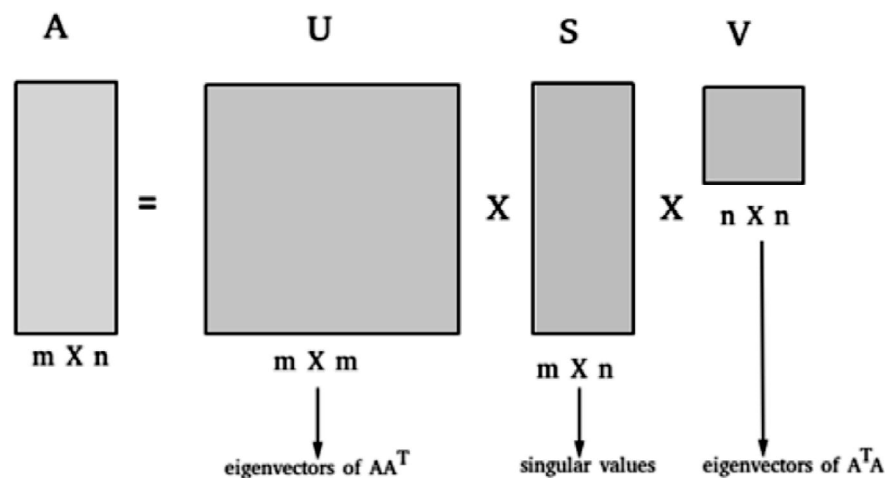
Eigenvalue decomposition

Multiplying these constituent matrices together, or combining the transformations represented by the matrices will result in the original matrix.

A decomposition operation does not result in a compression of the matrix; instead, it breaks it down into constituent parts to make certain operations on the matrix easier to perform. Like other matrix decomposition methods, Eigendecomposition is used as an element to simplify the calculation of other more complex matrix operations.

2. Singular Value Decomposition (SVD)

Singular value decomposition is a method of decomposing a matrix into three other matrices.



SVD

Mathematically Singular value decomposition is given by:

$$A = USV^T$$

Where:

- A is an $m \times n$ matrix
- U is an $m \times m$ orthogonal matrix
- S is an $n \times n$ diagonal matrix
- V is an $n \times n$ orthogonal matrix

As shown, SVD produces three matrices U , S & V . U and V orthogonal matrices whose columns represent eigenvectors of AA^T and $AA^T A$ respectively. The matrix S is a diagonal matrix and diagonal values are called singular values. Each singular value is the square-root of the corresponding eigenvalue.

Q10. How does dimension reduction fit into these mathematical equations?

Ans :

Well, once you have calculated eigenvalues and eigenvectors choose the important eigenvectors to form a set of principal axes.

Selection of EigenVectors

The importance of an eigenvector is measured by the percentage of total variance explained by the corresponding eigenvalue. Suppose V_1 & V_2 are two eigenvectors with 40% & 10% of total variance along with their directions respectively. If asked to pick one from these two eigenvectors, our choice would be V_1 because it gives us more information about data.

All eigenvectors are arranged according to their eigenvalues in descending order. Now, we have to decide how many eigenvectors to retain and for that we need to discuss two methods Total variance explained and Scree Plot for that.

Total Variance Explained

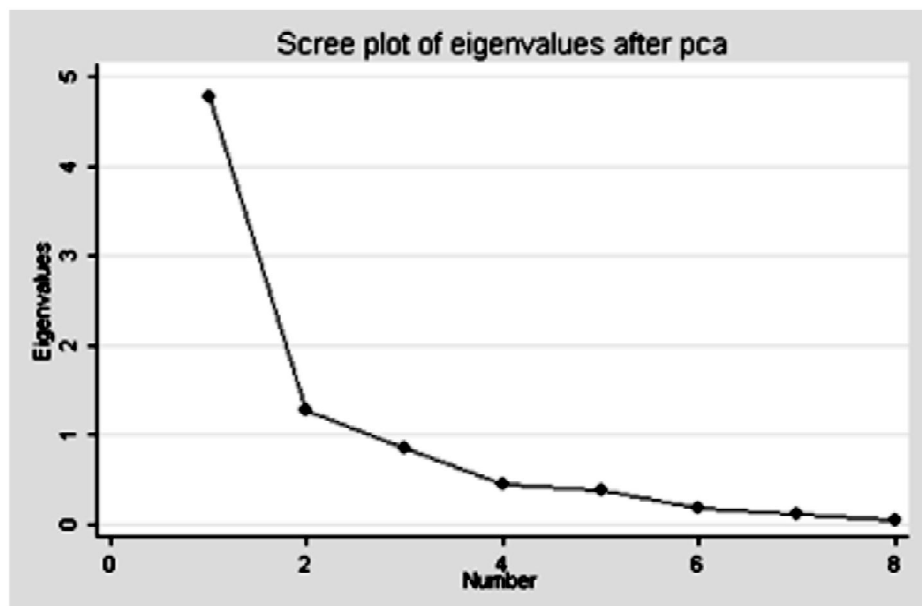
Total Explained variance is used to measure the discrepancy between a model and actual data. It is the part of the model's total variance that is explained by factors that are present.

Suppose, we have a vector of n eigenvalues(e_0, \dots, e_n) sorted in descending order. Take the cumulative sum of eigenvalues at every index until the sum is greater than 95% of the total variance. Reject all eigenvalues and eigenvectors after that index.

Scree Plot

From the scree plot, we can read off the percentage of the variance in the data explained as we add principal components.

It shows the eigenvalues on the y-axis and the number of factors on the x-axis. It always displays a downward curve. The point where the slope of the curve is leveling off (the "elbow") indicates the number of factors.



Scree plot

For example in the scree plot shown above the sharp bend(elbow) is at 4. So, the number of principal axes should be 4.

2.2.2. Partial Least Squares

Q11. How PLS techniques is used in linear decomposition? Expalin

Ans :

(Imp.)

PLS is also a feature reduction method but it offers a supervised alternative to PCR. The new set of features are also the linear combinations of the original regressors, but in computing them, the method makes use of the target variable. As a result, the results of this technique not only explain the linear combinations of original features but the response variable as well.

The model first requires the standardization of all predictors. PLS then starts computing the first linear combination of features by setting constants used in computing the Z values (Z values represent the linear combinations of original predictors) equal to the coefficient of simple OLS regression between the target and a specific regressor. PLS puts more weight on variables that are more correlated with the target variable.

Description of the Technique

Assume X is an $n \times p$ matrix and Y is a $n \times q$ matrix. The PLS technique works by successively extracting factors from both X and Y such that covariance between the extracted factors is maximized. PLS method can work with multivariate response variables (i.e., when Y is an $n \times q$ vector with $q > 1$). However, for our purpose we will assume that we have a single response (target) variable i.e., Y is $n \times 1$ and X is $n \times p$, as before.

PLS technique tries to find a linear decomposition of X and Y such that $X = TPT' + E$ and $Y = UQT' + F$, where

$$T \ n \times r = X \text{ - scores } U \ n \times r = Y \text{ - scores}$$

$$P \ p \times r = X \text{ - loadings } Q \ 1 \times r = Y \text{ - loadings}$$

$$E \ n \times p = X \text{ - residuals}$$

$$F \ n \times 1 = Y \text{ - residuals (1)}$$

Decomposition is finalized so as to maximize covariance between T and U . There are multiple algorithms available to solve the PLS problem. However, all algorithms follow an iterative process to extract the X -scores and Y -scores.

The factors or scores for X and Y are extracted successively and the number of factors extracted (r) depends on the rank of X and Y . In our case, Y is a vector and all possible X factors will be extracted.

Eigen Value Decomposition Algorithm

Each extracted x -score are linear combinations of X . For example, the first extracted x -score t of X is of the form $t = Xw$, where w is the eigen vector corresponding to the first eigen value of $XTYTX$. Similarly the first y -score is $u = Yc$, where c is the eigen vector corresponding to the first eigen value of $YTXXTY$. Note that XTY denotes the covariance of X and Y .

Once the first factors have been extracted we deflate the original values of X and Y as,

$$X_1 = X - tTX \text{ and } Y_1 = Y - tTY. \quad (2)$$

The above process is now repeated to extract the second PLS factors.

The process continues until we have extracted all possible latent factors t and u , i.e., when X is reduced to a null matrix. The number of latent factors extracted depends on the rank of X .

Numerical Example For PLS

In this section we will illustrate how to use the PLS technique to obtain X -scores that will then be used in regression. The data we used for this numerical example is the same as we used for the last numerical example of PCA. The target variable and all the predictive variables used in the last numerical example will be also used in this numerical example.

Partial Least Squares

As we described in the last section, PLS tries to find a linear decomposition of X and Y such that

$$X = TPT' + E \text{ and } Y = UQT' + F, \text{ where}$$

$$T = X\text{-scores } U = Y\text{-scores}$$

$$P = X\text{-loadings } Q = Y\text{-loadings}$$

$$E = X\text{-residuals } F = Y\text{-residuals}$$

Decomposition is finalized so as to maximize covariance between T and U . The PLS algorithm works in the same fashion whether Y is single response or multi-response.

Note that the PLS algorithm automatically predicts Y using the extracted Y -scores (U).

However, our aim here is just to obtain the X -scores (T) from the PLS decomposition and use them separately for a regression to predict Y . This provides us the flexibility to use PLS to extract orthogonal factors from X while not restricting ourselves to the original model of PLS.

UNIT III

Classification

Linear Classification, Logistic Regression, Linear Discriminant Analysis, Quadratic Discriminant Analysis, Perceptron, Support Vector Machines + Kernels, Artificial Neural Networks + Back Propagation, Decision Trees, Bayes Optimal Classifier, Naive Bayes.

3.1 CLASSIFICATION

Q1. What is the Classification Algorithm in Machine Learning? Explain.

Ans : (Imp.)

Classification Algorithm in Machine Learning

Supervised Machine Learning algorithm can be broadly classified into Regression and Classification Algorithms. In Regression algorithms, we have predicted the output for continuous values, but to predict the categorical values, we need Classification algorithms.

The Classification algorithm is a Supervised Learning technique that is used to identify the category of new observations on the basis of training data. In Classification, a program learns from the given dataset or observations and then classifies new observation into a number of classes or groups. Such as, Yes or No, 0 or 1, Spam or Not Spam, cat or dog, etc. Classes can be called as targets/labels or categories.

Unlike regression, the output variable of Classification is a category, not a value, such as "Green or Blue", "fruit or animal", etc. Since the Classification algorithm is a Supervised learning technique, hence it takes labeled input data, which means it contains input with the corresponding output.

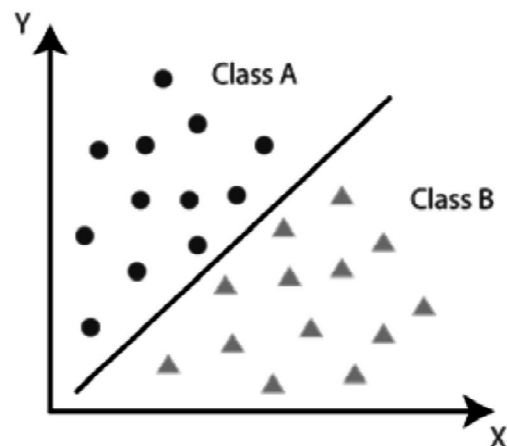
In classification algorithm, a discrete output function(y) is mapped to input variable(x).

$y = f(x)$, where y = categorical output

The best example of an ML classification algorithm is Email Spam Detector.

The main goal of the Classification algorithm is to identify the category of a given dataset, and these algorithms are mainly used to predict the output for the categorical data.

Classification algorithms can be better understood using the below diagram. In the below diagram, there are two classes, class A and Class B. These classes have features that are similar to each other and dissimilar to other classes.



The algorithm which implements the classification on a dataset is known as a classifier. There are two types of Classifications:

➤ Binary Classifier

If the classification problem has only two possible outcomes, then it is called as Binary Classifier.

Examples: YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.

➤ **Multi-class Classifier**

If a classification problem has more than two outcomes, then it is called as Multi-class Classifier.

Example: Classifications of types of crops, Classification of types of music.

Learners in Classification Problems

In the classification problems, there are two types of learners:

1. Lazy Learners

Lazy Learner firstly stores the training dataset and wait until it receives the test dataset. In Lazy learner case, classification is done on the basis of the most related data stored in the training dataset. It takes less time in training but more time for predictions.

Example: K-NN algorithm, Case-based reasoning

2. Eager Learners

Eager Learners develop a classification model based on a training dataset before receiving a test dataset. Opposite to Lazy learners, Eager Learner takes more time in learning, and less time in prediction.

Example: Decision Trees, Naïve Bayes, ANN.

Types of ML Classification Algorithms

Classification Algorithms can be further divided into the Mainly two category:

➤ **Linear Models**

- Logistic Regression
- Support Vector Machines

➤ **Non-linear Models**

- K-Nearest Neighbours
- Kernel SVM
- Naive Bayes
- Decision Tree Classification
- Random Forest Classification

Evaluating a Classification model:

Once our model is completed, it is necessary to evaluate its performance; either it is a Classification or Regression model. So for evaluating a Classification model, we have the following ways:

1. Log Loss or Cross-Entropy Loss

- It is used for evaluating the performance of a classifier, whose output is a probability value between the 0 and 1.
- For a good binary Classification model, the value of log loss should be near to 0.
- The value of log loss increases if the predicted value deviates from the actual value.
- The lower log loss represents the higher accuracy of the model.

- For Binary classification, cross-entropy can be calculated as:

$$(y \log(p) + (1/y) \log(1/p))$$

Where y = Actual output, p = predicted output.

2. Confusion Matrix

- The confusion matrix provides us a matrix/table as output and describes the performance of the model.
- It is also known as the error matrix.
- The matrix consists of predictions result in a summarized form, which has a total number of correct predictions and incorrect predictions. The matrix looks like as below table:

	Actual Positive	Actual Negative
Predicted Positive	True Positive	False Positive
Predicted Negative	False Negative	True Negative

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{Total Population}}$$

3. AUC-ROC curve

- ROC curve stands for Receiver Operating Characteristics Curve and AUC stands for Area Under the Curve.
- It is a graph that shows the performance of the classification model at different thresholds.
- To visualize the performance of the multi-class classification model, we use the AUC-ROC Curve.
- The ROC curve is plotted with TPR and FPR, where TPR (True Positive Rate) on Y-axis and FPR (False Positive Rate) on X-axis.

Use cases of Classification Algorithms

Classification algorithms can be used in different places. Below are some popular use cases of Classification Algorithms:

- Email Spam Detection
- Speech Recognition
- Identifications of Cancer tumor cells.
- Drugs Classification
- Biometric Identification, etc.

3.1.1. linear classification

Q2. Write about linear classification and its types.

Ans :

Linear classifiers classify data into labels based on a linear combination of input features. Therefore, these classifiers separate data using a line or plane or a hyperplane (a plane in more than 2 dimensions). They can only be used to classify data that is linearly separable. They can be modified to classify non-linearly separable data.

There are mainly three major types in linear binary classification.

1, Perceptron

In Perceptron, we take weighted linear combination of input features and pass it through a thresholding function which outputs 1 or 0. The sign of $w^T x$ tells us which side of the plane $w^T x = 0$, the point x lies on. Thus by taking threshold as 0, perceptron classifies data based on which side of the plane the new point lies on.

The task during training is to arrive at the plane (defined by w) that accurately classifies the training data. If the data is linearly separable, perceptron training always converges.

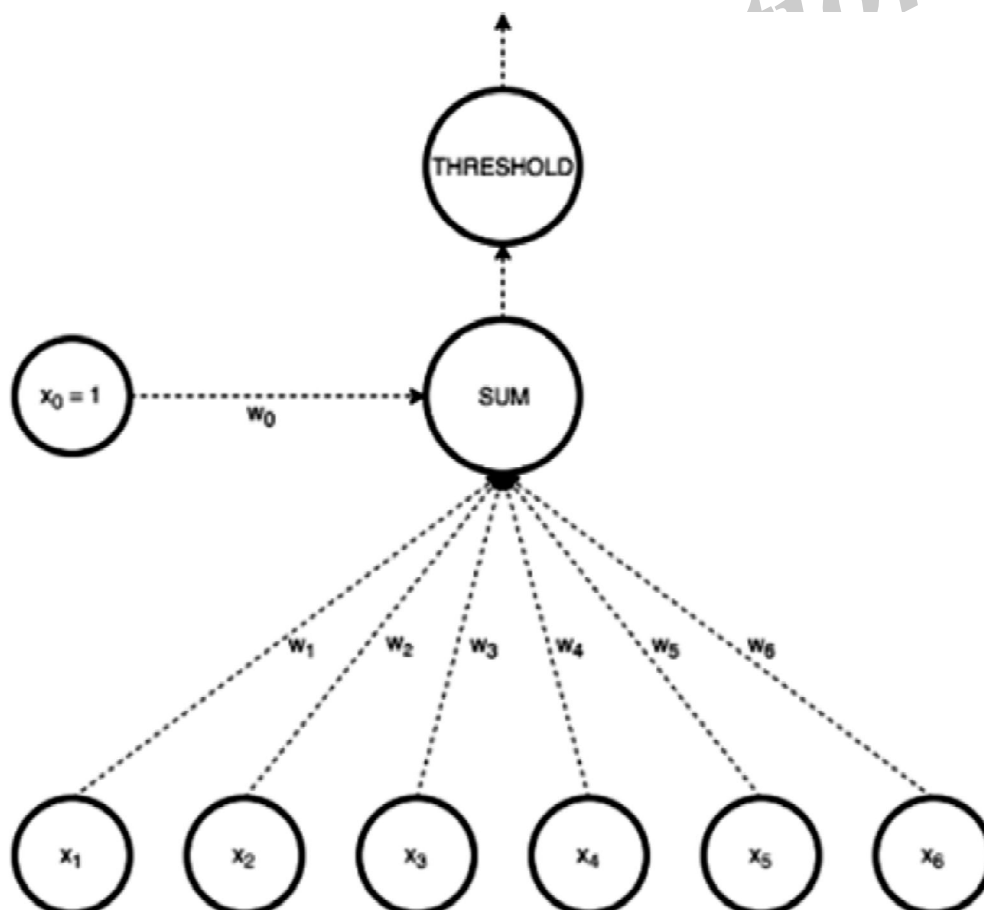


Fig. 2 : Perceptron

2. Logistic Regression

In Logistic regression, we take weighted linear combination of input features and pass it through a sigmoid function which outputs a number between 1 and 0. Unlike perceptron, which just tells us which side of the plane the point lies on, logistic regression gives a probability of a point lying on a particular side of the plane. The probability of classification will be very close to 1 or 0 as the point goes far away from the plane. The probability of classification of points very close to the plane is close to 0.5

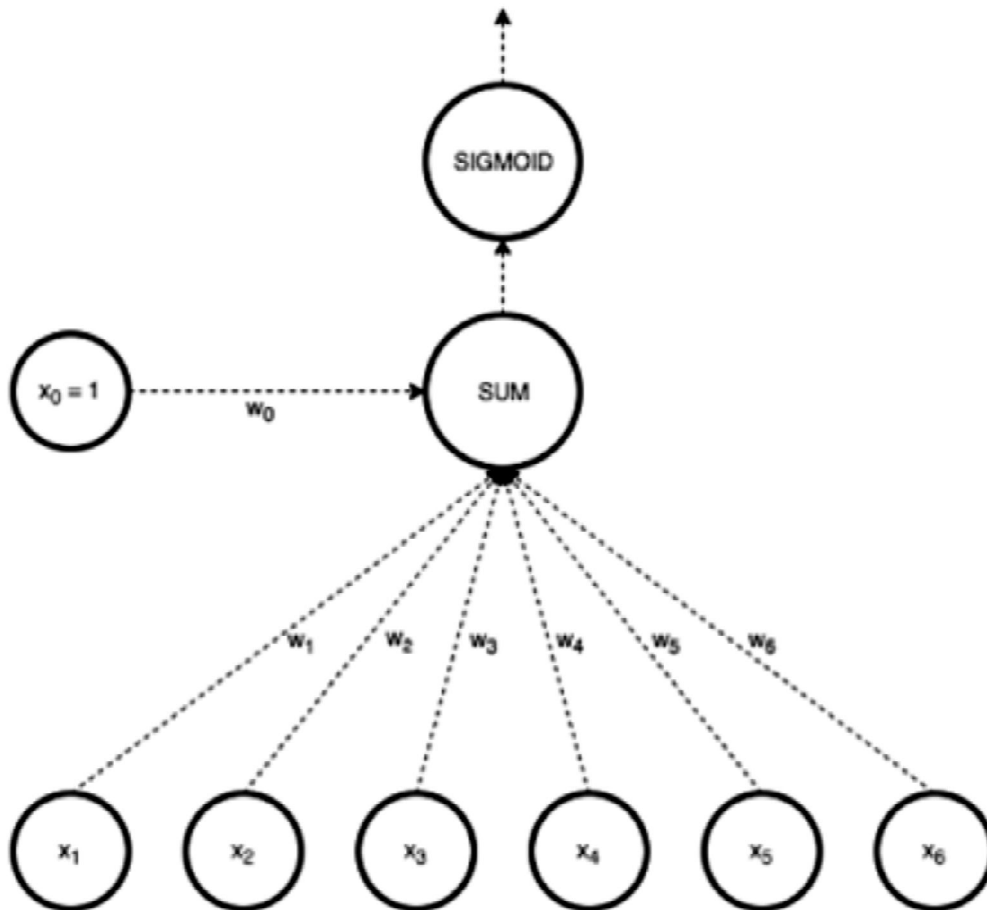


Fig. 2: Logistic Regression

3. SVM

There can be multiple hyperplanes that separate linearly separable data. SVM calculates the optimal separating hyperplane using concepts of geometry

Binary Classification Problems

As an example of a typical of a binary classification problem let us consider:

- A sequence of N
- data points $x^{(i)} \in \mathbb{R}^n, 1 \leq i \leq N$;
- each having n characteristic features $x^{(i)} = (x_1^{(i)}, x_2^{(i)}, \dots, x_n^{(i)})$;

- and the task to assign to each element $x^{(i)}$ a label $y^{(i)} \in \{-1, +1\}$;
- thereby dividing the data points into two classes labeled -1 and +1.

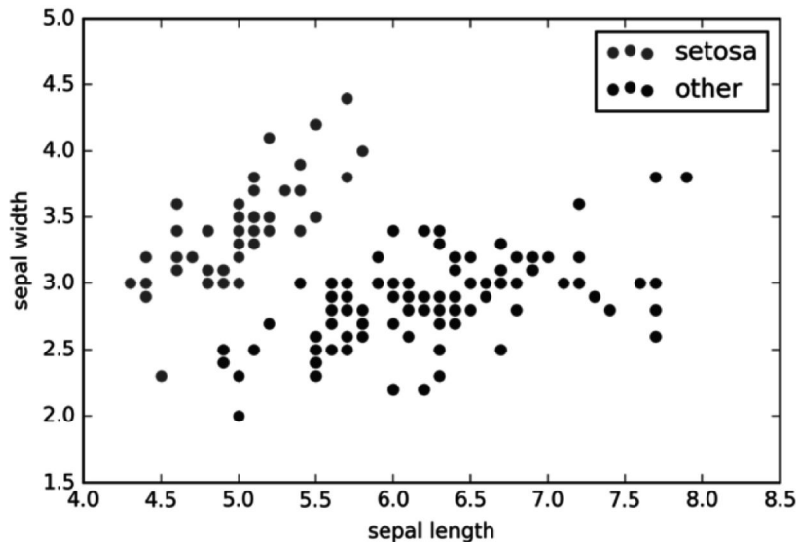


Figure : 3

Fig. 3 Labelled $n=2$ dimensional example data points ($x^{(i)} \in \mathbb{R}^2$) describing the sepal length and sepal width, i.e., $x_1^{(i)}$ and $x_2^{(i)}$, respectively, of species of the Iris flower. The class label names 'setosa' and 'other', i.e., $y^{(i)} = -1$ and $y^{(i)} = +1$, respectively, are encoded in the colors red and blue.

The goal of the classification problem is, given some pre-labeled training data:

$$(x^{(i)}, y^{(i)})_{1 \leq i \leq M}, \quad M < N$$

to make the machine find a function $f: \mathbb{R}^n \rightarrow \{-1, +1\}$

that:

- predicts accurately the labels of pre-labeled training data $(x^{(i)}, y^{(i)})_{1 \leq i \leq M}$, $M < N$ i.e., for most indices $1 \leq i \leq M$ it should hold $f(x^{(i)}) = y^{(i)}$;
- and generalizes well the remaining data points $x^{(i)}$, for $1 > i \leq M$ or even completely unknown data.

A general approach to this problem is to specify a space of candidates for f , the hypotheses set. Then the art of the game is to find sensible and mathematical precise objects encoding the vague expressions 'accurately', 'most', and 'generalizes' and to find, in that sense, an optimal functions f .

- Typically one tries to find an adequate coordinization of the hypotheses set, so that the search for an 'optimal' f can be recast into a search for finitely many 'optimal' coordinates – one often refers to the choice of coordinization and potential functions f as the 'model' and to the particular coordinate as the 'parameters of the model';
- In which sense parameters are better or worse than others is usually encoded by a non-negative function on the set of possible parameters and the entire training data set, often called 'loss', 'regret', 'energy' or 'error' function;

- The search for optimal parameters is then recast into a search of minima of this loss function.

The following plot shows the data points of the iris data set shown above with a possible hyperplane as decision boundary between the two different classes.

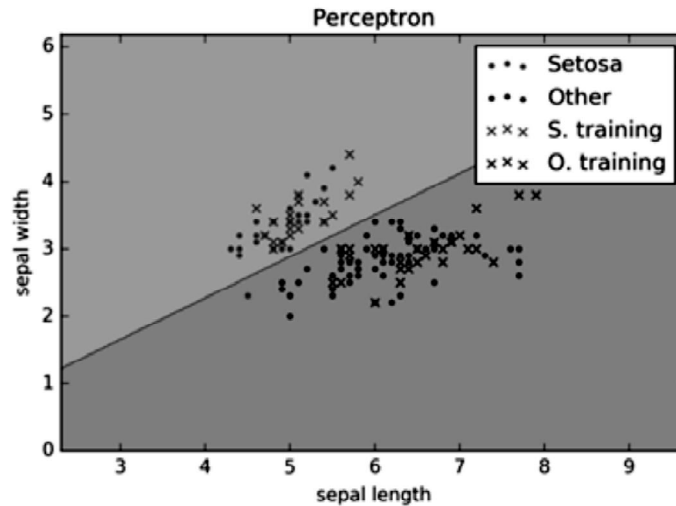


Figure : 4

Fig. 4 Decision boundaries for a possible classification function f . The dots denote unknown data points, e.g., $x^{(i)}$, for $M < i \leq N$ and the crosses denote pre-labeled data points, $x^{(i)}$, for $i \leq M$, which were used to train the model in order to find an optimal f .

Note that in Fig. 4, although the classification of the pre-labeled data points (the crosses) seems to be perfect, the classification of the unknown data (the dots) is not. This may be due to the following reasons:

- the data is simply not separable using just a hyperplane, i.e., it is a non-linear classification problem,
- there are errors in the pre-labeled data,
- or the classifier function f is not optimal yet.

It is quite a typical situation that a perfect classification is not possible. It is therefore important to specify mathematically in which sense we allow for errors and what can be done to minimize them – this will be encoded in the mathematical sense given to the expressions ‘accurately’ and ‘generalizes’ that is usually encoded by means of choice in the loss function, as discussed above.

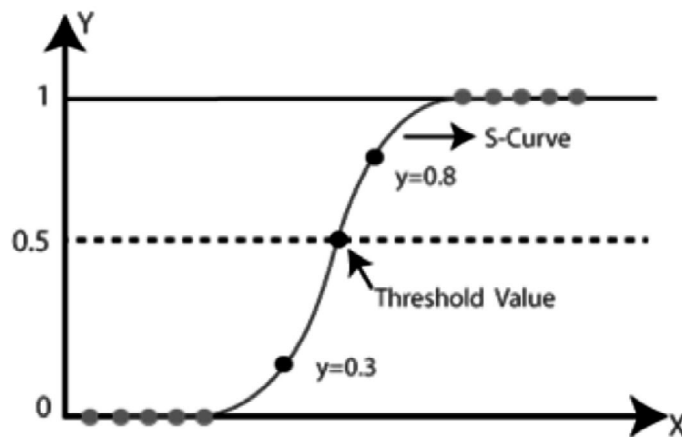
3.1.2. Logistic Regression

Q3. Write about Logistic Regression.

Ans :

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.

- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:



Logistic Function (Sigmoid Function)

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1.
- The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

Assumptions for Logistic Regression

- The dependent variable must be categorical in nature.
- The independent variable should not have multi-collinearity.

Logistic Regression Equation

The Logistic regression equation can be obtained from the Linear Regression equation. The mathematical steps to get Logistic Regression equations are given below:

- We know the equation of the straight line can be written as:

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

- In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by $(1-y)$:

$$\frac{y}{1-y}; 0 \text{ for } y = 0, \text{ and infinity for } y = 1$$

- But we need range between $-\infty$ to $+\infty$, then take logarithm of the equation it will become:

$$\log \left[\frac{y}{1-y} \right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

The above equation is the final equation for Logistic Regression.

Type of Logistic Regression

On the basis of the categories, Logistic Regression can be classified into three types:

➤ **Binomial**

In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.

➤ **Multinomial**

In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"

➤ **Ordinal**

In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

3.1.3. Linear Discriminant Analysis

Q4. How can we use LDA technique for dimensionally Reduction.

Ans : (Imp.)

Linear Discriminant Analysis or LDA is a dimensionality reduction technique. It is used as a pre-processing step in Machine Learning and applications of pattern classification. The goal of LDA is to project the features in higher dimensional space onto a lower-dimensional space in order to avoid the curse of dimensionality and also reduce resources and dimensional costs.

The original Linear Discriminant was described as a two-class technique. The multi-class version was later generalized by C.R Rao as Multiple Discriminant Analysis. They are all simply referred to as the Linear Discriminant Analysis.

LDA is a supervised classification technique that is considered a part of crafting competitive machine learning models. This category of dimensionality reduction is used in areas like image recognition and predictive analysis in marketing.

Example of LDA

Consider another simple example of dimensionality reduction and feature extraction, you want to check the quality of soap based on the information provided related to a soap including various features such as weight and volume of soap, peoples' preferential score, odor, color, contrasts, etc.

A small scenario to understand the problem more clearly;

1. **Object to be tested** : Soap
2. **To check the quality of a product**

class category as 'good' or 'bad' (dependent variable, categorical variable, measurement scale as a nominal scale)

3. **Features to describe the product**

Various parameters that describe the soap (independent variable, measurement scale as nominal, ordinal, internal scale).

When the target variable or dependent variable is decided then other related information can be dragged out from existing datasets to check the effectivity of features on the target variables.

And hence, the data dimension gets reduced out and important related-features have stayed in the new dataset.

Extensions to LDA

1. Quadratic Discriminant Analysis (QDA)

Each class deploys its own estimate of variance, or the covariance where there are multiple input variables.

2. Flexible Discriminant Analysis (FDA)

Where the combinations of non-linear sets of inputs are deployed such as splines.

3. Regularized Discriminant Analysis (RDA)

It adds regularization into the estimate of the variance, or covariance that controls the impact of various variables on LDA.

Moreover, the limitations of logistic regression can make demand for linear discriminant analysis.

Limitations of Logistic Regression

Logistics regression is a significant linear classification algorithm but also has some limitations that leads to making requirements for an alternate linear classification algorithm.

➤ Two-Class Problems

Logistic regression is proposed for two-class or binary classification problems that further be expanded for multi-class classification, but is rarely used for this purpose.

➤ Unstable With Well Separated Classes

Logistic regression is restricted and unstable when the classes are well-separated.

➤ Unstable With Few Examples

Logistic regression behaves as an unstable method while dealing with few examples from which parameters are estimated.

Linear Discriminant Analysis can handle all the above points and acts as the linear method for multi-class classification problems.

Working of Linear Discriminant Analysis

Assumptions

1. Every feature either be variable, dimension, or attribute in the dataset has gaussian distribution, i.e, features have a bell-shaped curve.
2. Each feature holds the same variance, and has varying values around the mean with the same amount on average.
3. Each feature is assumed to be sampled randomly.
4. Lack of multicollinearity in independent features and there is an increment in correlations between independent features and the power of prediction decreases.

While focusing on projecting the features in higher dimension space onto a lower dimensional space, LDA achieve this via three step process;

1. First step

To compute the separate ability amid various classes, i.e, the distance between the mean of different classes, that is also known as between-class variance.

$$S_b = \sum_{i=1}^g N_i (\bar{x}_i - \bar{x})(\bar{x}_i - \bar{x})^T$$

2. Second Step

To compute the distance among the mean and sample of each class, that is also known as the within class variance.

$$S_w = \sum_{i=1}^g (N_i - 1) S_i = \sum_{i=1}^g \sum_{j=1}^{N_i} (x_{i,j} - \bar{x}_i)(x_{i,j} - \bar{x}_i)^T$$

3. Third step

To create the lower dimensional space that maximizes the between class variance and minimizes the within class variance.

Assuming P as the lower dimensional space projection that is known as Fisher's criterion.

$$P_{lda} = \arg \max_P \frac{|P^T S_b P|}{|P^T S_w P|}$$

Application of Linear Discriminant Analysis

There are various techniques used for the classification of data and reduction in dimension, among which Principal Component Analysis(PCA) and Linear Discriminant Analysis(LDA) are commonly used techniques.

The condition where within -class frequencies are not equal, Linear Discriminant Analysis can assist data easily, their performance ability can be checked on randomly distributed test data.

This method results in the maximization of the ratio between-class variance to the within-class variance for any dataset and maximizes separability.

LDA has been successfully used in various applications, as far as a problem is transformed into a classification problem, this technique can be implemented.

For example, LDA can be used as a classification task for speech recognition, microarray data classification, face recognition, image retrieval, bioinformatics, biometrics, chemistry, etc. below are other applications of LDA;

➤ For customers' recognition

LDA helps here to identify and choose the parameters to describe the components of a group of customers who are highly likely to buy similar products.

➤ For face recognition

It is the most famous application in the field of computer vision, every face is drawn with a large number of pixel values. Here, LDA reduces the number of features to a more controllable number first before implementing the classification task. A template is created with newly produced dimensions which are a linear combination of pixel values.

➤ In medical

LDA is used here to classify the state of patients' diseases as mild, moderate or severe based on the various parameters and the

medical treatment the patient is going through in order to decrease the movement of treatment.

➤ For predictions

LDA is firmly used for prediction and hence in decision making, "will you read a book" gives you a predicted result through one or two possible class as a reading book or not.

➤ In learning

Nowadays, robots are trained to learn and talk to work as human beings, this can be treated as classification problems. LDA makes similar groups based on various parameters such as frequencies, pitches, sounds, tunes, etc.

3.1.4. Quadratic Discriminant Analysis

Q5. Write about Quadratic Discriminant analysis.

Ans : (Imp.)

QDA is a variant of LDA in which an individual covariance matrix is estimated for every class of observations. QDA is particularly useful if there is prior knowledge that individual classes exhibit distinct covariances. A disadvantage of QDA is that it cannot be used as a dimensionality reduction technique.

In QDA, we need to estimate Σ_k for each class $k \in \{1, \dots, K\}$ rather than assuming $\Sigma_k = \Sigma$ as in LDA. The discriminant function of LDA is quadratic in x :

$$\delta_k(x) = -12 \log |\Sigma_k| - 12(x - \mu_k)^T \Sigma^{-1} (x - \mu_k) + \log \pi_k.$$

Since QDA estimates a covariance matrix for each class, it has a greater number of effective parameters than LDA. We can derive the number of parameters in the following way.

➤ We need K class priors π_k . Since $\sum K_i = 1$, $\pi_k = 1$, we do not need a parameter for one of the priors. Thus, there are $K-1$ free parameters for the priors.

➤ Since there are K centroids, μ_k , with p entries each, there are Kp parameters relating to the means.

- From the covariance matrix, Σ_k , we only need to consider the diagonal and the upper right triangle. This region of the covariance matrix has $p(p+1)/2$ elements. Since K such matrices need to be estimated, there are $Kp(p+1)/2$ parameters relating to the covariance matrices.

Thus, the effective number of QDA parameters is $K-1 + Kp + Kp(p+1)/2$.

Since the number of QDA parameters is quadratic in p , QDA should be used with care when the feature space is large.

The accuracy of QDA is slightly below that of full-rank LDA. This could indicate that the assumption of common covariance is suitable for this data set.

3.1.5. Perceptron

Q6. What is Perceptron? Explain about it.

Ans :

Perceptron

Perceptron is linear binary classification algorithm. Its functioning is similar to the functioning of a single neuron in our brain. A neuron in our brain takes input signals from many other neurons and based on those inputs, it decides whether to fire or not

Algorithm

Let the input feature vector be -

$$X = [x_1, x_2, x_3, \dots, x_{n-1}, x_n]$$

A perceptron consists of a transfer function and an activation function. The transfer function takes the weighted sum of features from feature vector as input. Let the weight vector be -

$$W = [\omega_1, \omega_2, \omega_3, \dots, \omega_{n-1}, \omega_n]$$

To include a bias term, we augment the feature vector with $x_0 = 1$ and weight vector by w_0

$$X = [1, x_0, x_1, \dots, x_{n-1}, x_n]$$

$$W = [\omega_0, \omega_1, \omega_2, \dots, \omega_{n-1}, \omega_n]$$

The transfer function then becomes

$$f(\omega, x) = \sum_{i=1}^n \omega_i x_i = W^T X$$

The output of transfer function is fed to the activation function. The activation function in this case acts like a threshold

$$\text{output} = \begin{cases} 1, & f(\omega, x) \geq 0 \\ 0, & f(\omega, x) < 0 \end{cases}$$

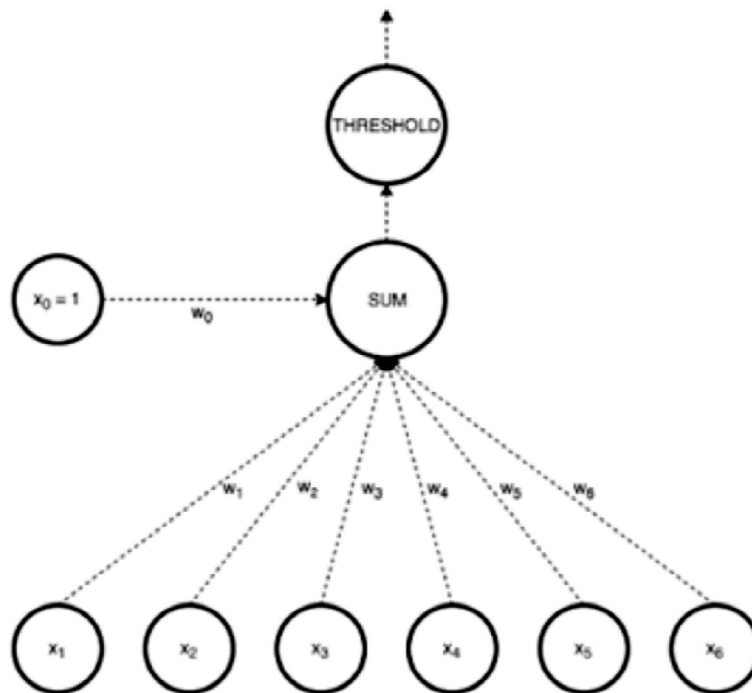


Fig.: 1. Perceptron

The problem now becomes an optimization problem where we have to find a set of weights which best classify the training data into their correct labels

Learning Weights

Perceptron is an example of online learning algorithm. The hypothesis function is learned one example at a time.

The following steps show how to learn weights in perceptron -

Until convergence

1. Randomly initialize weights
2. Calculate $f(w, X_1)$ for first sample X_1
3. Suppose the actual label of sample is Y_1 (which can be 1 or 0), calculate $Y_1 - f(w, X_1)$ as error e
4. update weights array as $W = W + eX_1$
5. Move to next sample.

A complete scan of all samples is called one epoch. For offline learning, the algorithm could go over multiple epochs to give better accuracy.

Intuition

For example, you have 2 friends who go out for a newly released movie every Friday. When they return, you ask each one of them whether the movie was good or bad and decide whether to watch the movie or not.

Friend 1's opinion	Friend 2's opinion	Your Decision
week 1 Good	Good	You watch the movie and like it
week 2 Good	Bad	You watch the movie and do not like it
week 3	Bad	Good Would you watch the movie

Since you didn't like the movie released on week 2, but friend 1 did, you are likely to give less consideration to his opinion on week 3. Suppose your opinions about movies do not match with friend 1 multiple times, you are unlikely to ever consider his opinion to decide whether to watch a movie or not.

Suppose your opinions about movies always clash, you will probably go to every movie he does not like. Thus every week you update the weights that you have assigned to each of your friend's opinion. The Perceptron algorithm works in a similar fashion.

3.1.6. Support Vector Machines + Kernels

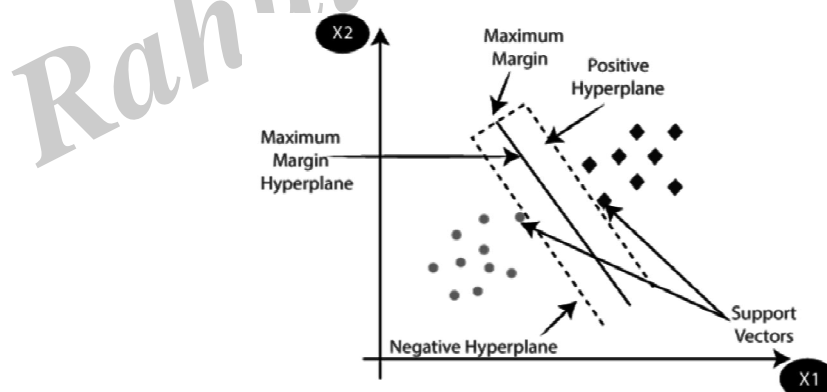
Q7. Explain Support Vector Machine Algorithm.

Ans :

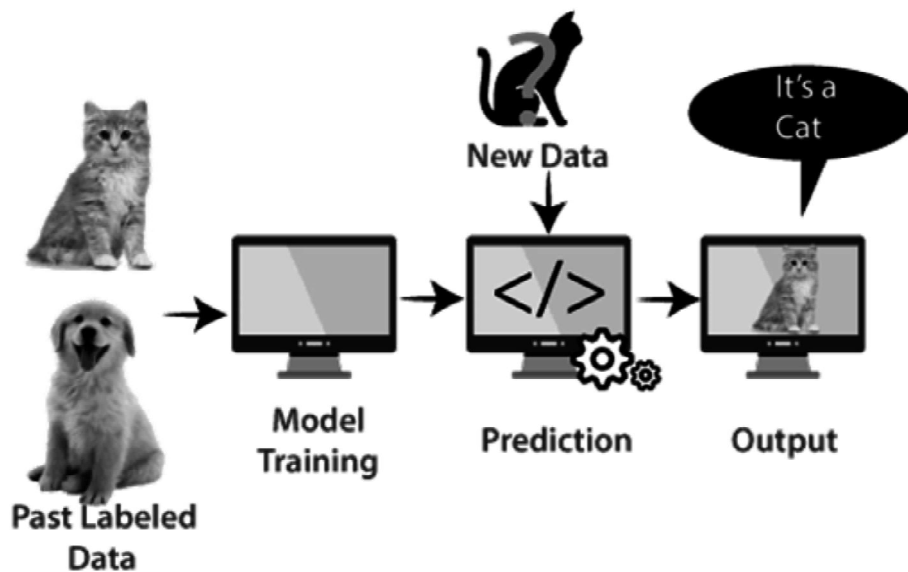
Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:



Example: SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat. Consider the below diagram:



SVM algorithm can be used for Face detection, image classification, text categorization, etc.

Types of SVM

SVM can be of two types:

➤ Linear SVM

Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

➤ Non-linear SVM

Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Hyperplane and Support Vectors in the SVM algorithm

Hyperplane

There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

Support Vectors

The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

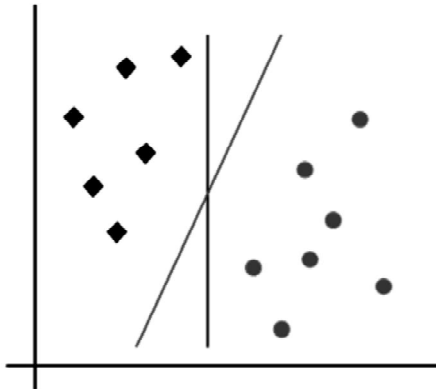
How does SVM works?

Linear SVM

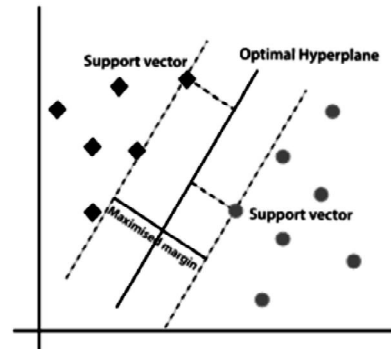
The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x_1 and x_2 . We want a classifier that can classify the pair(x_1 , x_2) of coordinates in either green or blue. Consider the below image:



So as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the below image:

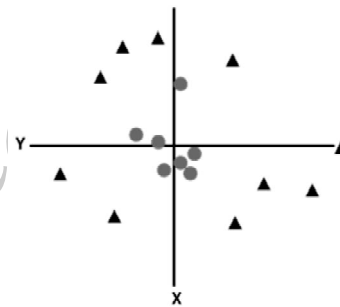


Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a hyperplane. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and the hyperplane is called as margin. And the goal of SVM is to maximize this margin. The hyperplane with maximum margin is called the optimal hyperplane.



Non-Linear SVM

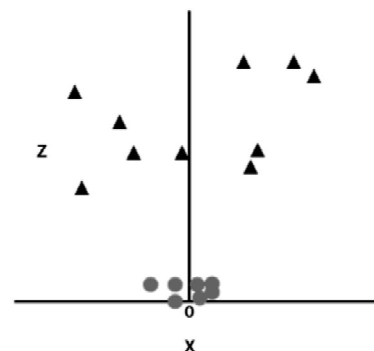
If data is linearly arranged, then we can separate it by using a straight line, but for non-linear data, we cannot draw a single straight line. Consider the below image:



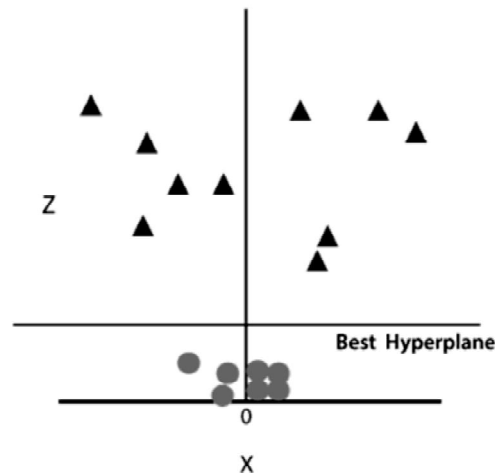
So to separate these data points, we need to add one more dimension. For linear data, we have used two dimensions x and y , so for non-linear data, we will add a third dimension z . It can be calculated as:

$$z = x^2 + y^2$$

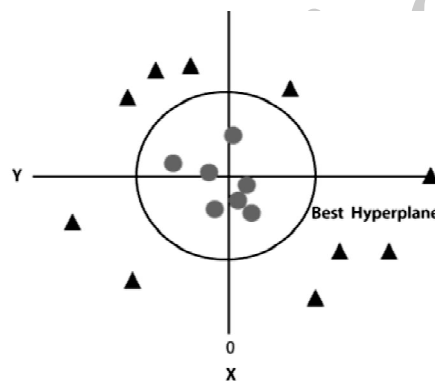
By adding the third dimension, the sample space will become as below image:



So now, SVM will divide the datasets into classes in the following way. Consider the below image:



Since we are in 3-d Space, hence it is looking like a plane parallel to the x-axis. If we convert it in 2d space with $z=1$, then it will become as:



Hence we get a circumference of radius 1 in case of non-linear data.

Q8. What are kernels? Write about how Kernels function for SVMs.

Ans :

Kernels are functions that, given x_1 and x_2 , return $(\phi(x_1), \phi(x_2))$ without ever calculating $\phi(x)$. This means we can transform feature vectors to even infinite dimensions and still not have any extra computational burden for calculating dot products!

If you observe the formulation of SVM optimization problem, all we ever need is the dot product. We never actually need the vectors individually. This makes SVM easy to solve even if we transform the features into infinite dimensions

Suppose we need a transformation

$$x = [x_1, x_2] \Rightarrow [x_1 \sqrt{2x_1 x_2}, x_2] = \phi(x)$$

$$(\phi(x), \phi(x)) = x_1^2 + 2x_1x_2 + x_2^2 = (x_1 + x_2)^2$$

Thus the kernel function in this case is just $(x_1 + x_2)^2$. Therefore instead of calculating all terms, we can just calculate $x_1 + x_2$ which is a number and square it. There was no need to calculate the higher dimensional explicitly to calculate the dot product. Imagine if the projection was on 100 dimensions instead of 3. Finding a kernel function like this would greatly simplify the computations.

Lets us see this with an example of polynomial kernel of degree 2

$$x = [x_1, x_2] \Rightarrow [1, x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2] = \phi(x)$$

We can verify that kernel function is : $(1 + x_1x_2)^2$

Here instead of calculating the full expanded version of dot product, we just have to calculate $1 + x_1x_2$ which is a number and square it

Optimization problem for SVM with non-linear kernel

Following from the derivation in linear SVM, after transforming the features to higher dimensions, the optimization problem becomes.

Since we only have to calculate the dot product of higher order features, we can use the kernel trick without increasing the computational load.

Solving the above problem gave us the Lagrange multipliers, but how do we get weights from them. Recall that

$$\omega = \sum_{i=1}^m \lambda_i y_i \phi(x_i)$$

Which means the to calculate w we need to calculate the higher dimensional vector. But remember that to classify new example, we don't need to explicitly know the weights, we just need to find out which side of the plane the point lies on. That is given by

$$\begin{aligned} & \text{sign}(\omega \phi(x) + b) \\ &= \text{sign}\left(\sum_{i=1}^m \lambda_i y_i \phi(x_i) \phi(x) + b\right) \end{aligned}$$

Which again requires just the dot product. Thus we never have to calculate anything in the higher dimensional space while using the kernel trick.

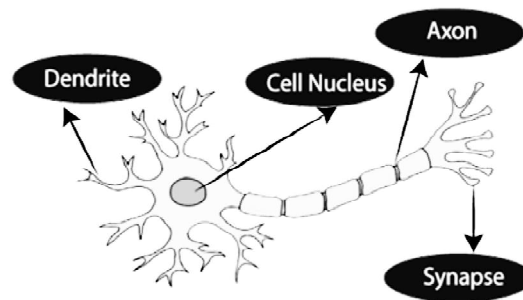
3.1.7. Artificial Neural Networks + Back Propagation

Q9. What is Artificial Neural Network? Explain about ANN with its architecture.

Ans :

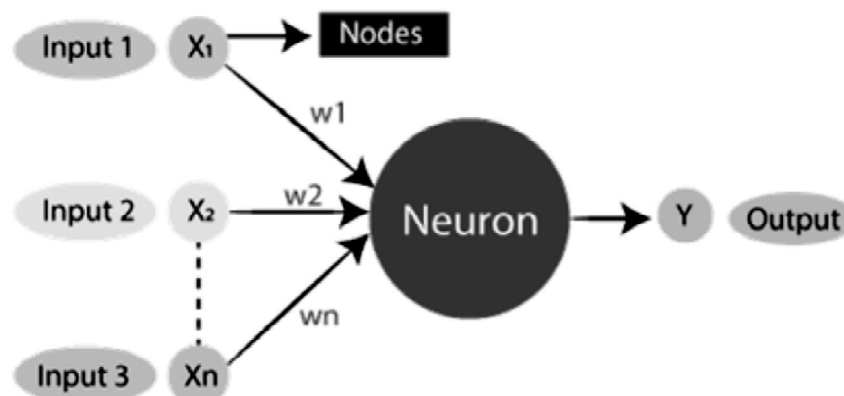
(Imp.)

The term "Artificial Neural Network" is derived from Biological neural networks that develop the structure of a human brain. Similar to the human brain that has neurons interconnected to one another, artificial neural networks also have neurons that are interconnected to one another in various layers of the networks. These neurons are known as nodes.



The given figure illustrates the typical diagram of Biological Neural Network.

The typical Artificial Neural Network looks something like the given figure.



Dendrites from Biological Neural Network represent inputs in Artificial Neural Networks, cell nucleus represents Nodes, synapse represents Weights, and Axon represents Output.

Relationship between Biological neural network and artificial neural network:

Biological Neural Network	Artificial Neural Network
Dendrites	Inputs
Cell nucleus	Nodes
Synapse	Weights
Axon	Output

An Artificial Neural Network in the field of Artificial intelligence where it attempts to mimic the network of neurons makes up a human brain so that computers will have an option to understand things and make decisions in a human-like manner. The artificial neural network is designed by programming computers to behave simply like interconnected brain cells.

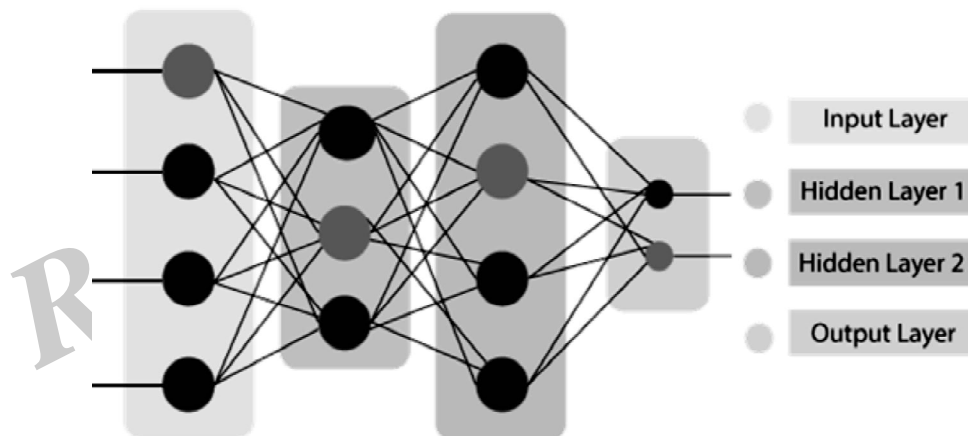
There are around 1000 billion neurons in the human brain. Each neuron has an association point somewhere in the range of 1,000 and 100,000. In the human brain, data is stored in such a manner as to be distributed, and we can extract more than one piece of this data when necessary from our memory parallelly. We can say that the human brain is made up of incredibly amazing parallel processors.

We can understand the artificial neural network with an example, consider an example of a digital logic gate that takes an input and gives an output. "OR" gate, which takes two inputs. If one or both the inputs are "On," then we get "On" in output. If both the inputs are "Off," then we get "Off" in output. Here the output depends upon input. Our brain does not perform the same task. The outputs to inputs relationship keep changing because of the neurons in our brain, which are "learning."

The architecture of an artificial neural network

To understand the concept of the architecture of an artificial neural network, we have to understand what a neural network consists of. In order to define a neural network that consists of a large number of artificial neurons, which are termed units arranged in a sequence of layers. Lets us look at various types of layers available in an artificial neural network.

Artificial Neural Network primarily consists of three layers



1. Input Layer

As the name suggests, it accepts inputs in several different formats provided by the programmer.

2. Hidden Layer

The hidden layer presents in-between input and output layers. It performs all the calculations to find hidden features and patterns.

3. Output Layer

The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer.

The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias. This computation is represented in the form of a transfer function.

$\sum_{i=1}^n W_i * X_i + b$ It determines weighted total is passed as an input to an activation function to produce

the output. Activation functions choose whether a node should fire or not. Only those who are fired make it to the output layer. There are distinctive activation functions available that can be applied upon the sort of task we are performing.

➤ **Advantages of Artificial Neural Network (ANN) Parallel processing capability**

Artificial neural networks have a numerical value that can perform more than one task simultaneously.

➤ **Storing data on the entire network**

Data that is used in traditional programming is stored on the whole network, not on a database. The disappearance of a couple of pieces of data in one place doesn't prevent the network from working.

➤ **Capability to work with incomplete knowledge**

After ANN training, the information may produce output even with inadequate data. The loss of performance here relies upon the significance of missing data.

➤ **Having a memory distribution**

For ANN is to be able to adapt, it is important to determine the examples and to encourage the network according to the desired output by demonstrating these examples to the network. The succession of the network is directly proportional to the chosen instances, and if the event can't appear to the network in all its aspects, it can produce false output.

➤ **Having fault tolerance**

Extortion of one or more cells of ANN does not prohibit it from generating output, and this feature makes the network fault-tolerance.

➤ **Disadvantages of Artificial Neural Network**

➤ **Assurance of proper network structure**

There is no particular guideline for determining the structure of artificial neural networks. The appropriate network structure is accomplished through experience, trial, and error.

➤ **Unrecognized behavior of the network**

It is the most significant issue of ANN. When ANN produces a testing solution, it does not provide insight concerning why and how. It decreases trust in the network.

➤ **Hardware dependence**

Artificial neural networks need processors with parallel processing power, as per their structure. Therefore, the realization of the equipment is dependent.

➤ **Difficulty of showing the issue to the network**

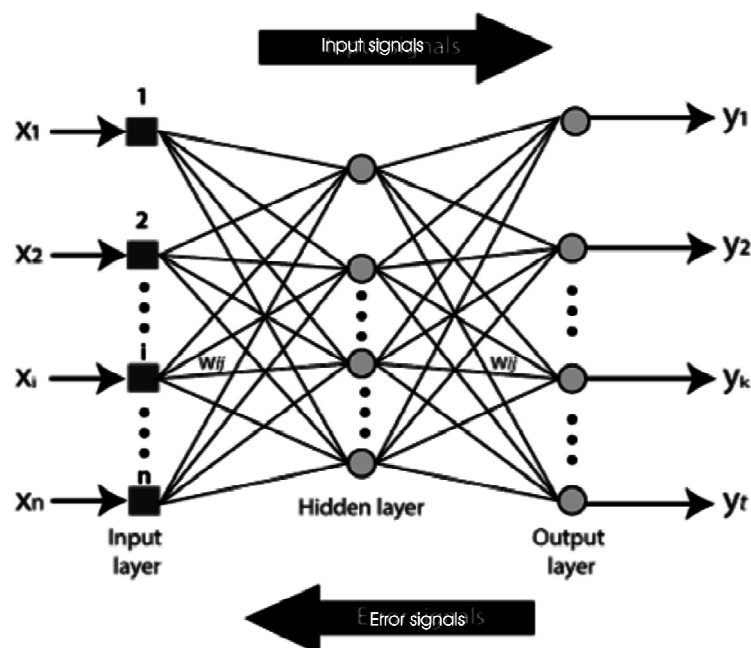
ANNs can work with numerical data. Problems must be converted into numerical values before being introduced to ANN. The presentation mechanism to be resolved here will directly impact the performance of the network. It relies on the user's abilities.

➤ **The duration of the network is unknown**

The network is reduced to a specific value of the error, and this value does not give us optimum results.

Q10. How do artificial neural networks work?*Ans :*

Artificial Neural Network can be best represented as a weighted directed graph, where the artificial neurons form the nodes. The association between the neurons outputs and neuron inputs can be viewed as the directed edges with weights. The Artificial Neural Network receives the input signal from the external source in the form of a pattern and image in the form of a vector. These inputs are then mathematically assigned by the notations $x(n)$ for every n number of inputs.



Afterward, each of the input is multiplied by its corresponding weights (these weights are the details utilized by the artificial neural networks to solve a specific problem). In general terms, these weights normally represent the strength of the interconnection between neurons inside the artificial neural network. All the weighted inputs are summarized inside the computing unit.

If the weighted sum is equal to zero, then bias is added to make the output non-zero or something else to scale up to the system's response. Bias has the same input, and weight equals to 1. Here the total of weighted inputs can be in the range of 0 to positive infinity. Here, to keep the response in the limits of the desired value, a certain maximum value is benchmarked, and the total of weighted inputs is passed through the activation function.

The activation function refers to the set of transfer functions used to achieve the desired output. There is a different kind of the activation function, but primarily either linear or non-linear sets of functions. Some of the commonly used sets of activation functions are the Binary, linear, and Tan hyperbolic sigmoidal activation functions. Let us take a look at each of them in details:

Binary

In binary activation function, the output is either a one or a 0. Here, to accomplish this, there is a threshold value set up. If the net weighted input of neurons is more than 1, then the final output of the activation function is returned as one or else the output is returned as 0.

Sigmoidal Hyperbolic

The Sigmoidal Hyperbola function is generally seen as an "S" shaped curve. Here the tan hyperbolic function is used to approximate output from the actual net input.

Types of Artificial Neural Network

There are various types of Artificial Neural Networks (ANN) depending upon the human brain neuron and network functions, an artificial neural network similarly performs tasks. The majority of the artificial neural networks will have some similarities with a more complex biological partner and are very effective at their expected tasks. For example, segmentation or classification.

Feedback ANN

In this type of ANN, the output returns into the network to accomplish the best-evolved results internally. As per the University of Massachusetts, Lowell Centre for Atmospheric Research. The feedback networks feed information back into itself and are well suited to solve optimization issues. The Internal system error corrections utilize feedback ANNs.

Feed-Forward ANN

A feed-forward network is a basic neural network comprising of an input layer, an output layer, and at least one layer of a neuron. Through assessment of its output by reviewing its input, the intensity of the network can be noticed based on group behavior of the associated neurons, and the output is decided. The primary advantage of this network is that it figures out how to evaluate and recognize input patterns.

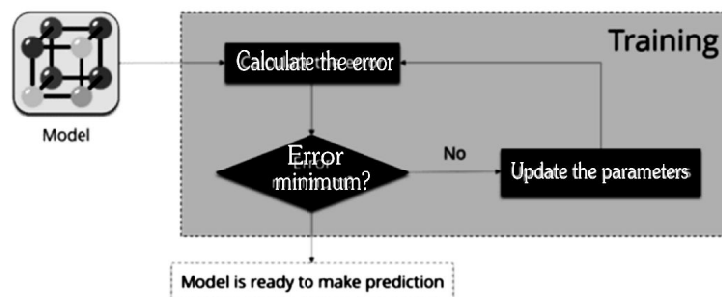
Q11. What is Backpropagation? Explain about the working of backpropagation with example.

Ans :

(Imp.)

- Backpropagation is the essence of neural network training. It is the method of fine-tuning the weights of a neural network based on the error rate obtained in the previous epoch (i.e., iteration). Proper tuning of the weights allows you to reduce error rates and make the model reliable by increasing its generalization.
- Backpropagation in neural network is a short form for "backward propagation of errors." It is a standard method of training artificial neural networks. This method helps calculate the gradient of a loss function with respect to all the weights in the network.
- While designing a Neural Network, in the beginning, we initialize weights with some random values or any variable for that fact.
- Basically, what we need to do, we need to somehow explain the model to change the parameters (weights), such that error becomes minimum.
- Let's put it in another way, we need to train our model.

One way to train our model is called as Backpropagation. Consider the diagram below:



Let me summarize the steps for you

Calculate the error – How far is your model output from the actual output.

Minimum Error – Check whether the error is minimized or not.

- Update the parameters – If the error is huge then, update the parameters (weights and biases). After that again check the error. Repeat the process until the error becomes minimum.
- Model is ready to make a prediction – Once the error becomes minimum, you can feed some inputs to your model and it will produce the output.

I am pretty sure, now you know, why we need Backpropagation or why and what is the meaning of training a model.

Let's understand how it works with an example:

You have a dataset, which has labels.

Consider the below table:

Input	Desired Output
0	0
1	2
2	4

Now the output of your model when 'W' value is 3:

Input	Desired Output	Model output (W=3)
0	0	0
1	2	3
2	4	6

Notice the difference between the actual output and the desired output:

Input	Desired Output	Model output (W=3)	Absolute Error	Square Error
0	0	0	0	0
1	2	3	1	1
2	4	6	2	4

Let's change the value of 'W'. Notice the error when 'W' = '4'

Input	Desired Output	Model output (W=3)	Absolute Error	Square Error	Model output (W=4)	Square Error
0	0	0	0	0	0	0
1	2	3	1	1	4	4
2	4	6	2	4	8	16

Now if you notice, when we increase the value of 'W' the error has increased. So, obviously there is no point in increasing the value of 'W' further. But, what happens if I decrease the value of 'W'? Consider the table below:

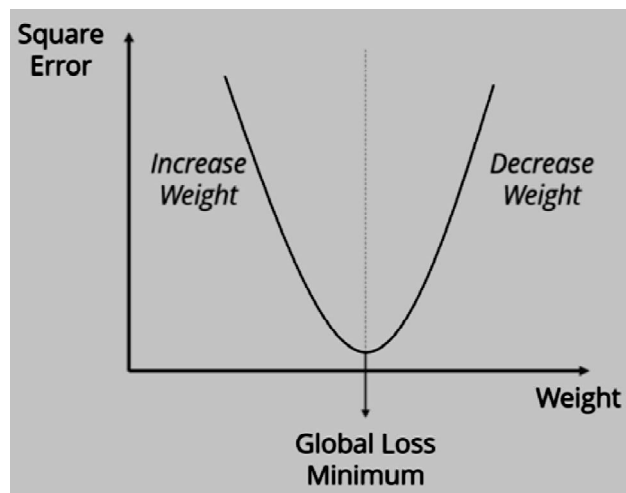
Input	Desired Output	Model output (W=3)	Absolute Error	Square Error	Model output (W=2)	Square Error
0	0	0	0	0	0	0
1	2	3	2	4	3	0
2	4	6	2	4	4	0

Now, what we did here

- We first initialized some random value to 'W' and propagated forward.
- Then, we noticed that there is some error. To reduce that error, we propagated backwards and increased the value of 'W'.
- After that, also we noticed that the error has increased. We came to know that, we can't increase the 'W' value.
- So, we again propagated backwards and we decreased 'W' value.
- Now, we noticed that the error has reduced.

So, we are trying to get the value of weight such that the error becomes minimum. Basically, we need to figure out whether we need to increase or decrease the weight value. Once we know that, we keep on updating the weight value in that direction until error becomes minimum. You might reach a point, where if you further update the weight, the error will increase. At that time you need to stop, and that is your final weight value.

Consider the graph below



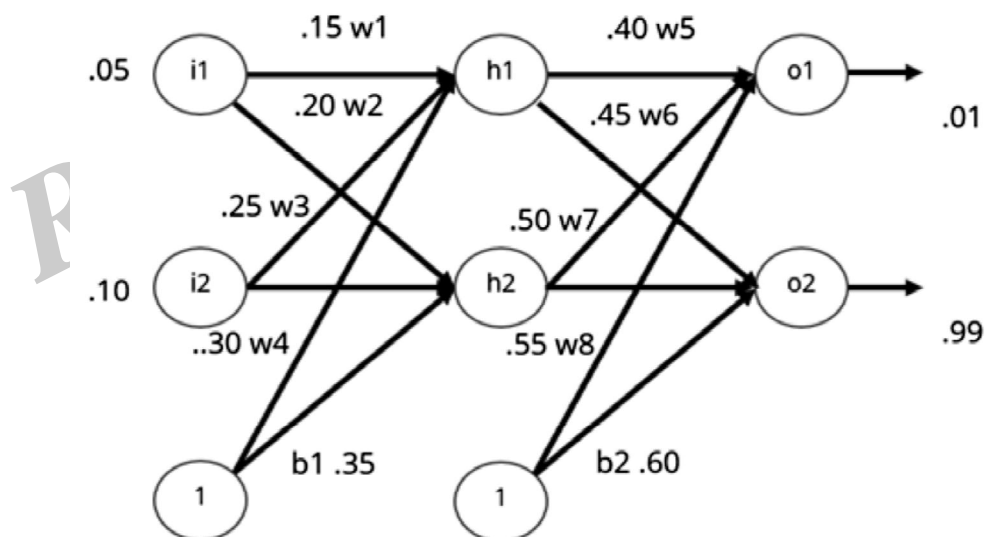
We need to reach the 'Global Loss Minimum'.

This is nothing but Backpropagation.

Let's now understand the math behind Backpropagation.

How Backpropagation Works?

Consider the below Neural Network



The above network contains the following:

- two inputs
- two hidden neurons
- two output neurons
- two biases

Below are the steps involved in Backpropagation:

Step – 1: Forward Propagation

Step – 2: Backward Propagation

Step – 3: Putting all the values together and calculating the updated weight value

Step – 1: Forward Propagation

We will start by propagating forward.

Net Input For h1:

$$\text{net h1} = w_1 * i_1 + w_2 * i_2 + b_1 * 1$$

$$\text{net h1} = 0.15 * 0.05 + 0.2 * 0.1 + 0.35 * 1 = 0.3775$$

Output Of h1:

$$\text{out h1} = 1 / 1 + e^{-\text{net h1}}$$

$$1 / 1 + e^{-0.3775} = 0.593269992$$

Output Of h2:

$$\text{out h2} = 0.596884378$$

We will repeat this process for the output layer neurons, using the output from the hidden layer neurons as inputs.

Output For o1:

$$\text{net o1} = w_5 * \text{out h1} + w_6 * \text{out h2} + b_2 * 1$$

$$0.4 * 0.593269992 + 0.45 * 0.596884378 + 0.6 * 1 = 1.105905967$$

$$\text{Out o1} = 1 / 1 + e^{-\text{net o1}}$$

$$1 / 1 + e^{-1.105905967} = 0.75136507$$

Output For o2:

$$\text{Out o2} = 0.772928465$$

Now, let's see what is the value of the error:

Error For o1:

$$E_{o1} = \sum 1/2 (\text{target} - \text{output})^2$$

$$\frac{1}{2} (0.01 - 0.75136507)^2 = 0.274811083$$

Error For o2:

$$E_{o2} = 0.023560026$$

Total Error:

$$E_{\text{total}} = E_{o1} + E_{o2}$$

$$0.274811083 + 0.023560026 = 0.298371109$$

Step – 2: Backward Propagation

Now, we will propagate backwards. This way we will try to reduce the error by changing the values of weights and biases. Consider W5, we will calculate the rate of change of error w.r.t change in weight W5.



Since we are propagating backwards, first thing we need to do is, calculate the change in total errors w.r.t the output O1 and O2.

$$E_{total} = 1/2(\text{target } o1 - \text{out } o1)^2 + 1/2(\text{target } o2 - \text{out } o2)^2$$

$$\frac{\delta E_{total}}{\delta out\ o1} = -(\text{target } o1 - \text{out } o1) = -(0.01 - 0.75136507) = 0.74136507$$

Now, we will propagate further backwards and calculate the change in output O1 w.r.t to its total net input.

$$\text{out } o1 = 1/1 + e^{-net\ o1}$$

$$\frac{\delta out\ o1}{\delta net\ o1} = \text{out } o1 (1 - \text{out } o1) = 0.75136507 (1 - 0.75136507) = 0.186815602$$

Let's see now how much does the total net input of O1 changes w.r.t W5?

$$\text{net } o1 = w5 * \text{out } h1 + w6 * \text{out } h2 + b2 * 1$$

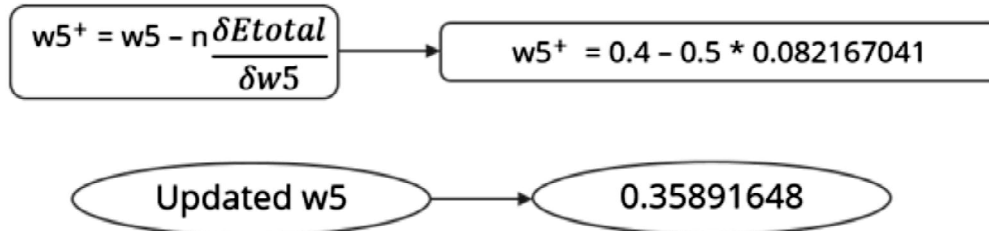
$$\frac{\delta net\ o1}{\delta w5} = 1 * \text{out } h1 = 0.593269992$$

Step – 3: Putting all the values together and calculating the updated weight value

Now, let's put all the values together:

$$\frac{\delta E_{total}}{\delta w5} = \frac{\delta E_{total}}{\delta out\ o1} * \frac{\delta out\ o1}{\delta net\ o1} * \frac{\delta net\ o1}{\delta w5} = 0.082167041$$

Let's calculate the updated value of W5



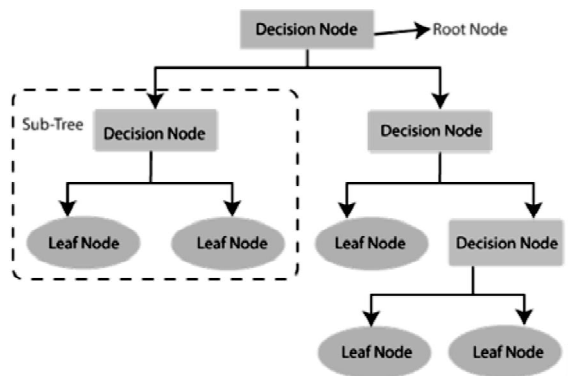
- Similarly, we can calculate the other weight values as well.
- After that we will again propagate forward and calculate the output. Again, we will calculate the error.
- If the error is minimum we will stop right there, else we will again propagate backwards and update the weight values.
- This process will keep on repeating until error becomes minimum.

3.1.8 Decision Trees

Q12. Write about decision trees

Ans :

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
- Below diagram explains the general structure of a decision tree:



Use of Decision Trees

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- The logic behind the decision tree can be easily understood because it shows a tree-like structure.

Decision Tree Terminologies

- **Root Node**
Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- **Leaf Node**
Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- **Splitting**
Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- **Branch/Sub Tree**
A tree formed by splitting the tree.
- **Pruning**
Pruning is the process of removing the unwanted branches from the tree.

➤ Parent/Child node

The root node of the tree is called the parent node, and other nodes are called the child nodes.

Pruning: Getting an Optimal Decision tree

Pruning is a process of deleting the unnecessary nodes from a tree in order to get the optimal decision tree.

A too-large tree increases the risk of overfitting, and a small tree may not capture all the important features of the dataset. Therefore, a technique that decreases the size of the learning tree without reducing accuracy is known as Pruning. There are mainly two types of tree pruning technology used:

- Cost Complexity Pruning
- Reduced Error Pruning.

Advantages of the Decision Tree

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

Disadvantages of the Decision Tree

- The decision tree contains lots of layers, which makes it complex.
- It may have an overfitting issue, which can be resolved using the Random Forest algorithm.
- For more class labels, the computational complexity of the decision tree may increase.

Q13. How does the Decision Tree algorithm Work?

Ans : (Imp.)

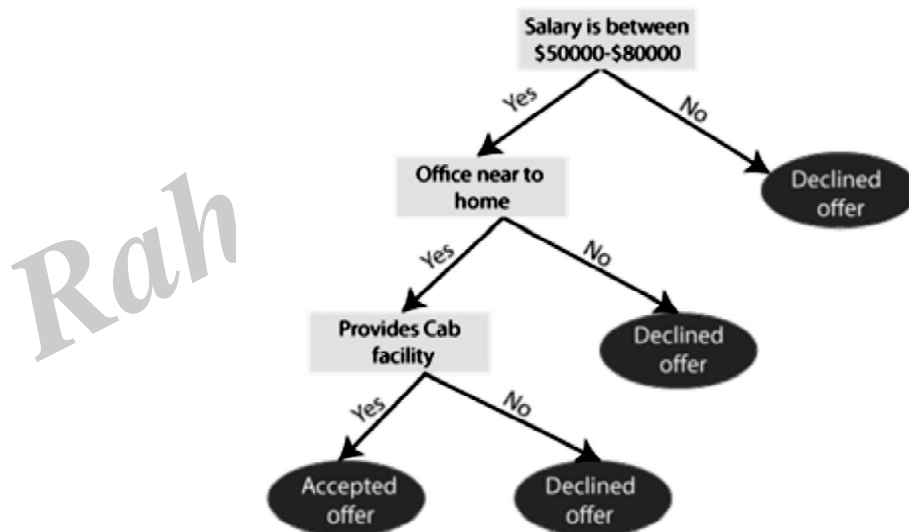
In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the

values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

- **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
- **Step-2:** Find the best attribute in the dataset using Attribute Selection Measure (ASM).
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
- **Step-4:** Generate the decision tree node, which contains the best attribute.
- **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Example : Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node. Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer). Consider the below diagram:



Attribute Selection Measures

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as Attribute selection measure or ASM. By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

1. Information Gain
2. Gini Index

1. Information Gain

- Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
- It calculates how much information a feature provides us about a class.
- According to the value of information gain, we split the node and build the decision tree.
- A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first.

Definition: Suppose S is a set of instances, A is an attribute, S_v is the subset of S with $A = v$, and $\text{Values}(A)$ is the set of all possible values of A , then

$$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{v \in \text{Value}(A)} \frac{|S_v|}{|S|} \cdot \text{Entropy}(S_v)$$

Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data.

Definition: Suppose S is a set of instances, A is an attribute, S_v is the subset of S with $A = v$, and $\text{Values}(A)$ is the set of all possible values of A , then

$$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{v \in \text{Value}(A)} \frac{|S_v|}{|S|} \cdot \text{Entropy}(S_v)$$

Example

For the set $X = \{a, a, a, b, b, b, b, b\}$

Total instances : 8

Instances of b : 5

Instances of a : 3

$$\begin{aligned} \text{Entropy } H(X) &= - \left[\left(\frac{3}{8} \right) \log_2 \frac{3}{8} + \left(\frac{5}{8} \right) \log_2 \frac{5}{8} \right] \\ &= - [0.375 * (-1.415) + 0.625 * (-0.678)] \\ &= - (-0.53 - 0.424) \\ &= 0.954 \end{aligned}$$

Building Decision Tree using Information Gain

The essentials

- Start with all training instances associated with the root node
- Use info gain to choose which attribute to label each node with
- Note: No root-to-leaf path should contain the same discrete attribute twice
- Recursively construct each subtree on the subset of training instances that would be classified down that path in the tree.

The border cases

- If all positive or all negative training instances remain, label that node “yes” or “no” accordingly
- If no attributes remain, label with a majority vote of training instances left at that node
- If no instances remain, label with a majority vote of the parent’s training instances

Example : Now, lets draw a Decision Tree for the following data using Information gain.

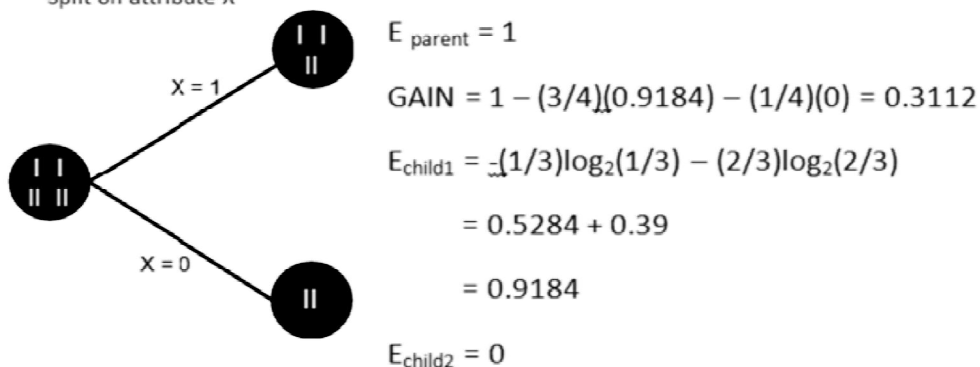
Training set: 3 features and 2 classes

X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Here, we have 3 features and 2 output classes.

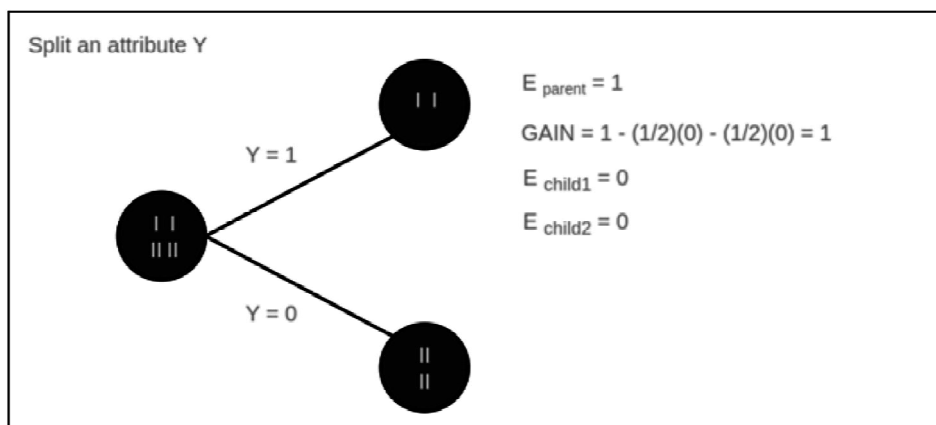
To build a decision tree using Information gain. We will take each of the feature and calculate the information for each feature.

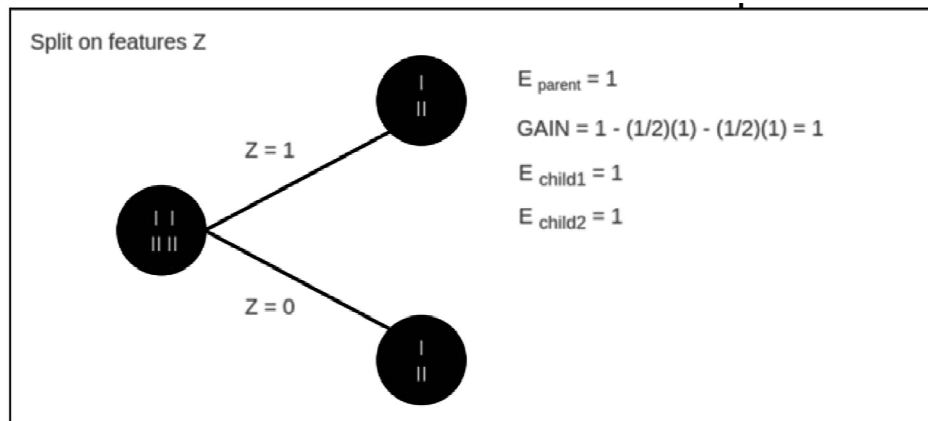
Split on attribute X



Split on feature X

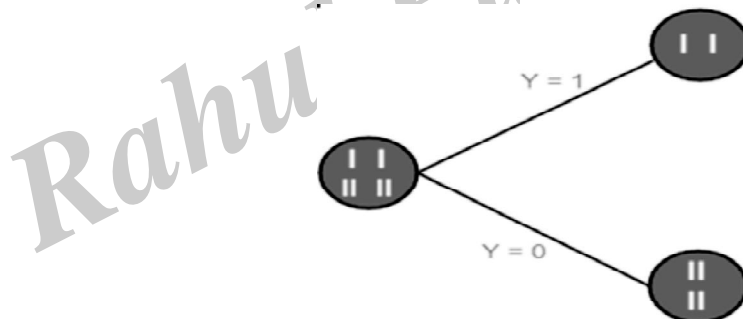
Split an attribute Y



Split on feature Y**Split on feature Z**

From the above images we can see that the information gain is maximum when we make a split on feature Y. So, for the root node best suited feature is feature Y. Now we can see that while splitting the dataset by feature Y, the child contains pure subset of the target variable. So we don't need to further split the dataset.

The final tree for the above dataset would be look like this:

**2. Gini Index**

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.
- An attribute with the low Gini index should be preferred as compared to the high Gini index.
- It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
- Gini index can be calculated using the below formula:

$$\text{Gini Index} = 1 - \sum_j P_j^2$$

Example:

Lets consider the dataset in the image below and draw a decision tree using gini index.

Index	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	1.2	positive
3	5	3.4	1.6	0.2	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.7	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative

In the dataset above there are 5 attributes from which attribute E is the predicting feature which contains 2(Positive & Negative) classes. We have an equal proportion for both the classes.

In Gini Index, we have to choose some random values to categorize each attribute. These values for this dataset are:

A	B	C	D
≥ 5	≥ 3.0	≥ 4.2	≥ 1.4
< 5	< 3.0	< 4.2	< 1.4

Calculating Gini Index for Var A

Value ≥ 5 : 12

Attribute A ≥ 5 & class = positive : $\frac{5}{12}$

Attribute A ≥ 5 & class = negative : $\frac{7}{12}$

$$\text{Gini}(5, 7) = 1 - \left[\left(\frac{5}{12} \right)^2 + \left(\frac{7}{12} \right)^2 \right] = 0.4860$$

Value < 5 : 4

Attribute A < 5 & class = positive : $\frac{3}{4}$

Attribute A < 5 & class = negative : $\frac{1}{4}$

$$\text{Gini}(3, 1) = 1 - \left[\left(\frac{3}{4} \right)^2 + \left(\frac{1}{4} \right)^2 \right] = 0.4860$$

By adding weight and sum each of the gini indices:

$$\text{gini}(\text{Target}, A) = \left(\frac{12}{16} \right) * (0.486) + \left(\frac{4}{16} \right) * (0.375) = 0.45825$$

Calculating Gini Index for Var B

Value >= 3 : 12

Attribute B >= 3 & class = positive : $\frac{8}{12}$

Attribute B >= 5 & class = negative : $\frac{4}{12}$

$$\text{Gini}(5, 7) = 1 - \left[\left(\frac{8}{12} \right)^2 + \left(\frac{4}{12} \right)^2 \right] = 0.4460$$

Value < 3 : 4

Attribute A < 3 & class = positive : $\frac{0}{4}$

Attribute A < 3 & class = negative : $\frac{4}{4}$

$$\text{Gini}(3, 1) = 1 - \left[\left(\frac{0}{4} \right)^2 + \left(\frac{4}{4} \right)^2 \right] = 1$$

By adding weight and sum each of the gini indices:

$$\text{gini}(\text{Target}, B) = \left(\frac{12}{16} \right) * (0.446) + \left(\frac{0}{16} \right) * (0) = 0.3345$$

Using the same approach we can calculate the Gini index for C and D attributes.

	Positive	Negative
For A ≥ 5.0	5	7
< 5	3	1

Gini Index of A = 0.45825

	Positive	Negative
For B ≥ 3.0	8	4
< 3.0	0	4

Gini Index of B = 0.3345

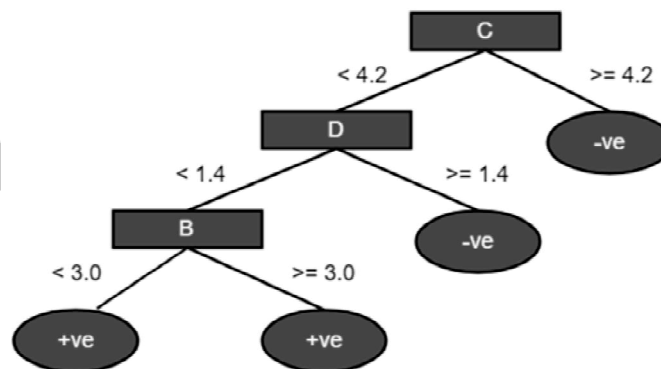
	Positive	Negative
For C ≥ 4.2	0	6
< 4.2	8	2

Gini Index of C = 0.2

	Positive	Negative
For D ≥ 1.4	0	5
< 1.4	8	3

Gini Index of D = 0.273

Decision tree for above dataset



The most notable types of decision tree algorithms are:-

1. Iterative Dichotomiser 3 (ID3)

This algorithm uses Information Gain to decide which attribute is to be used classify the current subset of the data. For each level of the tree, information gain is calculated for the remaining data recursively.

2. C4.5

This algorithm is the successor of the ID3 algorithm. This algorithm uses either Information gain or Gain ratio to decide upon the classifying attribute. It is a direct improvement from the ID3 algorithm as it can handle both continuous and missing attribute values.

3. Classification and Regression Tree (CART)

It is a dynamic learning algorithm which can produce a regression tree as well as a classification tree depending upon the dependent variable.

Q14. Explain ID3 algorithm with example.

Ans :

ID3 algorithm

ID3 Algorithm will perform following tasks recursively:

1. Create root node for the tree
2. If all examples are positive, return leaf node "positive"
3. Else if all examples are negative, return leaf node "negative"
4. Calculate the entropy of current state $H(S)$
5. For each attribute, calculate the entropy with respect to the attribute „x“ denoted by $H(S, x)$
6. Select the attribute which has maximum value of $IG(S, x)$
7. Remove the attribute that offers highest IG from the set of attributes
8. Repeat until we run out of all attributes, or the decision tree has all leaf nodes.

We'll build a decision tree to do that using **ID3 algorithm**.

Day	Outlook	Temperature	Humidity	Wind	Play Golf
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Now we'll go ahead and grow the decision tree. The initial step is to calculate $H(S)$, the Entropy of the current state.

In the above example, we can see in total there are 5 No's and 9 Yes's

Yes	No	Total
9	5	14.

$$\text{Entropy}(S) = \sum_{x \in X} p(x) \log_2 \frac{1}{p(x)}$$

$$\begin{aligned} \text{Entropy}(S) &= -\left(\frac{9}{14}\right) \log_2 \left(\frac{9}{14}\right) - \left(\frac{5}{14}\right) \log_2 \left(\frac{5}{14}\right) \\ &= 0.940 \end{aligned}$$

where „x“ are the possible values for an attribute. Here, attribute „Wind “ takes two possible values in the sample data,

hence $x = \{\text{Weak, Strong}\}$ we ll have to calculate:

- 1) $H(S_{\text{weak}})$
- 2) $H(S_{\text{Strong}})$
- 3) $P(S_{\text{weak}})$
- 4) $P(S_{\text{strong}})$
5. $H(S) = 0.94$ which we had already calculated in the previous example.

Amongst all the 14 examples we have 8 places where the wind is weak and 6 where the wind is Strong.

Wind = Weak	Wind = Strong	Total
8	6	14

$$P(S_{\text{weak}}) = \frac{\text{Number of weak}}{\text{Total}}$$

$$= \frac{8}{14}$$

$$P(S_{\text{strong}}) = \frac{\text{Number of Strong}}{\text{Total}}$$

$$= \frac{6}{14}$$

Now out of the 8 Weak examples, 6 of them were „Yes“ for Play Golf and 2 of them were „No“ for „Play Golf“. So, we have,

$$\begin{aligned} \text{Entropy}(S_{\text{weak}}) &= -\left(\frac{6}{8}\right) \log_2 \left(\frac{6}{8}\right) - \left(\frac{2}{8}\right) \log_2 \left(\frac{2}{8}\right) \\ &= 0.811 \end{aligned}$$

Similarly, out of 6 Strong examples, we have 3 examples where the outcome was „Yes“ for Play Golf and 3 where we had „No“ for Play Golf.

$$\begin{aligned} \text{Entropy}(S_{\text{strong}}) &= -\left(\frac{3}{6}\right) \log_2 \left(\frac{3}{6}\right) - \left(\frac{3}{6}\right) \log_2 \left(\frac{3}{6}\right) \\ &= 1.000 \end{aligned}$$

Remember, here half items belong to one class while other half belong to other. Hence we have perfect randomness. Now we have all the pieces required to calculate the Information Gain,

$$\text{IG}(S, \text{Wind}) = H(S) - \sum_{i=0}^n P(x) * H(x)$$

$$\begin{aligned} \text{IG}(S, \text{Wind}) &= H(S) - P(S_{\text{weak}}) * H(S_{\text{weak}}) \\ &\quad - P(S_{\text{strong}}) * H(S_{\text{strong}}) \end{aligned}$$

$$\begin{aligned} &= 0.940 - \left(\frac{8}{14}\right)(0.811) - \left(\frac{6}{14}\right)(1.00) \\ &= 0.048 \end{aligned}$$

Which tells us the Information Gain by considering „Wind“ as the feature and give us information gain of 0.048.

Now we must similarly calculate the Information Gain for all the features

$$\text{IG}(S, \text{Outlook}) = 0.246$$

$$\text{IG}(S, \text{Temperature}) = 0.029$$

$$\text{IG}(S, \text{Humidity}) = 0.151$$

$$\text{IG}(S, \text{Wind}) = 0.048 \text{ (Previous example)}$$

We can clearly see that $\text{IG}(S, \text{Outlook})$ has the highest information gain of 0.246, hence we chose Outlook attribute as the root node. At this point, the decision tree looks like.



Here we observe that whenever the outlook is Overcast, Play Golf is always 'Yes', it's no coincidence by any chance, the simple tree resulted because of the highest information gain is given by the attribute Outlook. Now how do we proceed from this point? We can simply apply recursion, you might want to look at the algorithm steps described earlier. Now that we've used Outlook, we've got three of them remaining Humidity, Temperature, and Wind. And, we had three possible values of Outlook: Sunny, Overcast, Rain. Where the Overcast node already ended up having leaf node 'Yes', so we're left with two subtrees to compute: Sunny and Rain.

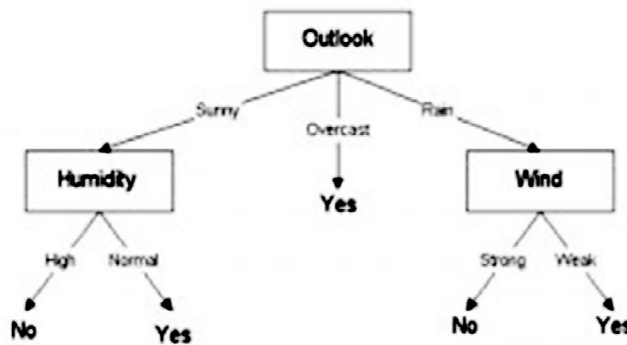
Next step would be computing $H(S_{\text{sunny}})$.

Table where the value of Outlook is Sunny looks like:

Temperature	Humidity	Wind	Play Golf
Hot	High	Weak	No
Hot	High	Strong	No
Mild	High	Weak	No
Cool	Normal	Weak	Yes
Mild	Normal	Strong	Yes

$$H(S_{\text{sunny}}) = \left(\frac{3}{5}\right) \log_2\left(\frac{3}{5}\right) - \left(\frac{2}{5}\right) \log_2\left(\frac{2}{5}\right) = 0.96$$

As we can see the highest Information Gain is given by Humidity. Proceeding in the same way with S_{rain} will give us Wind as the one with highest information gain. The final Decision Tree looks something like this. The final Decision Tree looks something like this.



Q15. write about C4.5 algorithm.

Ans :

C4.5 algorithm

C4.5 builds decision trees from a set of training data in the same way as ID3, using the concept of information entropy. The training data is a set $S = s_1, s_2, \dots$ of already classified samples. Each sample s_i consists of a p -dimensional vector $(x_{1,i}, s_{2,i}, \dots, x_{p,i})$ where the x_i represent attribute values of features of the sample, as well as the class in which s_i falls.

At each node of the tree, C4.5 chooses the attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. The splitting criterion is the normalized information

gain (difference in entropy). The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurses on the partitioned sublists.

This algorithm has a few base cases.

- All the samples in the list belong to the same class. When this happens, it simply creates a leaf node for the decision tree saying to choose that class.
- None of the features provide any information gain. In this case, C4.5 creates a decision node higher up the tree using the expected value of the class.
- Instance of previously-unseen class encountered. Again, C4.5 creates a decision node higher up the tree using the expected value.

Pseudocode

In pseudocode, the general algorithm for building decision trees is:

1. Check for the above base cases.
2. For each attribute a , find the normalized information gain ratio from splitting on a .
3. Let a_{best} be the attribute with the highest normalized information gain.
4. Create a decision node that splits on a_{best} .
5. Recurse on the sublists obtained by splitting on a_{best} , and add those nodes as children of node.

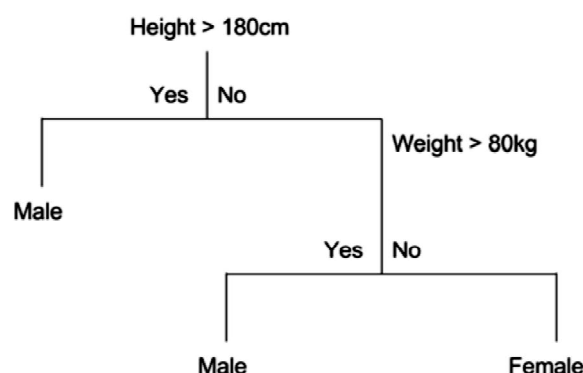
Q16. Give the brief introduction about classification and regression trees

Ans :

Classification Trees

A classification tree is an algorithm where the target variable is fixed or categorical. The algorithm is then used to identify the "class" within which a target variable would most likely fall.

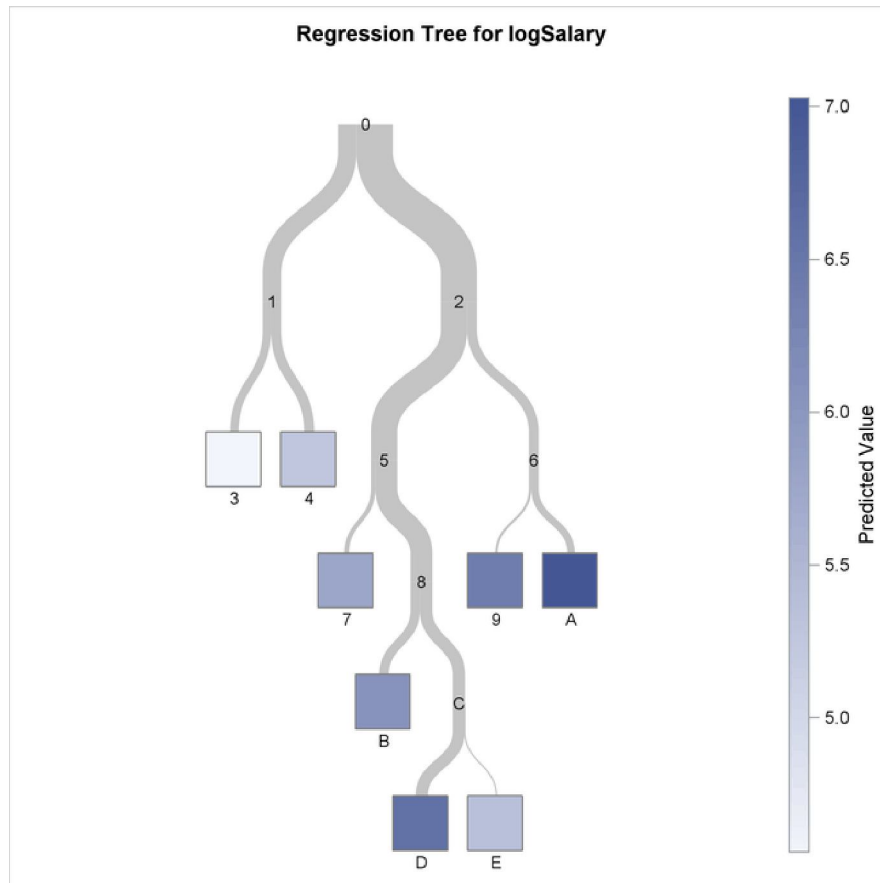
An example of a classification-type problem would be determining who will or will not subscribe to a digital platform; or who will or will not graduate from high school.



Regression Trees

A regression tree refers to an algorithm where the target variable is and the algorithm is used to predict its value. As an example of a regression type problem, you may want to predict the selling prices of a residential house, which is a continuous dependent variable.

This will depend on both continuous factors like square footage as well as categorical factors like the style of home, area in which the property is located and so on.



When to use Classification and Regression Trees

Classification trees are used when the dataset needs to be split into classes which belong to the response variable. In many cases, the classes are Yes or No.

In other words, they are just two and mutually exclusive. In some cases, there may be more than two classes in which case a variant of the classification tree algorithm is used.

Regression trees, on the other hand, are used when the response variable is continuous. For instance, if the response variable is something like the price of a property or the temperature of the day, a regression tree is used.

In other words, regression trees are used for prediction-type problems while classification trees are used for classification-type problems.

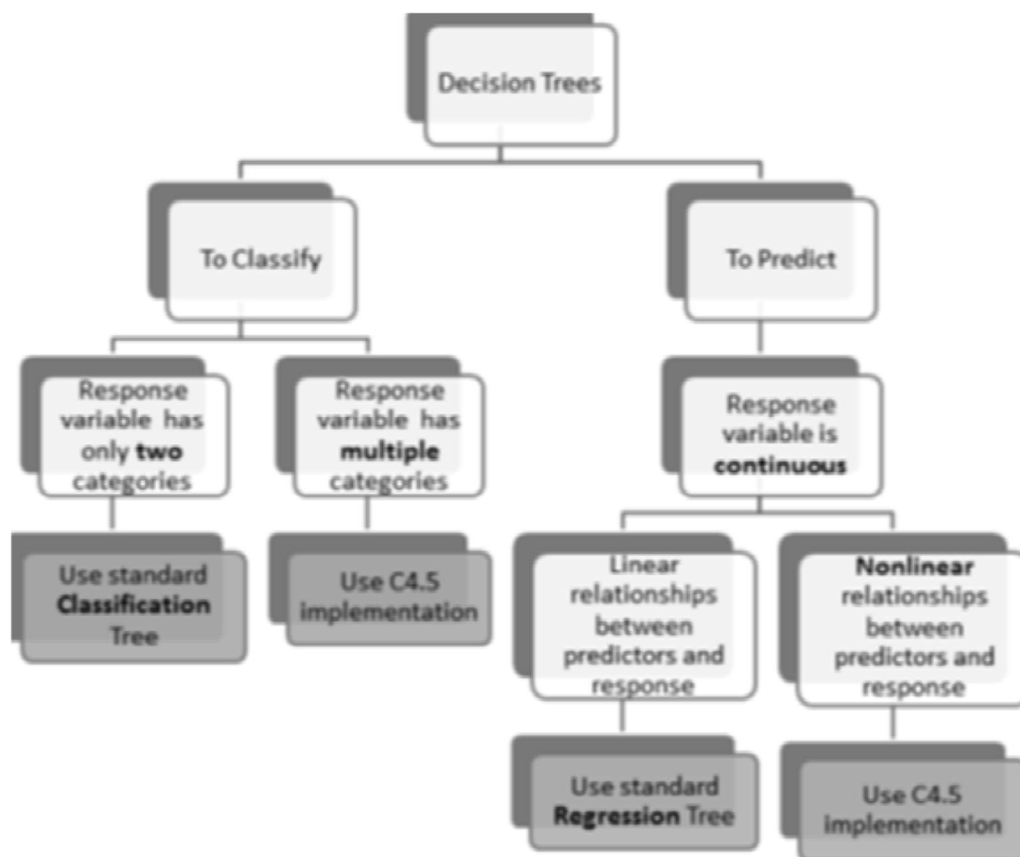
How Classification and Regression Trees Work

A classification tree splits the dataset based on the homogeneity of data. Say, for instance, there are two variables; income and age; which determine whether or not a consumer will buy a particular kind of phone.

If the training data shows that 95% of people who are older than 30 bought the phone, the data gets split there and age becomes a top node in the tree. This split makes the data "95% pure". Measures

of impurity like entropy or Gini index are used to quantify the homogeneity of the data when it comes to classification trees. In a regression tree, a regression model is fit to the target variable using each of the independent variables. After this, the data is split at several points for each independent variable.

At each such point, the error between the predicted values and actual values is squared to get "A Sum of Squared Errors" (SSE). The SSE is compared across the variables and the variable or point which has the lowest SSE is chosen as the split point. This process is continued recursively.



Advantages of Classification and Regression Trees

The purpose of the analysis conducted by any classification or regression tree is to create a set of if-else conditions that allow for the accurate prediction or classification of a case.

(i) The Results are Simplistic

The interpretation of results summarized in classification or regression trees is usually fairly simple. The simplicity of results helps in the following ways.

- It allows for the rapid classification of new observations. That's because it is much simpler to evaluate just one or two logical conditions than to compute scores using complex nonlinear equations for each group.
- It can often result in a simpler model which explains why the observations are either classified or predicted in a certain way. For instance, business problems are much easier to explain with if-then statements than with complex nonlinear equations.

- (ii) Classification and Regression Trees are Nonparametric & Nonlinear

The results from classification and regression trees can be summarized in simplistic if-then conditions.

This negates the need for the following implicit assumptions.

- The predictor variables and the dependent variable are linear.
- The predictor variables and the dependent variable follow some specific nonlinear link function.
- The predictor variables and the dependent variable are monotonic.

Since there is no need for such implicit assumptions, classification and regression tree methods are well suited to data mining. This is because there is very little knowledge or assumptions that can be made beforehand about how the different variables are related.

As a result, classification and regression trees can actually reveal relationships between these variables that would not have been possible using other techniques.

- (iii) Classification and Regression Trees Implicitly Perform Feature Selection

Feature selection or variable screening is an important part of analytics. When we use decision trees, the top few nodes on which the tree is split are the most important variables within the set. As a result, feature selection gets performed automatically and we don't need to do it again.

Limitations of Classification and Regression Trees

Classification and regression tree tutorials, as well as classification and regression tree ppts, exist in abundance. This is a testament to the popularity of these decision trees and how frequently they are used. However, these decision trees are not without their disadvantages.

There are many classification and regression trees examples where the use of a decision tree has not led to the optimal result. Here are some of the limitations of classification and regression trees.

- (i) **Overfitting**

Overfitting occurs when the tree takes into account a lot of noise that exists in the data and comes up with an inaccurate result.

- (ii) **High variance**

In this case, a small variance in the data can lead to a very high variance in the prediction, thereby affecting the stability of the outcome.

- (iii) **Low bias**

A decision tree that is very complex usually has a low bias. This makes it very difficult for the model to incorporate any new data.

3.1.9. Bayes Optimal Classifier

Q17. What is the use of Bayes optimal classifier in the machine learning?

Ans : (Imp.)

- The Bayes Optimal Classifier is a probabilistic model that makes the most probable prediction for a new example.
- It is described using the Bayes Theorem that provides a principled way for calculating a conditional probability. It is also closely related to the Maximum a Posteriori: a probabilistic framework referred to as MAP that finds the most probable hypothesis for a training dataset
- The Bayes optimal classifier is a probabilistic model that makes the most probable prediction for a new example, given the training dataset.
- This model is also referred to as the Bayes optimal learner, the Bayes classifier, Bayes optimal decision boundary, or the Bayes optimal discriminant function.
- In general, the most probable classification of the new instance is obtained by combining the predictions of all hypotheses, weighted by their posterior probabilities.
- Bayesian probability basically talks about the interpretation of "partial beliefs".
- Bayesian Estimation calculates the validity of the proposition.

Validity of the Proposition depends on two things:

- (i) Prior Estimate.
- (ii) New Relevant evidence.

Hypothesis Space (H)

Hypothesis space is the set of all the possible legal hypothesis. This is the set from which the machine learning algorithm would determine the best possible (only one) which would best describe the target function or the outputs.

Hypothesis (h)

A hypothesis is a function that best describes the target in supervised machine learning. The hypothesis that an algorithm would come up depends upon the data and also depends upon the restrictions and bias that we have imposed on the data.

Bayes Theorem

$$P(h | D) = \frac{P(D | h)P(h)}{P(D)}$$

$P(h | D)$ = Posterior probability of h

$P(h)$ = Prior probability of h

$P(D | h)$ = Probability of observing D
given that h holds

$P(D)$ = Probability of observing D

MAP Hypothesis

In Bayesian statistics, a maximum a posterior probability (MAP) estimate is an estimate of an unknown quantity, that equals the mode of the posterior distribution. The MAP can be used to obtain a point estimate of an unobserved quantity on the basis of empirical data.

$h(\text{MAP}) = \operatorname{argmax} P(h/D) \{ \text{where } h \in H, \text{ also } H\text{-hypothesis Space} \}$

$h(\text{MAP}) = \operatorname{argmax} P(D/h)P(h) \{ \text{where } h \in H \}$

ML Hypothesis

Maximum likelihood estimation is a method that determines values for the parameters of a model. The parameter values are found such that they maximize the likelihood that the process described by the model produced the data that were actually observed.

$h(\text{ML}) = \operatorname{argmax} P(D/h)$ is maximum {where $h \in H$ }

If the possible classification of the new example can take on any value v_j from some set V , then the probability $P(v_j | D)$ that the correct classification for the new instance is v_j , is just

$$P(V_j | D) = \sum_{h_i \in H} P(v_j | h_i)P(h_i | D)$$

The optimal classification of the new instance is the value v_j , for which $P(v_j | D)$ is maximum.

$$\operatorname{argmax}_{v_j \in V} \sum_{h_i \in H} P(v_j | h_i)P(h_i | D)$$

Example

To develop some intuitions consider a hypothesis space containing three hypotheses, h_1 , h_2 , and h_3 . Suppose that the posterior probabilities of these hypotheses given the training data are 0.4, 0.3, and 0.3 respectively.

Thus, h_1 is the MAP hypothesis. Suppose a new instance x is encountered, which is classified positive by h_1 , but negative by h_2 and h_3 . Taking all hypotheses into account, the probability that x is positive is 0.4 (the probability associated with h_1), and the probability that it is negative is therefore 0.6.

The most probable classification (negative) in this case is different from the classification generated by the MAP hypothesis.

To illustrate in terms of the above example, the set of possible classifications of the new instance is $V = \{ \oplus \odot \}$, and

$$P(h_1 | D) = 0.4 \quad P(\odot | h_1) = 0 \quad P(\oplus | h_1) = 1$$

$$P(h_2 | D) = 0.3 \quad P(\odot | h_2) = 1 \quad P(\oplus | h_2) = 0$$

$$P(h_3 | D) = 0.3 \quad P(\odot | h_3) = 1 \quad P(\oplus | h_3) = 0$$

$$\text{therefore, } \sum_{h_i \in H} P(\oplus | h_i) P(h_i | D) = 0.4$$

$$\text{And } \sum_{h_i \in H} P(\odot | h_i) P(h_i | D) = 0.6$$

$$\arg \max_{v_j \in \{\oplus, \odot\}} \sum_{h_i \in H} P(v_j | h_i) P(h_i | D) = \odot$$

3.10 NAÏVE BAYES

Q18. Why is it called Naïve Bayes? Explain Naïve Bayes Algorithm with example

Ans :

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in text classification that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as:

- Naïve: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red,

spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.

- **Bayes:** It is called Bayes because it depends on the principle of Bayes' Theorem.

Bayes' Theorem

- Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
- The formula for Bayes' theorem is given as:

$$P(A | B) = \frac{P(B | A)P(A)}{P(B)}$$

Where,

P(A|B) is Posterior probability

Probability of hypothesis A on the observed event B.

P(B|A) is Likelihood probability

Probability of the evidence given that the probability of a hypothesis is true.

P(A) is Prior Probability

Probability of hypothesis before observing the evidence.

P(B) is Marginal Probability

Probability of Evidence.

Working of Naïve Bayes' Classifier

Working of Naïve Bayes' Classifier can be understood with the help of the below example:

Suppose we have a dataset of weather conditions and corresponding target variable "Play". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions. So to solve this problem, we need to follow the below steps:

1. Convert the given dataset into frequency tables.
2. Generate Likelihood table by finding the probabilities of given features.

3. Now, use Bayes theorem to calculate the posterior probability.

Problem

If the weather is sunny, then the Player should play or not?

Solution

To solve this, first consider the below dataset:

	Outlook	Play
0	Rainy	Yes
1	Sunny	Yes
2	Overcast	Yes
3	Overcast	Yes
4	Sunny	No
5	Rainy	Yes
6	Sunny	Yes
7	Overcast	Yes
8	Rainy	No
9	Sunny	No
10	Sunny	Yes
11	Rainy	No
12	Overcast	Yes

Frequency table for the Weather Conditions

Weather	Yes	No
Overcast	5	0
Rainy	2	2
Sunny	3	2
Total	10	5

Likelihood table weather condition

Weather	No	Yes	
Overcast	0	5	$5/14=0.35$
Rainy	2	2	$4/14=0.29$
Sunny	2	3	$5/14=0.35$
All	$4/14=0.29$	$10/14=0.71$	

Applying Bayes'theorem

$$P(\text{Yes} | \text{Sunny}) = P(\text{Sunny} | \text{Yes}) * P(\text{Yes}) / P(\text{Sunny})$$

$$P(\text{Sunny} | \text{Yes}) = 3/10 = 0.3$$

$$P(\text{Sunny}) = 0.35$$

$$P(\text{Yes}) = 0.71$$

$$\text{So } P(\text{Yes} | \text{Sunny}) = 0.3 * 0.71 / 0.35 = 0.60$$

$$P(\text{No} | \text{Sunny}) = P(\text{Sunny} | \text{No}) * P(\text{No}) / P(\text{Sunny})$$

$$P(\text{Sunny} | \text{No}) = 2/4 = 0.5$$

$$P(\text{No}) = 0.29$$

$$P(\text{Sunny}) = 0.35$$

$$\text{So } P(\text{No} | \text{Sunny}) = 0.5 * 0.29 / 0.35 = 0.41$$

So as we can see from the above calculation that $P(\text{Yes} | \text{Sunny}) > P(\text{No} | \text{Sunny})$

Hence on a Sunny day, Player can play the game.

Advantages of Naïve Bayes Classifier:

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.

- It can be used for Binary as well as Multi-class Classifications.
- It performs well in Multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for text classification problems.

Disadvantages of Naïve Bayes Classifier

- Naïve Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

Applications of Naïve Bayes Classifier

- It is used for Credit Scoring.
- It is used in medical data classification.
- It can be used in real-time predictions because Naïve Bayes Classifier is an eager learner.
- It is used in Text classification such as Spam filtering and Sentiment analysis.

Types of Naïve Bayes Model

There are three types of Naive Bayes Model, which are given below:

- **Gaussian**

The Gaussian model assumes that features follow a normal distribution. This means if predictors take continuous values instead of discrete, then the model assumes that these values are sampled from the Gaussian distribution.

- **Multinomial**

The Multinomial Naïve Bayes classifier is used when the data is multinomial distributed. It is primarily used for document classification problems, it means a particular document belongs to which category such as Sports, Politics, education, etc.

The classifier uses the frequency of words for the predictors.

- **Bernoulli**

The Bernoulli classifier works similar to the Multinomial classifier, but the predictor variables are the independent Booleans variables. Such as if a particular word is present or not in a document. This model is also famous for document classification tasks.

UNIT IV

Evaluation measures: Hypothesis testing, Ensemble Methods, Bagging, Adaboost Gradient Boosting, Clustering, K-means, K-medoids, Density-based Hierarchical, Spectral

4.1 EVALUATION MEASURES

4.1.1 Hypothesis Testing

Q1. What is hypothesis testing? Explain about it with help of example.

Ans : (Imp.)

Hypothesis are statement about the given problem. Hypothesis testing is a statistical method that is used in making a statistical decision using experimental data. Hypothesis testing is basically an assumption that we make about a population parameter. It evaluates two mutually exclusive statements about a population to determine which statement is best supported by the sample data.

Example:

You say an average student in the class is 30 or a boy is taller than girls. All those are an example in which we assume or need some statistic way to prove those. We need some mathematical conclusion whatever we are assuming is true.

Need for Hypothesis Testing

Hypothesis testing is an important procedure in statistics. Hypothesis testing evaluates two mutually exclusive population statements to determine which statement is most supported by sample data. When we say that the findings are statistically significant, it is thanks to hypothesis testing.

Parameters of hypothesis testing

- **Null hypothesis(H₀):** In statistics, the null hypothesis is a general given statement or default position that there is no relationship between two measured cases or no relationship among groups.

In other words, it is a basic assumption or made based on the problem knowledge.

Example: A company production is = 50 unit/per day etc.

- **Alternative hypothesis(H₁):** The alternative hypothesis is the hypothesis used in hypothesis testing that is contrary to the null hypothesis.

Example : A company production is not equal to 50 unit/per day etc.

- **Level of significance:** It refers to the degree of significance in which we accept or reject the null-hypothesis. 100% accuracy is not possible for accepting a hypothesis, so we, therefore, select a level of significance that is usually 5%. This is normally denoted with α and generally, it is 0.05 or 5%, which means your output should be 95% confident to give similar kind of result in each sample.

- **P-value:** The P value, or calculated probability, is the probability of finding the observed/extreme results when the null hypothesis(H₀) of a study given problem is true. If your P-value is less than the chosen significance level then you reject the null hypothesis i.e. accept that your sample claims to support the alternative hypothesis.

Steps to perform Hypothesis Testing:

1. Define null and alternative hypothesis
2. Examine data, check assumptions
3. Calculate Test Statistic
4. Determine the Corresponding p-value
5. Make a decision about the null hypothesis.

To perform all these steps, let us take an example to understand easily.

Problem

Considering Italian adults from the age group 18-30 living in Italy, Do males have significantly higher mean Body Mass Index (BMI) than females?

Here the population is Italian adults (18-30) in Italy and the parameter of interest is Body Mass Index (BMI)

Step-1: Define Hypotheses

- **Null:** There is no difference in mean BMI
 $H(0): U_1 = U_2$ [U_1 represents the population mean BMI for Males and U_2 represents the population mean BMI for females] Here $H(0)$ says that they are equal to each other
- **Alternative:** There is a significant difference in mean BMI
 $H(A): U_1 \neq U_2$ [U_1 represents the population mean BMI for Males and U_2 represents the population mean BMI for females] Here $H(A)$ says that they are not equal to each other
- Significance level = 5%

Step-2: Examine data and checking assumptions

In this step, the data was filtered to include only Italian adults that were between the ages of 18 and 30. After this, we need to do some statistics calculations like mean, minimum, maximum, standard deviation, and sample sizes for both males and females.

Some of the assumptions that we need to check are as follows:

- Samples are considered simple random samples
- Samples are independent of one another
- Both populations of responses are approximately normal or sample sizes are both large enough.

Step-3: Calculate Test Statistic

Test Statistic is a measure of how far our sample statistic is from our hypothesized population parameter, in terms of estimated standard errors.

- $Z = \frac{\text{Best Estimate} - \text{null value}}{\text{Estimated standard error}}$
- The Best Estimate is the difference between the male and female statistic sample mean
- The null value is the hypothesized null value
- The estimated standard error for two means can change depending on which approach we are going to use.
- The two approaches that you can use are the pooled approach and the unpooled approach.
- The pooled approach is the variance of two populations are assumed to be equal.
- The unpooled approach is the assumption of equal variances is dropped.

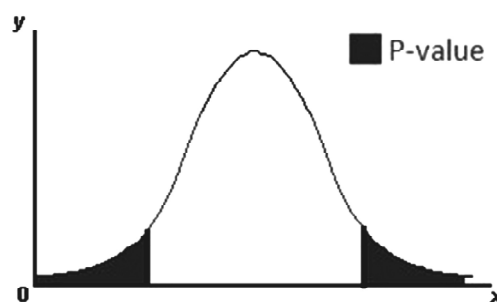
Step-4: Determining P-value

P-value is determined by assuming the null hypothesis is true, it is the probability of observing a test statistic of a value (Z) or more extreme.

So we are going to calculate this probability using Z-distribution where $df = n_1 + n_2 - 2$

We need to check both sides since it is a two-sided alternative hypothesis because our alternative is not equal too. so, we have to check both the upper and lower tail of our distribution.

The distribution graph looks like given below with its corresponding sample size and the degrees of freedom:



Distribution curve

From the above graph, we can see both our positive test statistic value and below the negative test statistic value. This means that if the difference in population mean BMI between males and females was really zero, so if that null hypothesis was true,

then observing a difference in sample means of the test statistic value or something more extreme is fairly likely. There is almost a 20 percent chance of seeing that because this value is so large, we are going to go ahead and fail to reject the null.

Step-5: Make a Decision

If P-value is larger than the significance level, which means there is weak evidence against the null. Thus we fail to reject the null hypothesis.

So, in summary, hypothesis tests are used to put theories about a parameter of interest to the test. Here, that parameter is the difference in population means. The basic steps for performing this hypothesis test. First, we're going to define our hypotheses. Then, we're going to examine our data while checking our assumptions and calculating our test statistic. With this test statistic, we'll determine our corresponding p-value, and then finally, we will make a decision based on this value.

The assumptions for the two-sample t-test for population means are that we need both sets of data to be two simple random samples and they need to be independent of one another. We need to make sure that both populations of responses are normally distributed. If not, we need to make sure we at least have a large sample size so we can apply the central limit theorem. Whether or not our population variances are equal is also crucial in determining whether we use a pooled or unpooled approach. Finally, we need to know how to interpret the p-value, the decision, and our final conclusion. These are all very important when conducting a hypothesis test.

Example :

Given a coin and it is not known whether that is fair or tricky so let's decide null and alternate hypothesis

- Null Hypothesis(H_0): a coin is a fair coin.
- Alternative Hypothesis(H_1): a coin is a tricky coin.
- Now let's toss the coin and calculate p-value (probability value).
- Toss a coin 1st time and assume that result is head- P-value = (as head and tail have equal probability)

- Toss a coin 2nd time and assume that result again is head, now p-value and similarly, we Toss 6 consecutive time and got the result as all heads, now P-value

But we set our significance level as error rate we allow and here we see we are beyond that level i.e. our null-hypothesis does not hold good so we need to reject and propose that this coin is a tricky coin which is actually because it gives us 6 consecutive heads.

4.1.2 Ensemble Methods

Q2. Write about different kinds of ensemble methods used in machine learning for predicting the data.

Ans : (Imp.)

The goal of ensemble methods is to combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability / robustness over a single estimator.

Two families of ensemble methods are usually distinguished:

- In averaging methods, the driving principle is to build several estimators independently and then to average their predictions. On average, the combined estimator is usually better than any of the single base estimator because its variance is reduced.

Examples: Bagging methods, Forests of randomized trees, ...

- By contrast, in boosting methods, base estimators are built sequentially and one tries to reduce the bias of the combined estimator. The motivation is to combine several weak models to produce a powerful ensemble.

Examples: AdaBoost, Gradient Tree Boosting, ...

Bagging meta-estimator

In ensemble algorithms, bagging methods form a class of algorithms which build several instances of a black-box estimator on random subsets of the original training set and then aggregate their individual predictions to form a final prediction. These methods are used as a way to reduce the variance of a base estimator (e.g., a decision tree),

by introducing randomization into its construction procedure and then making an ensemble out of it. In many cases, bagging methods constitute a very simple way to improve with respect to a single model, without making it necessary to adapt the underlying base algorithm. As they provide a way to reduce overfitting, bagging methods work best with strong and complex models (e.g., fully developed decision trees), in contrast with boosting methods which usually work best with weak models (e.g., shallow decision trees).

Bagging methods come in many flavours but mostly differ from each other by the way they draw random subsets of the training set:

- When random subsets of the dataset are drawn as random subsets of the samples, then this algorithm is known as Pasting
- When samples are drawn with replacement, then the method is known as Bagging
- When random subsets of the dataset are drawn as random subsets of the features, then the method is known as Random Subspaces
- Finally, when base estimators are built on subsets of both samples and features, then the method is known as Random Patches

Forests of randomized trees

The `sklearn.ensemble` module includes two averaging algorithms based on randomized decision trees: the `RandomForest` algorithm and the `Extra-Trees` method. Both algorithms are perturb-and-combine techniques specifically designed for trees. This means a diverse set of classifiers is created by introducing randomness in the classifier construction. The prediction of the ensemble is given as the averaged prediction of the individual classifiers.

1. Random Forests

In random forests (see `Random Forest Classifier` and `Random Forest Regressor` classes), each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set.

Furthermore, when splitting each node during the construction of a tree, the best split is found either from all input features or a random subset of size `max_features`. (See the parameter tuning guidelines for more details).

The purpose of these two sources of randomness is to decrease the variance of the forest estimator. Indeed, individual decision trees typically exhibit high variance and tend to overfit. The injected randomness in forests yield decision trees with somewhat decoupled prediction errors. By taking an average of those predictions, some errors can cancel out. Random forests achieve a reduced variance by combining diverse trees, sometimes at the cost of a slight increase in bias. In practice the variance reduction is often significant hence yielding an overall better model.

2. Extremely Randomized Trees

In extremely randomized trees (see `Extra Trees Classifier` and `Extra Trees Regressor` classes), randomness goes one step further in the way splits are computed. As in random forests, a random subset of candidate features is used, but instead of looking for the most discriminative thresholds, thresholds are drawn at random for each candidate feature and the best of these randomly-generated thresholds is picked as the splitting rule

3. Parameters

The main parameters to adjust when using these methods is `n_estimators` and `max_features`. The former is the number of trees in the forest. The larger the better, but also the longer it will take to compute. In addition, note that results will stop getting significantly better beyond a critical number of trees. The latter is the size of the random subsets of features to consider when splitting a node. The lower the greater the reduction of variance, but also the greater the increase in bias

4. Parallelization

Finally, this module also features the parallel construction of the trees and the parallel computation of the predictions through the `n_jobs` parameter. If `n_jobs=k` then computations are partitioned into `k` jobs, and run on `k` cores of the machine. If `n_jobs=-1` then all cores available on the machine are used. Note that because of inter-process communication overhead, the speedup might not be linear (i.e., using `k` jobs will unfortunately not be `k` times as fast). Significant speedup can still be achieved though when building a large number of trees, or when building a single tree requires a fair amount of time (e.g., on large datasets).

5. Feature importance evaluation

The relative rank (i.e. depth) of a feature used as a decision node in a tree can be used to assess the relative importance of that feature with respect to the predictability of the target variable. Features used at the top of the tree contribute to the final prediction decision of a larger fraction of the input samples. The expected fraction of the samples they contribute to can thus be used as an estimate of the relative importance of the features. In scikit-learn, the fraction of samples a feature contributes to is combined with the decrease in impurity from splitting them to create a normalized estimate of the predictive power of that feature.

By averaging the estimates of predictive ability over several randomized trees one can reduce the variance of such an estimate and use it for feature selection. This is known as the mean decrease in impurity, or MDI

6. Totally Random Trees Embedding

`RandomTreesEmbedding` implements an unsupervised transformation of the data. Using a forest of completely random

trees, `RandomTreesEmbedding` encodes the data by the indices of the leaves a data point ends up in. This index is then encoded in a one-of-K manner, leading to a high dimensional, sparse binary coding. This coding can be computed very efficiently and can then be used as a basis for other learning tasks. The size and sparsity of the code can be influenced by choosing the number of trees and the maximum depth per tree. For each tree in the ensemble, the coding contains one entry of one. The size of the coding is at most $n_estimators * 2^{**} max_depth$, the maximum number of leaves in the forest.

As neighboring data points are more likely to lie within the same leaf of a tree, the transformation performs an implicit, non-parametric density estimation.

7. AdaBoost

The core principle of AdaBoost is to fit a sequence of weak learners (i.e., models that are only slightly better than random guessing, such as small decision trees) on repeatedly modified versions of the data. The predictions from all of them are then combined through a weighted majority vote (or sum) to produce the final prediction. The data modifications at each so-called boosting iteration consist of applying weights w_1, w_2, \dots, w_N to each of the training samples. Initially, those weights are all set to $w_i = 1/N$, so that the first step simply trains a weak learner on the original data. For each successive iteration, the sample weights are individually modified and the learning algorithm is reapplied to the reweighted data.

AdaBoost can be used both for classification and regression problems:

8. Gradient Tree Boosting

Gradient Tree Boosting or Gradient Boosted Decision Trees (GBDT) is a generalization of boosting to arbitrary differentiable loss functions. GBDT is an accurate and effective

off-the-shelf procedure that can be used for both regression and classification problems in a variety of areas including Web search ranking and ecology.

Q3. Explain different kinds of voting techniques used for prediction.

Ans :

Voting Classifier

The idea behind the VotingClassifier is to combine conceptually different machine learning classifiers and use a majority vote or the average predicted probabilities (soft vote) to predict the class labels. Such a classifier can be useful for a set of equally well performing model in order to balance out their individual weaknesses.

Majority Class Labels (Majority/Hard Voting)

In majority voting, the predicted class label for a particular sample is the class label that represents the majority (mode) of the class labels predicted by each individual classifier.

E.g., if the prediction for a given sample is

- classifier 1 -> class 1
- classifier 2 -> class 1
- classifier 3 -> class 2

the VotingClassifier (with voting='hard') would classify the sample as "class 1" based on the majority class label.

In the cases of a tie, the VotingClassifier will select the class based on the ascending sort order. E.g., in the following scenario

- classifier 1 -> class 2
- classifier 2 -> class 1

the class label 1 will be assigned to the sample.

Weighted Average Probabilities (Soft Voting)

In contrast to majority voting (hard voting), soft voting returns the class label as argmax of the sum of predicted probabilities.

Specific weights can be assigned to each classifier via the weights parameter. When weights are provided, the predicted class probabilities for each classifier are collected, multiplied by the classifier weight, and averaged. The final class label is then derived from the class label with the highest average probability.

To illustrate this with a simple example, let's assume we have 3 classifiers and a 3-class classification problems where we assign equal weights to all classifiers: $w_1 = 1$, $w_2 = 1$, $w_3 = 1$.

The weighted average probabilities for a sample would then be calculated as follows:

classifier class 1 class 2 class 3

classifier 1 $w_1 * 0.2$ $w_1 * 0.5$ $w_1 * 0.3$

classifier 2 $w_2 * 0.6$ $w_2 * 0.3$ $w_2 * 0.1$

classifier 3 $w_3 * 0.3$ $w_3 * 0.4$ $w_3 * 0.3$

0.37 0.4 0.23

weighted average

Here, the predicted class label is 2, since it has the highest average probability.

Voting Regressor

The idea behind the VotingRegressor is to combine conceptually different machine learning regressors and return the average predicted values. Such a regressor can be useful for a set of equally well performing models in order to balance out their individual weaknesses.

Stacked generalization

Stacked generalization is a method for combining estimators to reduce their biases. More precisely, the predictions of each individual estimator are stacked together and used as input to a final estimator to compute the prediction. This final estimator is trained through cross-validation.

The Stacking Classifier and Stacking Regressor provide such strategies which can be applied to classification and regression problems.

Q4. Explain , how random forest algorithm works.

Ans :

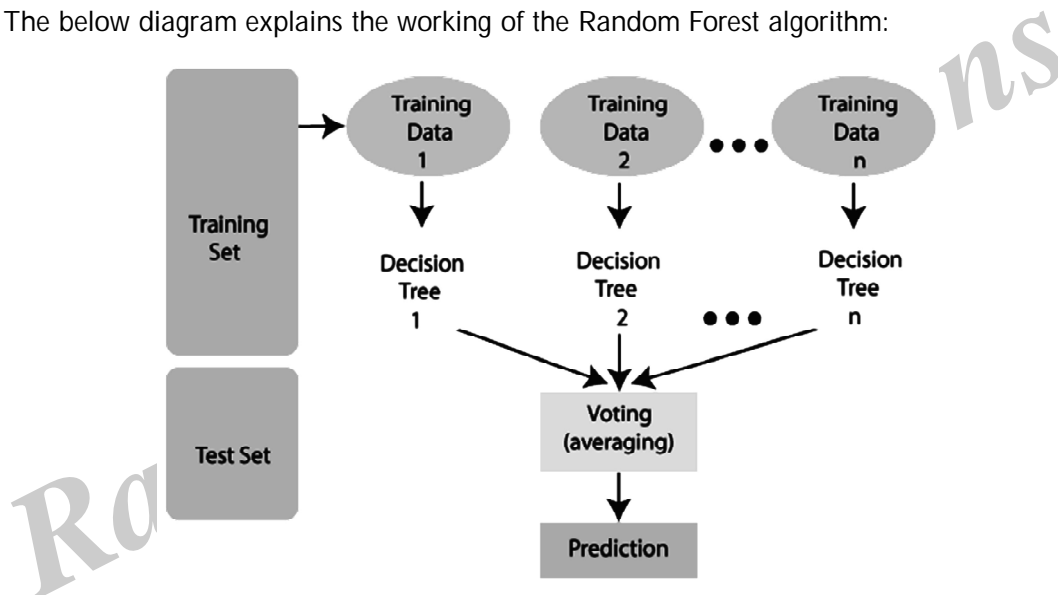
Random Forest Algorithm

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

The below diagram explains the working of the Random Forest algorithm:



Assumptions for Random Forest

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

Use of Random Forest

Below are some points that explain why we should use the Random Forest algorithm:

- It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

Random Forest algorithm

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

Step-1: Select random K data points from the training set.

Step-2: Build the decision trees associated with the selected data points (Subsets).

Step-3: Choose the number N for decision trees that you want to build.

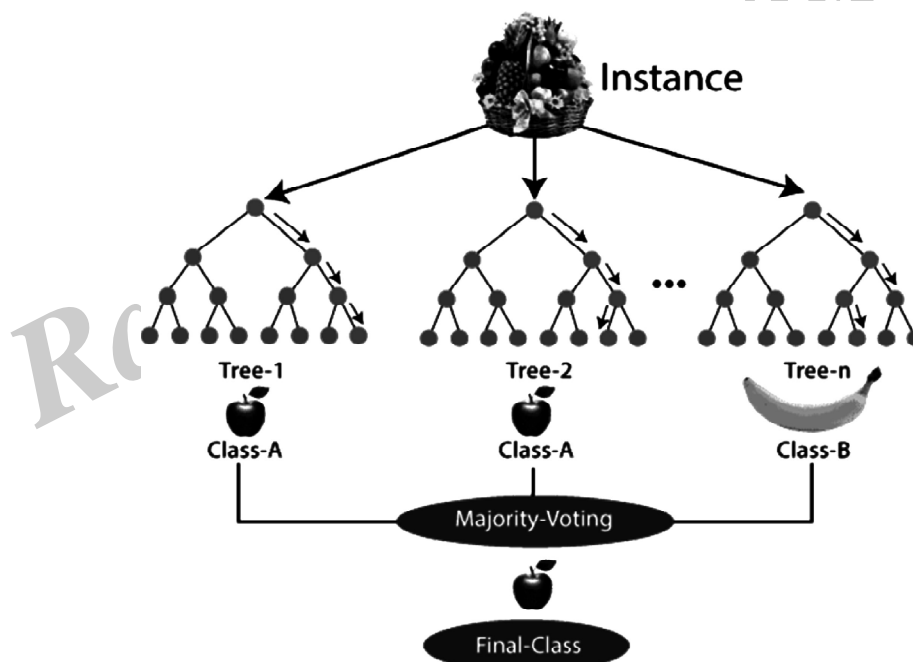
Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

The working of the algorithm can be better understood by the below example:

Example:

Suppose there is a dataset that contains multiple fruit images. So, this dataset is given to the Random forest classifier. The dataset is divided into subsets and given to each decision tree. During the training phase, each decision tree produces a prediction result, and when a new data point occurs, then based on the majority of results, the Random Forest classifier predicts the final decision. Consider the below image:



Applications of Random Forest

There are mainly four sectors where Random forest mostly used:

1. Banking: Banking sector mostly uses this algorithm for the identification of loan risk.
2. Medicine: With the help of this algorithm, disease trends and risks of the disease can be identified.
3. Land Use: We can identify the areas of similar land use by this algorithm.
4. Marketing: Marketing trends can be identified using this algorithm.

Advantages of Random Forest

- Random Forest is capable of performing both Classification and Regression tasks.
- It is capable of handling large datasets with high dimensionality.
- It enhances the accuracy of the model and prevents the overfitting issue.

Disadvantages of Random Forest

- Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

4.1.3 Bagging**Q5. What is Bagging technique? Why it is used for training data sets in ML**

Ans :

(Imp.)

Bagging is used when the goal is to reduce the variance of a decision tree classifier. Here the objective is to create several subsets of data from training sample chosen randomly with replacement. Each collection of subset data is used to train their decision trees. As a result, we get an ensemble of different models. Average of all the predictions from different trees are used which is more robust than a single decision tree classifier.

Bagging Steps

1. **Bootstrapping:** Bagging leverages a bootstrapping sampling technique to create diverse samples. This resampling method generates different subsets of the training dataset by selecting data points at random and with replacement. This means that each time you select a data point from the training dataset, you are able to select the same instance multiple times. As a result, a value/instance repeated twice (or more) in a sample.
2. **Parallel training:** These bootstrap samples are then trained independently and in parallel with each other using weak or base learners.
3. **Aggregation:** Finally, depending on the task (i.e. regression or classification), an average or a majority of the predictions are taken to compute a more accurate estimate. In the case of regression, an average is taken of all the outputs predicted by the individual classifiers; this is known as soft voting. For classification problems, the class with the highest majority of votes is accepted; this is known as hard voting or majority voting.

Benefits and challenges of bagging

There are a number of key advantages and challenges that the bagging method presents when used for classification or regression problems. The key benefits of bagging include:

- **Ease of implementation:** Python libraries such as scikit-learn (also known as sklearn) make it easy to combine the predictions of base learners or estimators to improve model performance. Their documentation (link resides outside IBM) lays out the available modules that you can leverage in your model optimization.
- **Reduction of variance:** Bagging can reduce the variance within a learning algorithm. This is particularly helpful with high-dimensional data, where missing values can lead to higher variance, making it more prone to overfitting and preventing accurate generalization to new datasets.

The key challenges of bagging include:

- **Loss of interpretability:** It's difficult to draw very precise business insights through bagging because due to the averaging involved across predictions. While the output is more precise than any individual

data point, a more accurate or complete dataset could also yield more precision within a single classification or regression model.

- **Computationally expensive:** Bagging slows down and grows more intensive as the number of iterations increase. Thus, it's not well-suited for real-time applications. Clustered systems or a large number of processing cores are ideal for quickly creating bagged ensembles on large test sets.
- **Less flexible:** As a technique, bagging works particularly well with algorithms that are less stable. One that are more stable or subject to high amounts of bias do not provide as much benefit as there's less variation within the dataset of the model. As noted in the Hands-On Guide to Machine Learning (link resides outside of IBM), "bagging a linear regression model will effectively just return the original predictions for large enough b ."

Applications of Bagging

The bagging technique is used across a large number of industries, providing insights for both real-world value and interesting perspectives, such as in the GRAMMY Debates with Watson. Key use cases include:

- **Healthcare:** Bagging has been used to form medical data predictions. For example, research (PDF, 2.8 MB) (link resides outside IBM) shows that ensemble methods have been used for an array of bioinformatics problems, such as gene and/or protein selection to identify a specific trait of interest. More specifically, this research (link resides outside IBM) delves into its use to predict the onset of diabetes based on various risk predictors.
- **IT:** Bagging can also improve the precision and accuracy in IT systems, such as ones network intrusion detection systems. Meanwhile, this research (link resides outside IBM) looks at how bagging can improve the accuracy of network intrusion detection—and reduce the rates of false positives.
- **Environment:** Ensemble methods, such as bagging, have been applied within the field of remote sensing. More specifically, this research (link resides outside IBM) shows how it has been used to map the types of wetlands within a coastal landscape.
- **Finance:** Bagging has also been leveraged with deep learning models in the finance industry, automating critical tasks, including fraud detection, credit risk evaluations, and option pricing problems. This research (link resides outside IBM) demonstrates how bagging among other machine learning techniques have been leveraged to assess loan default risk. This study (link resides outside IBM) highlights how bagging helps to minimize risk by to prevent credit card fraud within banking and financial institutions.

4.1.4 Adaboost/Gradient Boosting

Q6. Write about the importance of AdaBoosting.

Ans :

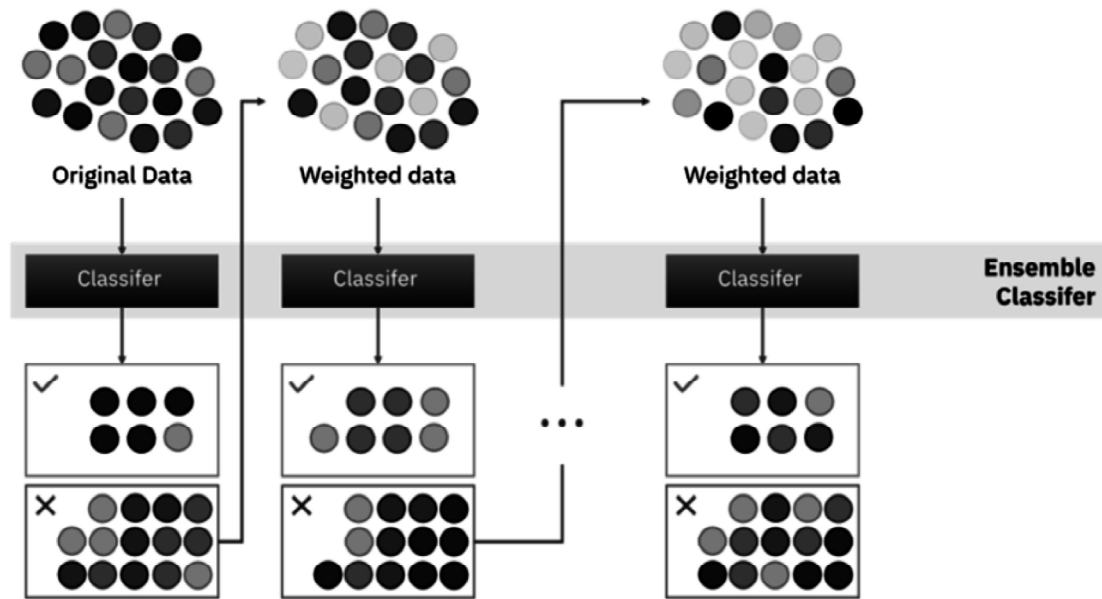
AdaBoost

This is a type of ensemble technique, where a number of weak learners are combined together to form a strong learner. Here, usually, each weak learner is developed as decision stumps (A stump is a tree with just a single split and two terminal nodes) that are used to classify the observations.

In contrary to the random forest, here each classifier has different weights assigned to it based on the classifier's performance (more weight is assigned to the classifier when accuracy is more and vice-verse), weights are also assigned to the observations at the end of every round, in such a way that wrongly

predicted observations have increased weight resulting in their probability of being picked more often in the next classifier's sample. So, the consecutive training set depends on their previous training set, and hence correlation exists between the built trees.

Thus, Adaboost increases the predictive accuracy by assigning weights to both observations at end of every tree and weights(scores) to every classifier. Hence, in Adaboost, every classifier has a different weightage on final prediction contrary to the random forest where all trees are assigned equal weights.



Mathematically, the error rate of a classifier is given by-

$$\epsilon_i = \frac{1}{N} \sum_{j=1}^N w_j I(C_i(x_j) \neq y_j)$$

Where w_j is weight assigned to each observation of training set D_i [Initially, for the first classifier, it is equally distributed]. I is nothing but the Identity function to filter out wrong predictions made by the classifier.

The importance of classifier is given by-

$$\alpha_i = \ln \left(\frac{1 - \epsilon_i}{\epsilon_i} \right)$$

Where α_i ranges from $(-\infty)$ if ϵ_i is 1 and $(+\infty)$ if ϵ_i is 0.

The mechanism of updating weights to observation is-

$$w_i^{j+1} = \frac{w_i^j}{Z_j} + \begin{cases} \exp^{-\alpha_i} & \text{if } C_i(x_j) = y_i \\ \exp^{\alpha_i} & \text{if } C_i(x_j) \neq y_j \end{cases}$$

The function $\exp^{(-\alpha(i))}$ reduces the weight of correctly classified observation and $\exp^{(\alpha(i))}$ increases the weight of misclassified observation, thus, increasing their chance of getting picked. Z_j is the normalization factor that is used to reassign weights of observation such that all weights sum up to 1.

The final prediction of each observation is made by aggregating the weighted average of the prediction made by each classifier. AdaBoost might result in overfitting. Hence, no. of trees should be checked and restricted.

Q7. What is the use of Gradient Boosting ? Explain

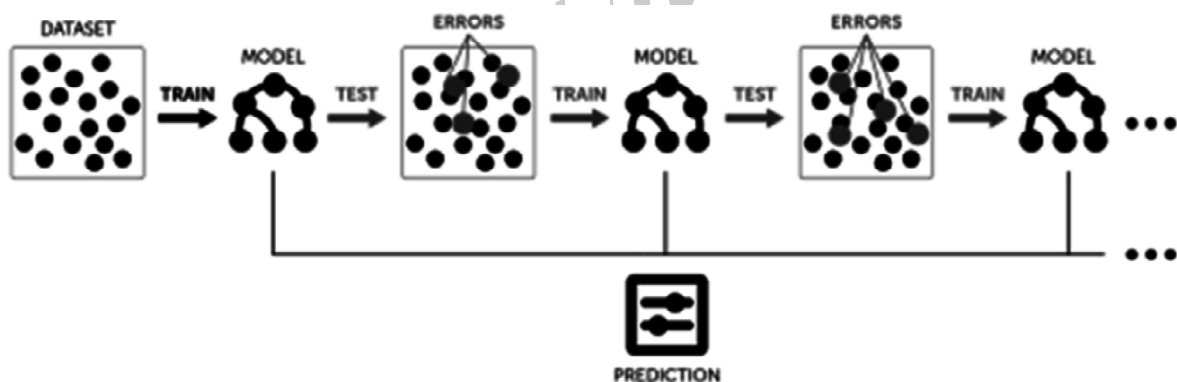
Ans :

Gradient Boosting Machine (GBM)

Just like AdaBoost, Gradient Boost also combines a no. of weak learners to form a strong learner. Here, the residual of the current classifier becomes the input for the next consecutive classifier on which the trees are built, and hence it is an additive model. The residuals are captured in a step-by-step manner by the classifiers, in order to capture the maximum variance within the data, this is done by introducing the learning rate to the classifiers.

By this method, we are slowly inching in the right direction towards better prediction (This is done by identifying negative gradient and moving in the opposite direction to reduce the loss, hence it is called Gradient Boosting in line with Gradient Descent where similar logic is employed). Thus, by no. of classifiers, we arrive at a predictive value very close to the observed value.

Initially, a tree with a single node is built which predicts the aggregated value of Y in case of regression or log(odds) of Y for classification problems after which trees with greater depth are grown on previous classifier's residuals. Usually, in GBM, every tree is grown with 8-32 terminal nodes, and around 100 trees are grown. Learning rates are given as constant for every tree so that the model takes small steps in the right direction to capture the variance and train the classifier on it. Unlike Adaboost, here all the trees are given equal weights.



Let us run through step by step execution of the GBM algorithm,

Input Data (x_i, y_i) where $i = 1(1)n$ where

n being total no. of observations in the sample and differentiable loss

function $L(Y_i, F(x))$.

For a regression problem, there are different variants of loss function like Absolute Loss, Huber Loss. Generally, we use the squared residual loss for the reason of computational simplicity. For a classification problem, the loss function is estimated from negative log-likelihood which is converted to log(odds).

Step 1: Initialize the model with a constant value

$$F_0(x) = \underbrace{\arg \min}_{\gamma} \sum_{i=1}^n L(Y_i, \gamma)$$

where L is the loss function and Y_i is observed and γ is the predicted value. This is the first tree that is grown with a single leaf after which trees with greater depth are built. Usually, it is the average of Y_i values for regression and $\log(\text{odds})$ value for classification.

Step 2: For $m = 1$ to M where M is the total no. of trees to be built

(a) Compute $\gamma_{im} = - \left[\frac{\partial L(Y_i, F(x_i))}{\partial F(x_i)} \right]_{F(x)=F_{(m-1)}(x)}$ for $i = 1, \dots, n$

This equation gives the negative gradient of each observation for every tree which includes the previous classifier's predicted values.

(b) Fit a regression tree on the γ_{im} values and create terminal regions R_{jm} for $j = 1, \dots, J_m$

(c) For $j = 1, \dots, J_m$ compute $\gamma_{jm} = \underbrace{\arg \min}_{\gamma} \sum_{x_i \in R_{jm}} L(Y_i, F_{m-1}(x_i) + \gamma)$

This equation finds the aggregated predicted values of each terminal nodes of every tree with minimized loss function including previous' learners prediction

(d) Update $F_m(x) = F_{m-1}(x) + \eta \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$

Where η is the learning rate of every tree which reduces the effect of each tree on final prediction thereby increasing the accuracy.

Step 3: Output $F_M(x)$

Thus, GBM makes the final prediction by cumulating inputs from all the trees.

Q8. Differentiate between AdaBoost and Gradient Boost Techniques

Ans :

Comparison between AdaBoost and Gradient Boost

S.No	Adaboost	Gradient Boost
1	An additive model where shortcomings of previous models are identified by high-weight data points.	An additive model where shortcomings of previous models are identified by the gradient.
2	The trees are usually grown as decision stumps.	The trees are grown to a greater depth usually ranging from 8 to 32 terminal nodes.
3	Each classifier has different weights assigned to the final prediction based on its performance.	All classifiers are weighed equally and their predictive capacity is restricted with learning rate to increase accuracy.
4	It gives weights to both classifiers and observations thus capturing maximum variance within data.	It builds trees on previous classifier's residuals thus capturing variance in data.

4.1.5 Clustering

Q9. What is Clustering ? Write about various types of clustering Methods and algorithms.

Ans :

Clustering or cluster analysis is a machine learning technique, which groups the unlabelled dataset. It can be defined as "A way of grouping the data points into different clusters, consisting of similar data points. The objects with the possible similarities remain in a group that has less or no similarities with another group."

It does it by finding some similar patterns in the unlabelled dataset such as shape, size, color, behavior, etc., and divides them as per the presence and absence of those similar patterns.

It is an unsupervised learning method, hence no supervision is provided to the algorithm, and it deals with the unlabeled dataset.

After applying this clustering technique, each cluster or group is provided with a cluster-ID. ML system can use this id to simplify the processing of large and complex datasets.

The clustering technique is commonly used for statistical data analysis.

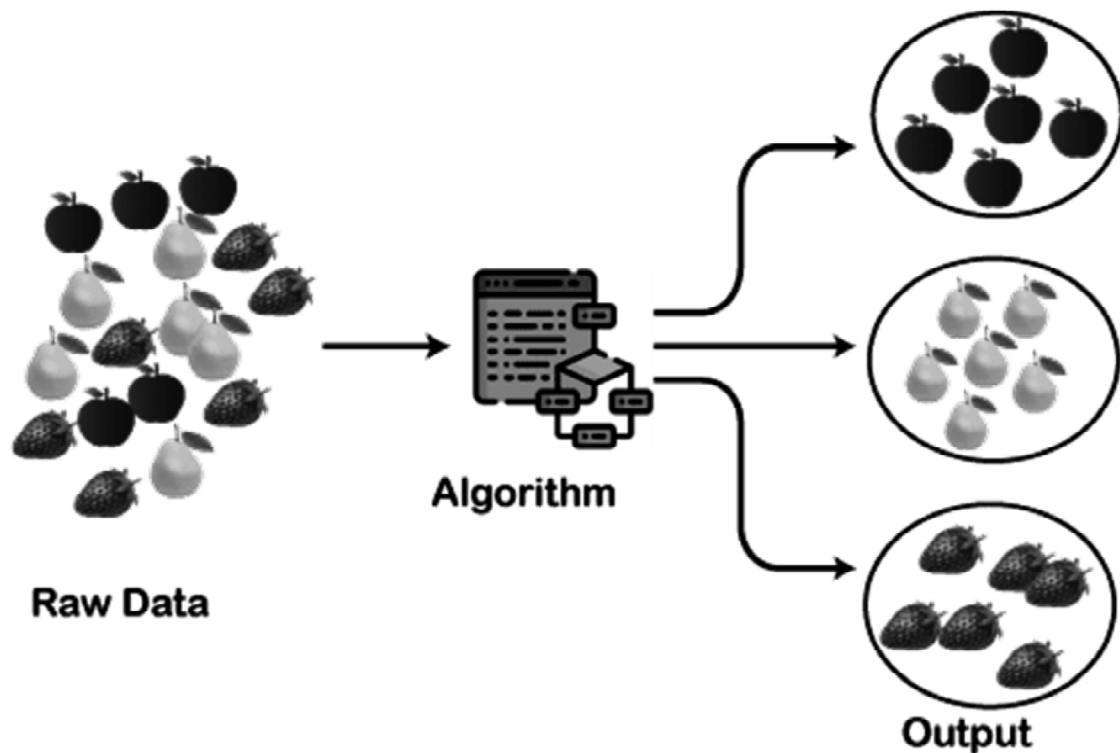
Example: Let's understand the clustering technique with the real-world example of Mall: When we visit any shopping mall, we can observe that the things with similar usage are grouped together. Such as the t-shirts are grouped in one section, and trousers are at other sections, similarly, at vegetable sections, apples, bananas, Mangoes, etc., are grouped in separate sections, so that we can easily find out the things. The clustering technique also works in the same way. Other examples of clustering are grouping documents according to the topic.

The clustering technique can be widely used in various tasks. Some most common uses of this technique are:

- Market Segmentation
- Statistical data analysis
- Social network analysis
- Image segmentation
- Anomaly detection, etc.

Apart from these general usages, it is used by the Amazon in its recommendation system to provide the recommendations as per the past search of products. Netflix also uses this technique to recommend the movies and web-series to its users as per the watch history.

The below diagram explains the working of the clustering algorithm. We can see the different fruits are divided into several groups with similar properties.



Types of Clustering Methods

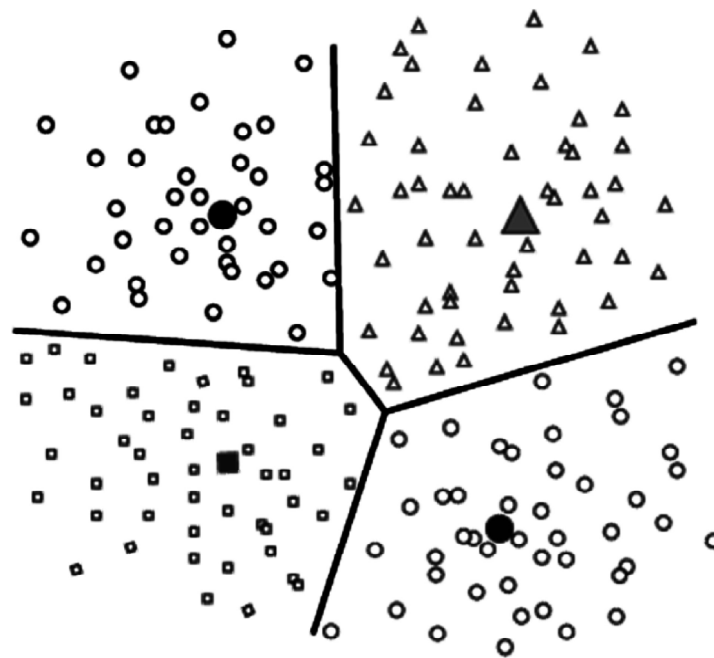
The clustering methods are broadly divided into Hard clustering (datapoint belongs to only one group) and Soft Clustering (data points can belong to another group also). But there are also other various approaches of Clustering exist. Below are the main clustering methods used in Machine learning:

1. Partitioning Clustering
2. Density-Based Clustering
3. Distribution Model-Based Clustering
4. Hierarchical Clustering
5. Fuzzy Clustering

1. Partitioning Clustering

It is a type of clustering that divides the data into non-hierarchical groups. It is also known as the centroid-based method. The most common example of partitioning clustering is the K-Means Clustering algorithm.

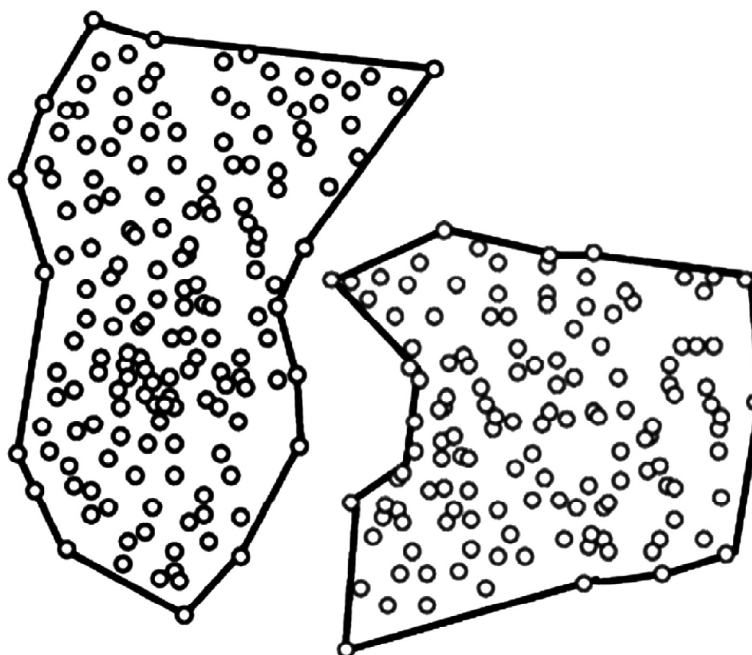
In this type, the dataset is divided into a set of k groups, where K is used to define the number of pre-defined groups. The cluster center is created in such a way that the distance between the data points of one cluster is minimum as compared to another cluster centroid.



2. Density-Based Clustering

The density-based clustering method connects the highly-dense areas into clusters, and the arbitrarily shaped distributions are formed as long as the dense region can be connected. This algorithm does it by identifying different clusters in the dataset and connects the areas of high densities into clusters. The dense areas in data space are divided from each other by sparser areas.

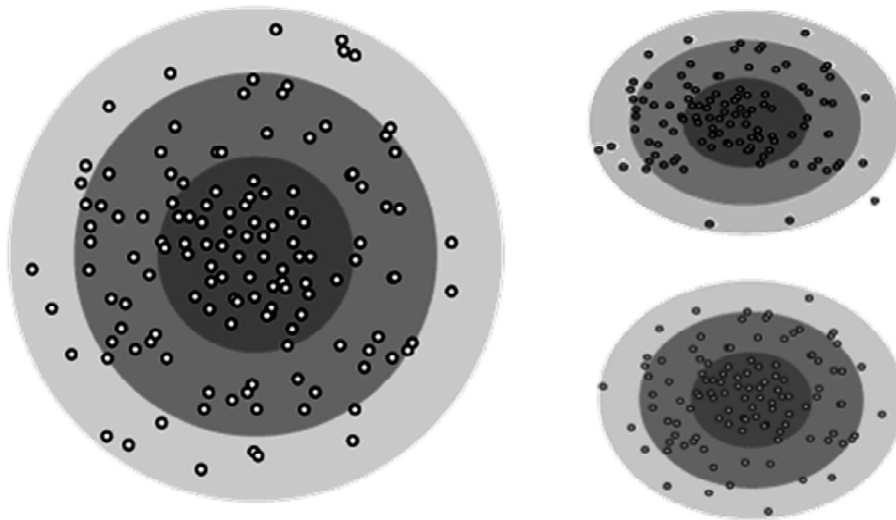
These algorithms can face difficulty in clustering the data points if the dataset has varying densities and high dimensions.



3. Distribution Model-Based Clustering

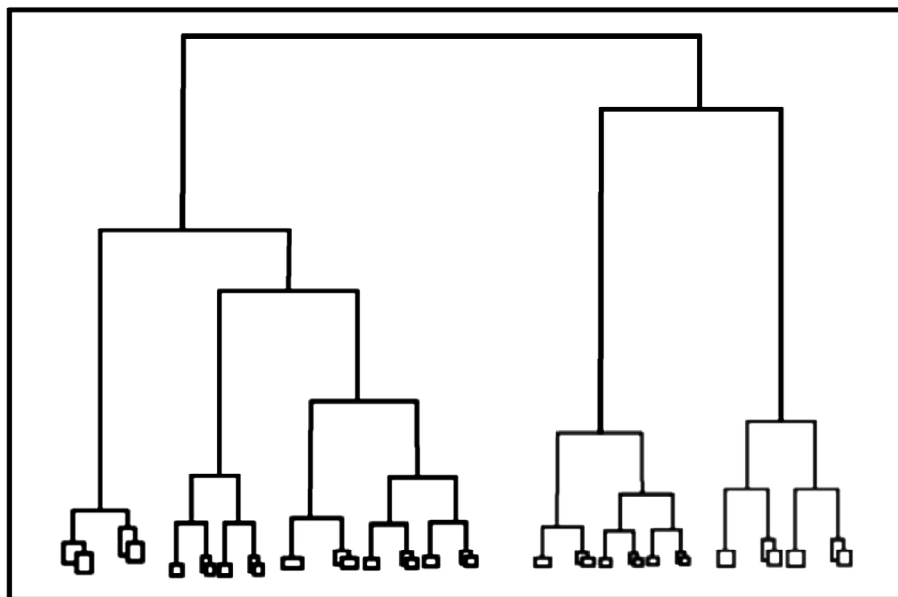
In the distribution model-based clustering method, the data is divided based on the probability of how a dataset belongs to a particular distribution. The grouping is done by assuming some distributions commonly Gaussian Distribution.

The example of this type is the Expectation-Maximization Clustering algorithm that uses Gaussian Mixture Models (GMM).



4. Hierarchical Clustering

Hierarchical clustering can be used as an alternative for the partitioned clustering as there is no requirement of pre-specifying the number of clusters to be created. In this technique, the dataset is divided into clusters to create a tree-like structure, which is also called a dendrogram. The observations or any number of clusters can be selected by cutting the tree at the correct level. The most common example of this method is the Agglomerative Hierarchical algorithm.



5. Fuzzy Clustering

Fuzzy clustering is a type of soft method in which a data object may belong to more than one group or cluster. Each dataset has a set of membership coefficients, which depend on the degree of membership to be in a cluster. Fuzzy C-means algorithm is the example of this type of clustering; it is sometimes also known as the Fuzzy k-means algorithm.

Clustering Algorithms

The Clustering algorithms can be divided based on their models that are explained above. There are different types of clustering algorithms published, but only a few are commonly used. The clustering algorithm is based on the kind of data that we are using. Such as, some algorithms need to guess the number of clusters in the given dataset, whereas some are required to find the minimum distance between the observation of the dataset.

Here we are discussing mainly popular Clustering algorithms that are widely used in machine learning

1. K-Means algorithm

The k-means algorithm is one of the most popular clustering algorithms. It classifies the dataset by dividing the samples into different clusters of equal variances. The number of clusters must be specified in this algorithm. It is fast with fewer computations required, with the linear complexity of $O(n)$.

2. Mean-shift algorithm

Mean-shift algorithm tries to find the dense areas in the smooth density of data points. It is an example of a centroid-based model, that works on updating the candidates for centroid to be the center of the points within a given region.

3. DBSCAN Algorithm

It stands for Density-Based Spatial Clustering of Applications with Noise. It is an example of a density-based model similar to the mean-shift, but with some remarkable advantages. In this algorithm, the areas of high density are separated by the areas of low density. Because of this, the clusters can be found in any arbitrary shape.

4. Expectation-Maximization Clustering using GMM

This algorithm can be used as an alternative for the k-means algorithm or for those cases where K-means can be failed. In GMM, it is assumed that the data points are Gaussian distributed.

5. Agglomerative Hierarchical algorithm

The Agglomerative hierarchical algorithm performs the bottom-up hierarchical clustering. In this, each data point is treated as a single cluster at the outset and then successively merged. The cluster hierarchy can be represented as a tree-structure.

6. Affinity Propagation

It is different from other clustering algorithms as it does not require to specify the number of clusters. In this, each data point sends a message between the pair of data points until convergence. It has $O(N^2T)$ time complexity, which is the main drawback of this algorithm.

Applications of Clustering

Below are some commonly known applications of clustering technique in Machine Learning:

- **In Identification of Cancer Cells:** The clustering algorithms are widely used for the identification of cancerous cells. It divides the cancerous and non-cancerous data sets into different groups.

- **In Search Engines:** Search engines also work on the clustering technique. The search result appears based on the closest object to the search query. It does it by grouping similar data objects in one group that is far from the other dissimilar objects. The accurate result of a query depends on the quality of the clustering algorithm used.
- **Customer Segmentation:** It is used in market research to segment the customers based on their choice and preferences.
- **In Biology:** It is used in the biology stream to classify different species of plants and animals using the image recognition technique.
- **In Land Use:** The clustering technique is used in identifying the area of similar lands use in the GIS database. This can be very useful to find that for what purpose the particular land should be used, that means for which purpose it is more suitable.

Q10. Explain Hierarchical Clustering in Machine Learning.

Ans :

(Imp.)

Hierarchical Clustering in Machine Learning

Hierarchical clustering is another unsupervised machine learning algorithm, which is used to group the unlabeled datasets into a cluster and also known as hierarchical cluster analysis or HCA.

In this algorithm, we develop the hierarchy of clusters in the form of a tree, and this tree-shaped structure is known as the dendrogram.

Sometimes the results of K-means clustering and hierarchical clustering may look similar, but they both differ depending on how they work. As there is no requirement to predetermine the number of clusters as we did in the K-Means algorithm.

The hierarchical clustering technique has two approaches:

1. Agglomerative

Agglomerative is a bottom-up approach, in which the algorithm starts with taking all data points as single clusters and merging them until one cluster is left.

2. Divisive

Divisive algorithm is the reverse of the agglomerative algorithm as it is a top-down approach.

In this topic, we will discuss the Agglomerative Hierarchical clustering algorithm.

Agglomerative Hierarchical clustering

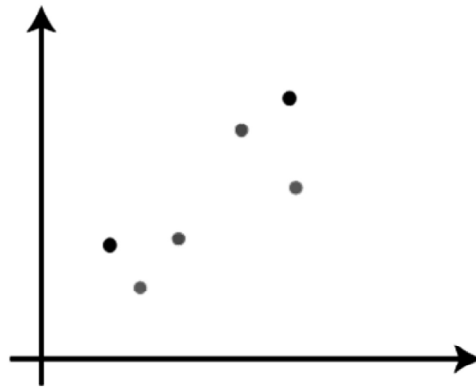
The agglomerative hierarchical clustering algorithm is a popular example of HCA. To group the datasets into clusters, it follows the bottom-up approach. It means, this algorithm considers each dataset as a single cluster at the beginning, and then start combining the closest pair of clusters together. It does this until all the clusters are merged into a single cluster that contains all the datasets.

This hierarchy of clusters is represented in the form of the dendrogram.

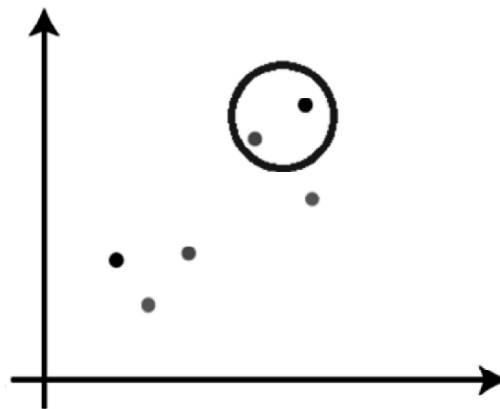
The working of the AHC algorithm can be explained using the below steps:

Step-1:

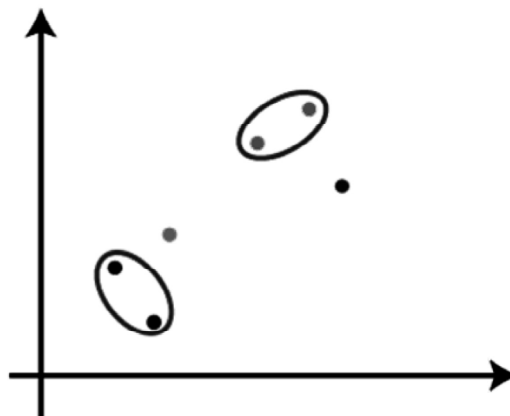
Create each data point as a single cluster. Let's say there are N data points, so the number of clusters will also be N.

**Step-2:**

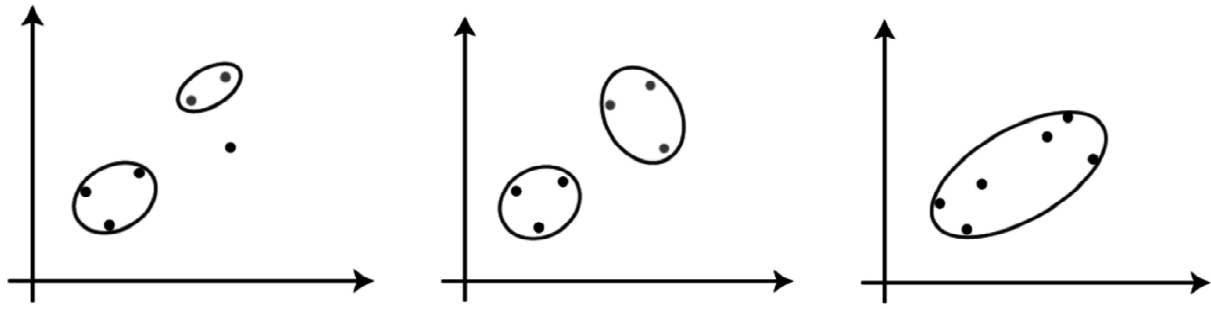
Take two closest data points or clusters and merge them to form one cluster. So, there will now be $N-1$ clusters.

**Step-3:**

Again, take the two closest clusters and merge them together to form one cluster. There will be $N-2$ clusters.

**Step-4:**

Repeat Step 3 until only one cluster left. So, we will get the following clusters. Consider the below images:

**Step-5:**

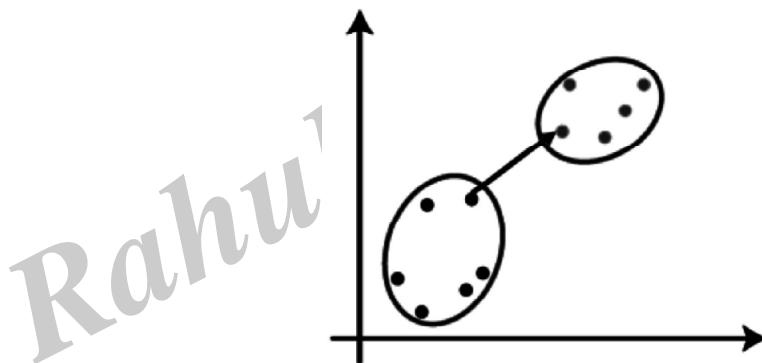
Once all the clusters are combined into one big cluster, develop the dendrogram to divide the clusters as per the problem.

Measure for the distance between two clusters

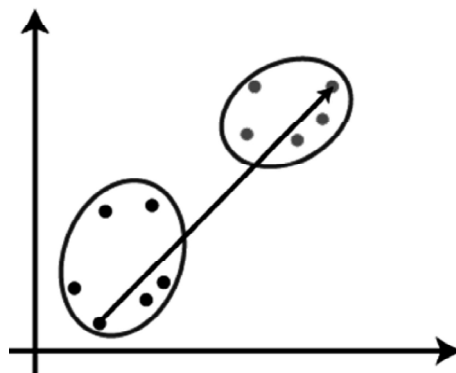
As we have seen, the closest distance between the two clusters is crucial for the hierarchical clustering. There are various ways to calculate the distance between two clusters, and these ways decide the rule for clustering. These measures are called Linkage methods. Some of the popular linkage methods are given below:

1. Single Linkage

It is the Shortest Distance between the closest points of the clusters. Consider the below image.

**2. Complete Linkage**

It is the farthest distance between the two points of two different clusters. It is one of the popular linkage methods as it forms tighter clusters than single-linkage.

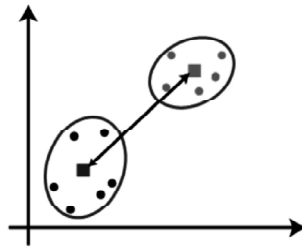


3. Average Linkage

It is the linkage method in which the distance between each pair of datasets is added up and then divided by the total number of datasets to calculate the average distance between two clusters. It is also one of the most popular linkage methods.

4. Centroid Linkage

It is the linkage method in which the distance between the centroid of the clusters is calculated. Consider the below image.

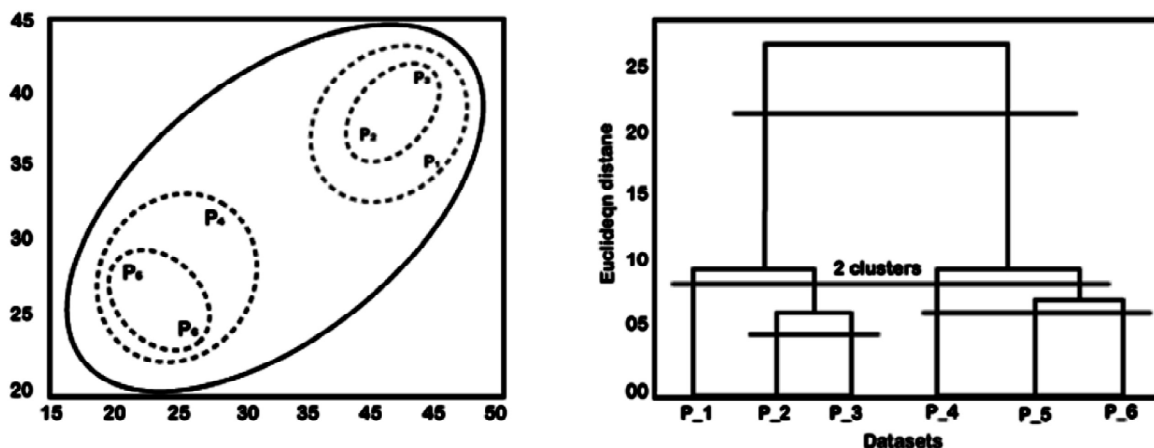


From the above-given approaches, we can apply any of them according to the type of problem or business requirement.

Working of Dendrogram in Hierarchical clustering

The dendrogram is a tree-like structure that is mainly used to store each step as a memory that the HC algorithm performs. In the dendrogram plot, the Y-axis shows the Euclidean distances between the data points, and the x-axis shows all the data points of the given dataset.

The working of the dendrogram can be explained using the below diagram:



In the above diagram, the left part is showing how clusters are created in agglomerative clustering, and the right part is showing the corresponding dendrogram.

- As we have discussed above, firstly, the datapoints P2 and P3 combine together and form a cluster, correspondingly a dendrogram is created, which connects P2 and P3 with a rectangular shape. The height is decided according to the Euclidean distance between the data points.
- In the next step, P5 and P6 form a cluster, and the corresponding dendrogram is created. It is higher than of previous, as the Euclidean distance between P5 and P6 is a little bit greater than the P2 and P3.

- Again, two new dendrograms are created that combine P1, P2, and P3 in one dendrogram, and P4, P5, and P6, in another dendrogram.
- At last, the final dendrogram is created that combines all the data points together.

We can cut the dendrogram tree structure at any level as per our requirement.

4.1.6 K-means

Q11. What is K-Means Algorithm? How does the K-Means Algorithm Work?

Ans.:

(Imp.)

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if $K=2$, there will be two clusters, and for $K=3$, there will be three clusters, and so on.

It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

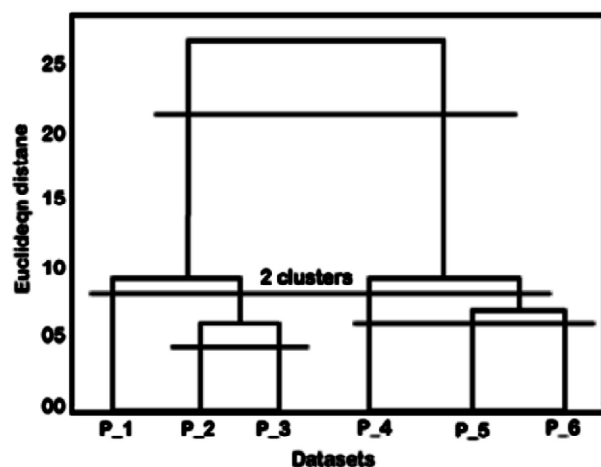
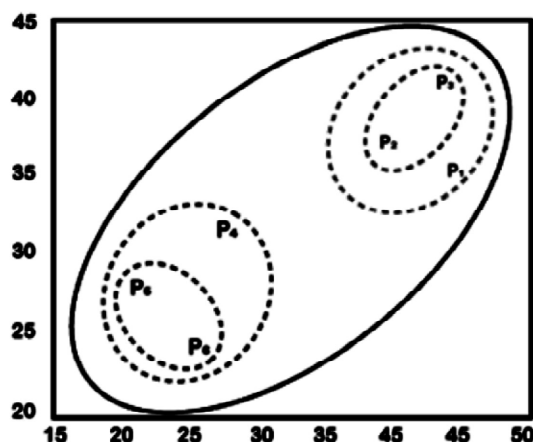
The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

Hence each cluster has datapoints with some commonalities, and it is away from other clusters.

The below diagram explains the working of the K-means Clustering Algorithm:



The working of the K-Means algorithm is explained in the below steps:

Step-1:

Select the number K to decide the number of clusters.

Step-2:

Select random K points or centroids. (It can be other from the input dataset).

Step-3:

Assign each data point to their closest centroid, which will form the predefined K clusters.

Step-4:

Calculate the variance and place a new centroid of each cluster.

Step-5:

Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

Step-6:

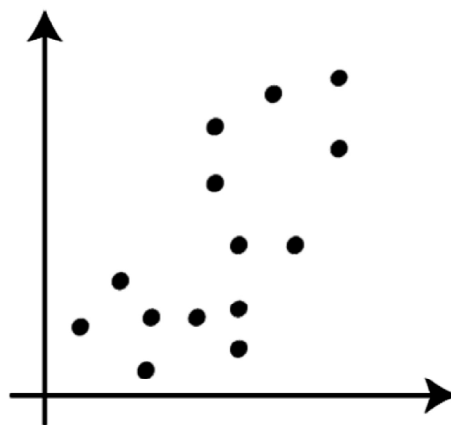
If any reassignment occurs, then go to step-4 else go to FINISH.

Step-7:

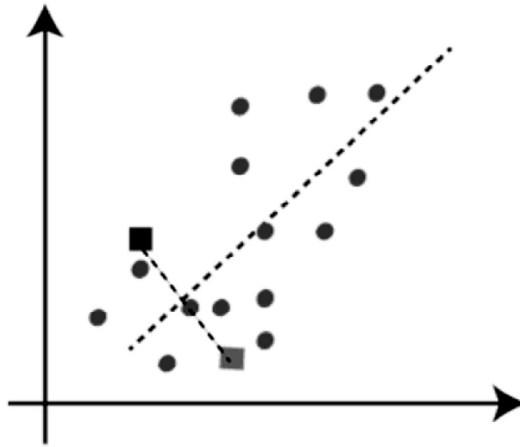
The model is ready.

Let's understand the above steps by considering the visual plots:

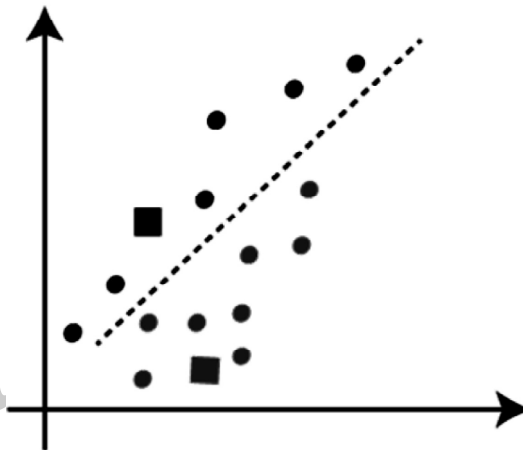
Suppose we have two variables M1 and M2. The x-y axis scatter plot of these two variables is given below:



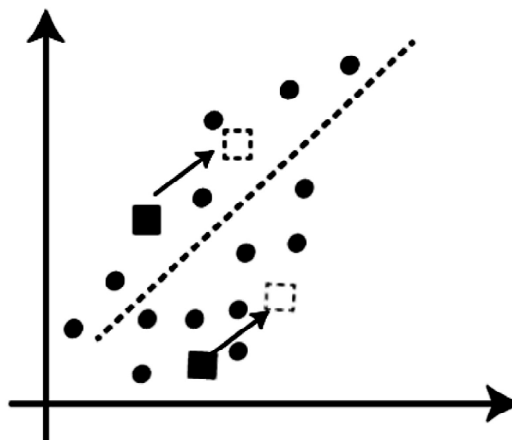
- Let's take number k of clusters, i.e., $K=2$, to identify the dataset and to put them into different clusters. It means here we will try to group these datasets into two different clusters.
- We need to choose some random k points or centroid to form the cluster. These points can be either the points from the dataset or any other point. So, here we are selecting the below two points as k points, which are not the part of our dataset. Consider the below image:



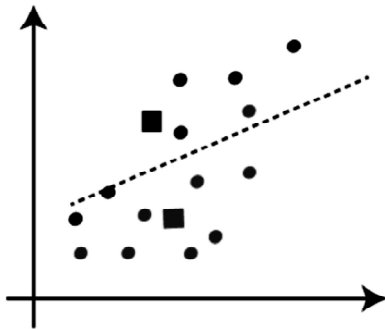
From the above image, it is clear that points left side of the line is near to the K1 or blue centroid, and points to the right of the line are close to the yellow centroid. Let's color them as blue and yellow for clear visualization.



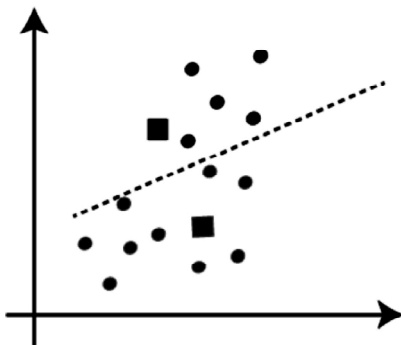
- As we need to find the closest cluster, so we will repeat the process by choosing a new centroid. To choose the new centroids, we will compute the center of gravity of these centroids, and will find new centroids as below:



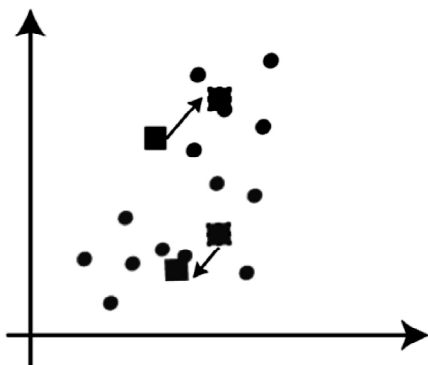
- Next, we will reassign each datapoint to the new centroid. For this, we will repeat the same process of finding a median line. The median will be like below image:



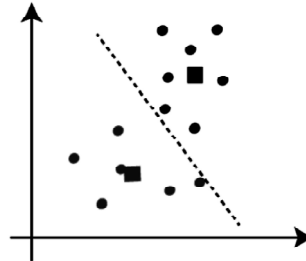
- From the above image, we can see, one yellow point is on the left side of the line, and two blue points are right to the line. So, these three points will be assigned to new centroids.



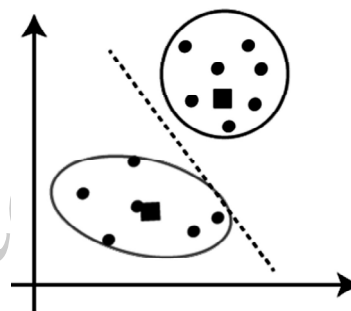
- As reassignment has taken place, so we will again go to the step-4, which is finding new centroids or K-points.
- We will repeat the process by finding the center of gravity of centroids, so the new centroids will be as shown in the below image:



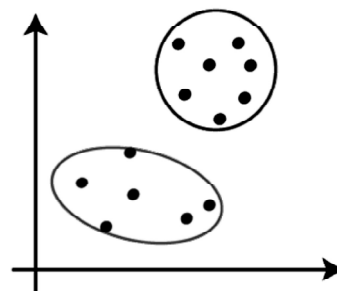
- As we got the new centroids so again will draw the median line and reassign the data points. So, the image will be:



- We can see in the above image; there are no dissimilar data points on either side of the line, which means our model is formed. Consider the below image:



As our model is ready, so we can now remove the assumed centroids, and the two final clusters will be as shown in the below image:



How to choose the value of "K number of clusters" in K-means Clustering?

The performance of the K-means clustering algorithm depends upon highly efficient clusters that it forms. But choosing the optimal number of clusters is a big task. There are some different ways to find the optimal number of clusters, but here we are discussing the most appropriate method to find the number of clusters or value of K. The method is given below:

Elbow Method

The Elbow method is one of the most popular ways to find the optimal number of clusters. This method uses the concept of WCSS value. WCSS stands for Within Cluster Sum of Squares, which defines the total variations within a cluster. The formula to calculate the value of WCSS (for 3 clusters) is given below:

$$WCSS = \sum_{P_i \text{ in Cluster } 1} \text{distance}(P_i, C_1)^2 + \sum_{P_i \text{ in Cluster } 2} \text{distance}(P_i, C_2)^2 + \sum_{P_i \text{ in Cluster } 3} \text{distance}(P_i, C_3)^2$$

In the above formula of WCSS,

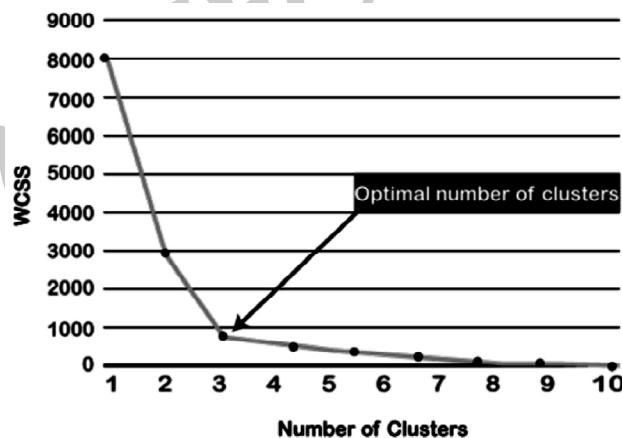
$\sum_{P_i \text{ in Cluster } 1} \text{distance}(P_i, C_1)^2$: It is the sum of the square of the distances between each data point and its centroid within a cluster1 and the same for the other two terms.

To measure the distance between data points and centroid, we can use any method such as Euclidean distance or Manhattan distance.

To find the optimal value of clusters, the elbow method follows the below steps:

- It executes the K-means clustering on a given dataset for different K values (ranges from 1-10).
- For each value of K, calculates the WCSS value.
- Plots a curve between calculated WCSS values and the number of clusters K.
- The sharp point of bend or a point of the plot looks like an arm, then that point is considered as the best value of K.

Since the graph shows the sharp bend, which looks like an elbow, hence it is known as the elbow method. The graph for the elbow method looks like the below image:



4.1.7 K-medoids

Q12. Explain K-Medoids Algorithm with example.

Ans :

- K-Medoids (also called as Partitioning Around Medoid) algorithm was proposed in 1987 by Kaufman and Rousseeuw.
- A medoid can be defined as the point in the cluster, whose dissimilarities with all the other points in the cluster is minimum.
- The dissimilarity of the medoid(C_i) and object(P_i) is calculated by using $E = |P_i - C_i|$

- The cost in K-Medoids algorithm is given as

$$c = \sum_{Ci} \sum_{Pi \in Ci} |Pi - Ci|$$

Algorithm

1. **Initialize:** select k random points out of the n data points as the medoids.
2. Associate each data point to the closest medoid by using any common distance metric methods.
3. While the cost decreases:

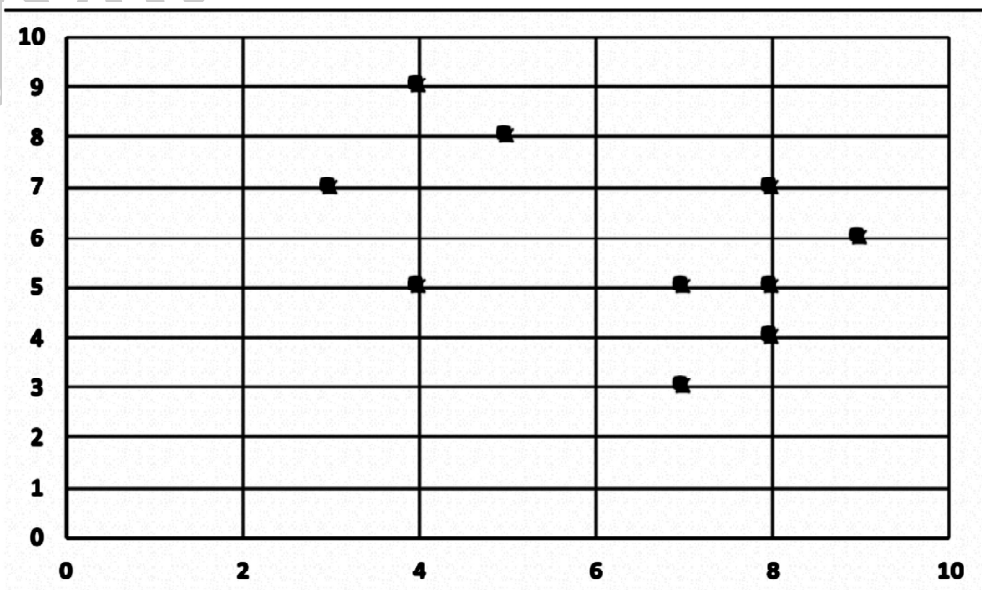
For each medoid m , for each data o point which is not a medoid:

1. Swap m and o , associate each data point to the closest medoid, recompute the cost.
2. If the total cost is more than that in the previous step, undo the swap.

Let's consider the following example:

	X	Y
0	8	7
1	3	7
2	4	9
3	9	6
4	8	5
5	5	8
6	7	3
7	8	4
8	7	5
9	4	5

If a graph is drawn using the above data points, we obtain the following:



Step 1:

Let the randomly selected 2 medoids, so select $k = 2$ and let **C1** **-(4, 5)** and **C2** **-(8, 5)** are the two medoids.

Step 2: Calculating cost.

The dissimilarity of each non-medoid point with the medoids is calculated and tabulated:

	X	Y	Dissimilarity from C1	Dissimilarity from C2
0	8	7	6	2
1	3	7	3	7
2	4	9	4	8
3	9	6	6	2
4	8	5	-	-
5	5	8	4	6
6	7	3	5	3
7	8	4	5	1
8	7	5	3	1
9	4	5	-	-

Each point is assigned to the cluster of that medoid whose dissimilarity is less.

The points 1, 2, 5 go to cluster C1 and 0, 3, 6, 7, 8 go to cluster C2.

The Cost = $(3 + 4 + 4) + (3 + 1 + 1 + 2 + 2) = 20$

Step 3: Randomly select one non-medoid point and recalculate the cost

Let the randomly selected point be (8, 4). The dissimilarity of each non-medoid point with the medoids – C1 (4, 5) and C2 (8, 4) is calculated and tabulated.

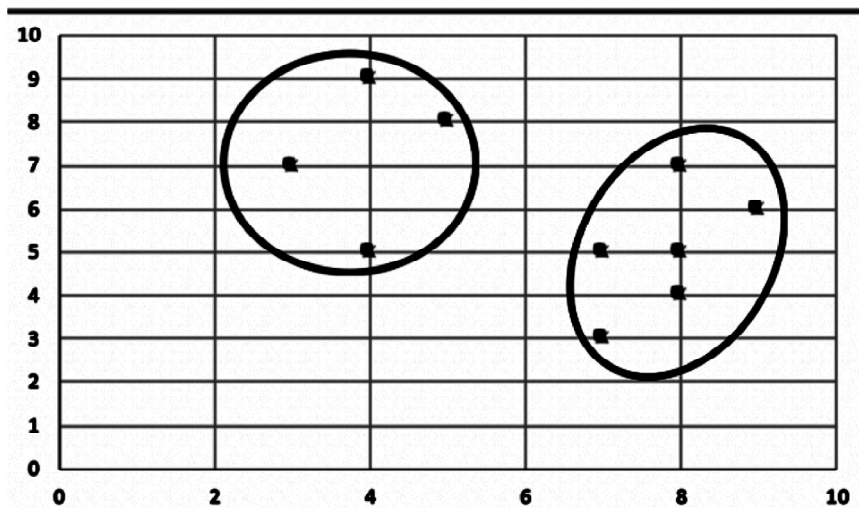
	X	Y	Dissimilarity from C1	Dissimilarity from C2
0	8	7	6	3
1	3	7	3	8
2	4	9	4	9
3	9	6	6	3
4	8	5	4	1
5	5	8	4	7
6	7	3	5	2
7	8	4	-	-
8	7	5	3	2
9	4	5	-	-

Each point is assigned to that cluster whose dissimilarity is less. So, the points 1, 2, 5 go to cluster C1 and 0, 3, 6, 7, 8 go to cluster C2.

The New cost = $(3 + 4 + 4) + (2 + 2 + 1 + 3 + 3) = 22$

Swap Cost = New Cost – Previous Cost = $22 - 20 = 2 > 0$

As the swap cost is not less than zero, we undo the swap. Hence (3, 4) and (7, 4) are the final medoids. The clustering would be in the following way



Advantages:

1. It is simple to understand and easy to implement.
2. K-Medoid Algorithm is fast and converges in a fixed number of steps.
3. PAM is less sensitive to outliers than other partitioning algorithms.

Disadvantages:

1. The main disadvantage of K-Medoid algorithms is that it is not suitable for clustering non-spherical (arbitrary shaped) groups of objects. This is because it relies on minimizing the distances between the non-medoid objects and the medoid (the cluster centre) – briefly, it uses compactness as clustering criteria instead of connectivity.
2. It may obtain different results for different runs on the same dataset because the first k medoids are chosen randomly.

4.1.8 Density-based Hierarchical Spectral

Q13. Explain DBSCAN Algorithm with example.

Ans :

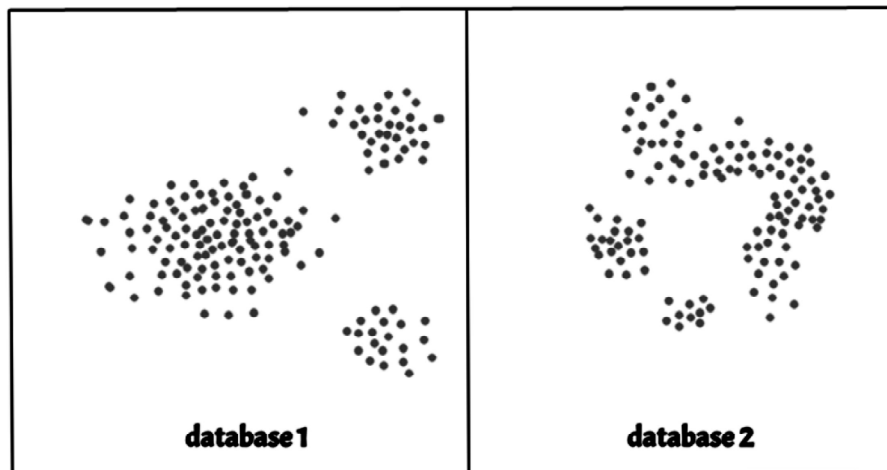
(Imp.)

Clustering analysis or simply Clustering is basically an Unsupervised learning method that divides the data points into a number of specific batches or groups, such that the data points in the same groups have similar properties and data points in different groups have different properties in some sense. It comprises of many different methods based on different evolution.

E.g. K-Means (distance between points), Affinity propagation (graph distance), Mean-shift (distance between points), DBSCAN (distance between nearest points), Gaussian mixtures (Mahalanobis distance to centers), Spectral clustering (graph distance) etc.

Fundamentally, all clustering methods use the same approach i.e. first we calculate similarities and then we use it to cluster the data points into groups or batches. Here we will focus on Density-based spatial clustering of applications with noise (DBSCAN) clustering method.

Clusters are dense regions in the data space, separated by regions of the lower density of points. The DBSCAN algorithm is based on this intuitive notion of “clusters” and “noise”. The key idea is that for each point of a cluster, the neighborhood of a given radius has to contain at least a minimum number of points.



Why DBSCAN ?

Partitioning methods (K-means, PAM clustering) and hierarchical clustering work for finding spherical-shaped clusters or convex clusters. In other words, they are suitable only for compact and well-separated clusters. Moreover, they are also severely affected by the presence of noise and outliers in the data.

Real life data may contain irregularities, like :

- i) Clusters can be of arbitrary shape such as those shown in the figure below.
- ii) Data may contain noise.



The figure below shows a data set containing nonconvex clusters and outliers/noises. Given such data, k-means algorithm has difficulties for identifying these clusters with arbitrary shapes.

DBSCAN algorithm requires two parameters:

1. eps

It defines the neighborhood around a data point i.e. if the distance between two points is lower or equal to 'eps' then they are considered as neighbors. If the eps value is chosen too small then large part of the data will be considered as outliers. If it is chosen very large then the clusters will merge and majority of the data points will be in the same clusters. One way to find the eps value is based on the k-distance graph.

2. MinPts

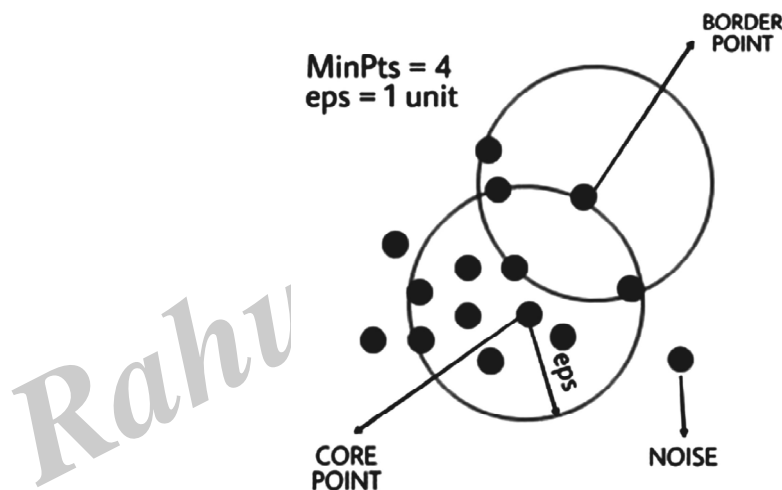
Minimum number of neighbors (data points) within eps radius. Larger the dataset, the larger value of MinPts must be chosen. As a general rule, the minimum MinPts can be derived from the number of dimensions D in the dataset as, $\text{MinPts} \geq D + 1$. The minimum value of MinPts must be chosen at least 3.

In this algorithm, we have 3 types of data points.

Core Point: A point is a core point if it has more than MinPts points within eps.

Border Point: A point which has fewer than MinPts within eps but it is in the neighborhood of a core point.

Noise or outlier: A point which is not a core point or border point.

**DBSCAN algorithm can be abstracted in the following steps:**

1. Find all the neighbor points within eps and identify the core points or visited with more than MinPtsneighbors.
2. For each core point if it is not already assigned to a cluster, create a new cluster.
3. Find recursively all its density connected points and assign them to the same cluster as the core point.

A point a and b are said to be density connected if there exist a point c which has a sufficient number of points in its neighbors and both the points a and b are within the eps distance. This is a chaining process. So, if b is neighbor of c, c is neighbor of d, d is neighbor of e, which in turn is neighbor of a implies that b is neighbor of a.

4. Iterate through the remaining unvisited points in the dataset. Those points that do not belong to any cluster are noise.

UNIT V

Miscellaneous topics: Expectation Maximization, GMMs, Learning theory Introduction to Reinforcement Learning.

Graphical Models: Bayesian Networks. Use Cases of various ML Algorithms in Manufacturing, Retail, Transport, Healthcare, weather, insurance sectors.

5.1 MISCELLANEOUS TOPICS

5.1.1 Expectation Maximization

Q1. What is known as Expectation minimization. Explain EM algorithm.

Ans : (Imp.)

The EM algorithm is the technique that can be deployed in order to determine the local maximum likelihood estimates/ parameters (MLE) or maximum a posteriori (MAP) estimates/parameters for latent variables (unobservable variables that are inferred from observable variables) in statistical models.

Or simply, the EM algorithm in machine learning uses observable instances of latent variables in order to predict values in instances, unobservable for learning, and continues till the convergence of the values takes place.

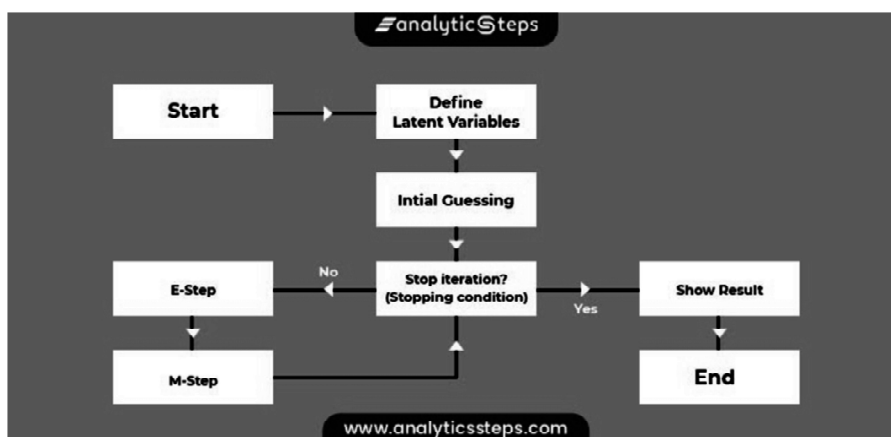
Being an iterative approach, the EM algorithm revolves amid two modes, the first mode estimates the missing or latent variables, called E-step, and the second step optimizes the parameters of the model that explains data more clearly, called M-step. i.e.

- **E-step:** Estimates the missing values in the dataset,
- **M-step:** Maximize the model parameters while the data is present.

The algorithm is used for predicting these values or in computing missing or incomplete data, given the generalized form of probability distribution that is connected with these latent variables.

Working of EM algorithm

Now, let's understand the working mechanism of this algorithm,



Workflow of EM algorithm

Step 1:

As having one set of missing or incomplete data and another set of starting parameters, we assume that observed data or initial values of the parameters are produced from the specific model.

Therefore, an entire set of incomplete observed data is provided to the system, assuming that an observed data comes from a specific model.

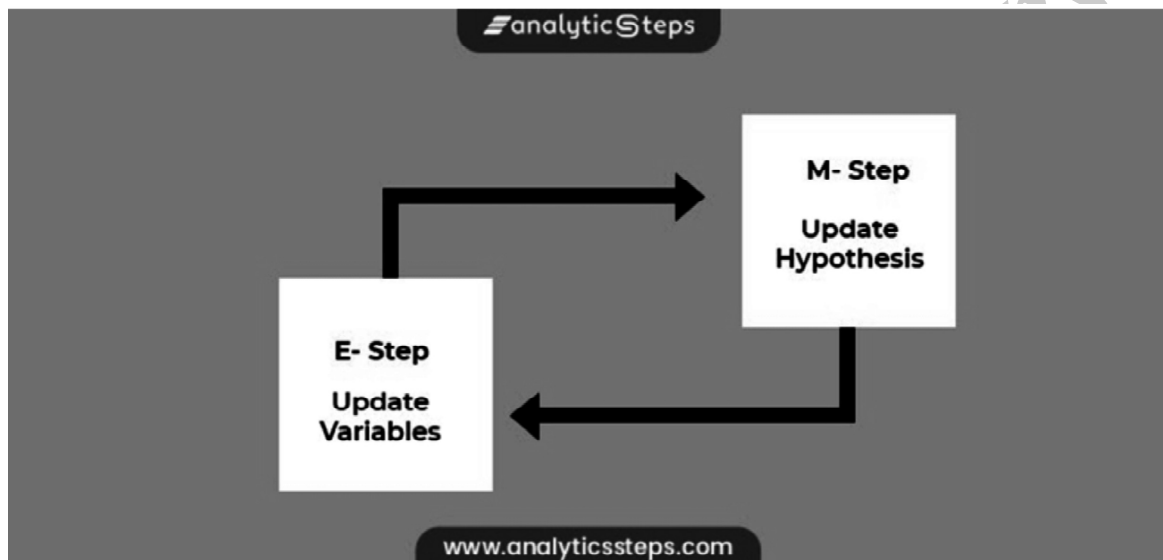
Step 2:

Depending on the observable value of the observable instances of the available data, next the values of unobservable instances, or missing data are predicted or estimated. This step is known as Expectation step, or E-step.

Basically, in this step, the observed data is used for estimating or guessing missing or incomplete data values that are used to update the variables.

Step 3:

By the produced data from E-step, next we update the parameters and complete the data set. This step is known as Maximization step, or M-step. And we update the hypothesis.

**Working of E-step and M-step**

As the last step, we check whether the values are converging or not, if yes, stop the process.

If not, then step 2 and step 3 will be imitated until the state of convergence is achieved, or simply, we will repeat E-step and M-step if the values are not converging. (From)

Properties of EM Algorithm

The EM algorithm has multiple applicable properties

1. It has numerical stability, with EM iteration increasing the likelihood.
2. In underlying-favorable circumstances, it has reliable-universal convergence.
3. Being implemented analytically and computationally, It is very easy to program and seek a tiny storage slot. Simply, by observing the continuance expansion in likelihood, in case computed easily, it becomes easier to regulate convergence and programming eros. (As stated in McLachlan and Krishnan, 1997, Section 1.7)

4. Having minor cost at per iteration, maximum number of iterations can be counterbalanced as such required for the EM algorithm as compared to other methods.
5. Most of the time, solutions to the M-steps reside in the closed form.
6. It can be globally accepted to obtain estimates of missing data.

Drawbacks of EM Algorithm

Besides that, some drawbacks are listed below:

1. It is unable to give automated estimates of the covariance matrix of the parameter estimates, yet such a drawback can be eliminated by applying appropriate methodology, concerning the EM algorithm.
2. Sometimes, it becomes very slow at convergence, and makes convergence only to local optima.
3. In a few cases, the E- and M-step could be unmanageable analytically.
4. It demands both forward and backward probabilities (as numerical optimization needs forward probability only). (Source)

Applications of EM Algorithm

The algorithm has plenty of applications of real-world applications in machine learning, some of them are;

1. Chosen in unsupervised data clustering and psychiatric analysis.
2. It has numerous uses in NLP, computer vision, and quantitative analysis of genetics.
3. Widely used in image reconstruction in the realm of medicine and structural engineering.
4. Adopted to measure the Gaussian density of a function.
5. Used in predicting the Hidden Markov Model (HMM) parameters and similar mixed models.
6. Used in filling missing data in a sample.
7. Explore the value of latent variables.

Q2. Explain, How can we present EM algorithm using probabilistic model.

Ans : (Imp.)

EM algorithm presentation using probabilistic model

Let us start again by formally presenting the problem using mathematical notations. In a probabilistic model, there are visible variables (y), latent variables (z) and associating parameter (θ). The likelihood $p(y|\theta)$, which we aim to maximize, measures the probability of the observables (height) given the parameters (group characteristics, such as mean and variance). Because θ depends on z , but z is hidden, we cannot directly apply maximum likelihood estimation to solve the argmax problem. Let us also define $q(z)$ as any arbitrary distribution of the latent variable z . As such, we have:

$$p(z/y, \theta) = \frac{p(y|z, \theta)p(z|\theta)}{p(y|\theta)} \quad \dots(1)$$

$$p(y, \theta) = \frac{p(y|z, \theta)p(z|\theta)}{p(z|y, \theta)} \quad \dots(2)$$

$$p(y, \theta) = \frac{p(y|z, \theta)p(z|\theta)}{q(z)} \frac{q(z)}{p(z|y, \theta)} \quad \dots(3)$$

It should be clear that Eq 1 is derived from Bayes' rule. Rearranging terms from the left hand side and right hand side of Eq 1, we can arrive at Eq 2. Then, artificially multiply and divide by $q(z)$, we get Eq3.

$$\log p(y|\theta) = \log \frac{p(y|z, \theta)p(z|\theta)}{q(z)} + \log \frac{q(z)}{p(z|y, \theta)} \quad \dots(4)$$

Taking logarithms on both sides of Eq 3 yields Eq 4. Computing the expectation of both sides with respect to $q(z)$, we get:

$$E(\log p(y|\theta)) = \int q(z) \log \frac{p(y|z, \theta)p(z|\theta)}{q(z)} + \int q(z) \log \frac{q(z)}{p(z|y, \theta)} dz \quad \dots(5)$$

Now, we need the help of Jensen's inequality. Recall that for any strictly convex function, we have $E(f(x)) \geq f(E(x))$. The inequality reverses for strictly concave functions. Going back to Eq 5, since the log function is indeed a concave function (i.e. 2nd order derivative is $-1/x^2$, which < 0), now we can derive that log likelihood of the y given θ has a lower bound as outlined in Eq 5. Moreover, the second term of Equation 5 is essentially the KL divergence between $q(z)$ and $p(z|y, \theta)$, and it is always non-negative. Therefore, the lower bound can be reduced to just the first term, which we denote as $F(q(z), \theta)$.

You may ask, to maximize likelihood, why not just take the first-order derivative, set it to 0 and solve for the parameter in any of the equation 1–4? If you think about it, the derivative part is actually not an easy task. However, $F(q(z), \theta)$ in Eq 5 is more manageable to work with, since it is sum of logs.

The E step starts with a fixed $\theta(t)$, and attempts to maximize the lower bound (LB) function $F(q(z), \theta)$ with respect to $q(z)$. Intuitively, this happens when the LB function meets the objective likelihood function. Mathematically, this is the case because the likelihood function is independent of $q(z)$, and so maximizing lower bound is equivalent to minimizing the KL divergence of $q(z)$ and $p(z|y, \theta(t))$. Therefore, the E step gives $q(z) = p(z|y, \theta(t))$.

On the other hand, the M step tries to maximize the lower bound function with respect to $\theta(t)$ based on the fixed $q(z)$. Recall from Eq 5 that

$$F(q(z), \theta) = \int q(z) \log \frac{p(y|z, \theta)p(z|\theta)}{q(z)} dz$$

We can ignore $q(z)$ in the denominator since it is independent of θ . Therefore, the M step becomes the following argmax problem at time t :

$$q_t = \arg \max \int q_t(z) \log(p(y|z, \theta_{t-1})p(z|\theta_{t-1})) dz \quad \dots(6)$$

Repeatedly executing the E step and the M step, it's expected to reach local maximum of the likelihood function, as indicated by the above chart.

5.1.2 Gmms

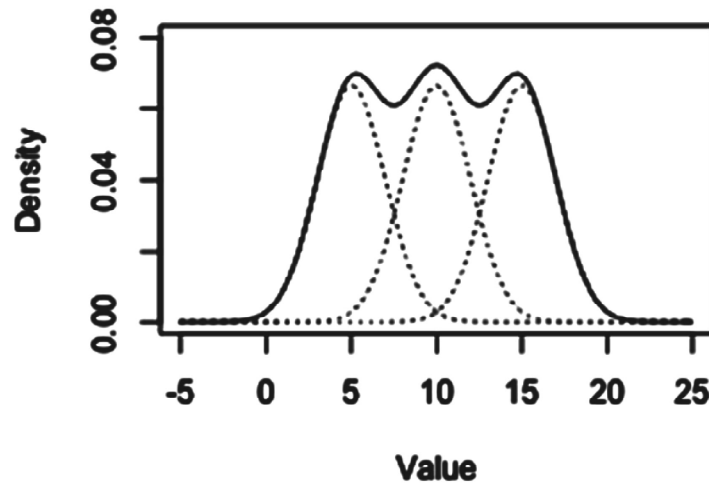
Q3. Write about Gaussian mixture models.

Ans :

Gaussian Mixture Model

Gaussian mixture models are a probabilistic model for representing normally distributed subpopulations within an overall population. Mixture models in general don't require knowing

which subpopulation a data point belongs to, allowing the model to learn the subpopulations automatically. Since subpopulation assignment is not known, this constitutes a form of unsupervised learning.



For example, in modeling human height data, height is typically modeled as a normal distribution for each gender with a mean of approximately 5'10" for males and 5'5" for females. Given only the height data and not the gender assignments for each data point, the distribution of all heights would follow the sum of two scaled (different variance) and shifted (different mean) normal distributions. A model making this assumption is an example of a Gaussian mixture model (GMM), though in general a GMM may have more than two components. Estimating the parameters of the individual normal distribution components is a canonical problem in modeling data with GMMs.

GMMs have been used for feature extraction from speech data, and have also been used extensively in object tracking of multiple objects, where the number of mixture components and their means predict object locations at each frame in a video sequence.

The Model

A Gaussian mixture model is parameterized by two types of values, the mixture component weights and the component means and variances/covariances. For a Gaussian mixture model with K components, the k^{th} component has a mean of μ_k and variance of σ_k for the univariate case and a mean of μ_k and covariance matrix of Σ_k for the multivariate case. The mixture component weights are defined as ϕ_k for component C_k , with the constraint that $\sum_{i=1}^K \phi_i = 1$ so that the total probability distribution normalizes to 1. If the component weights aren't learned, they can be viewed as an a-priori distribution over components such that $p(x \text{ generated by component } C_k) = \phi_k$. If they are instead learned, they are the a-posteriori estimates of the component probabilities given the data.

One-dimensional Model

$$p(x) = \sum_{i=1}^K \phi_i N(x | \mu_i, \sigma_i)$$

$$N(x | \mu_i, \sigma_i) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp\left(-\frac{(x - \mu_i)^2}{2\sigma_i^2}\right)$$

Multi-dimensional Model

$$p(\bar{x}) = \sum_{i=1}^K \phi_i N(\bar{x} | \bar{\mu}_i, \Sigma_i)$$

$$N(\bar{x} | \bar{\mu}_i, \Sigma_i) = \frac{1}{\sqrt{(2\pi)^K |\Sigma_i|}} \exp \left(-\frac{1}{2} (\bar{x} - \bar{\mu}_i)^T \Sigma_i^{-1} (\bar{x} - \bar{\mu}_i) \right)$$

$$\sum_{i=1}^K \phi_i = 1$$

Learning the Model

If the number of components K is known, expectation maximization is the technique most commonly used to estimate the mixture model's parameters. In frequentist probability theory, models are typically learned by using maximum likelihood estimation techniques, which seek to maximize the probability, or likelihood, of the observed data given the model parameters. Unfortunately, finding the maximum likelihood solution for mixture models by differentiating the log likelihood and solving for θ is usually analytically impossible.

Expectation maximization (EM) is a numerical technique for maximum likelihood estimation, and is usually used when closed form expressions for updating the model parameters can be calculated (which will be shown below). Expectation maximization is an iterative algorithm and has the convenient property that the maximum likelihood of the data strictly increases with each subsequent iteration, meaning it is guaranteed to approach a local maximum or saddle point.

Q4. Explain EM algorithms for Gaussian Mixture Models.

Ans :

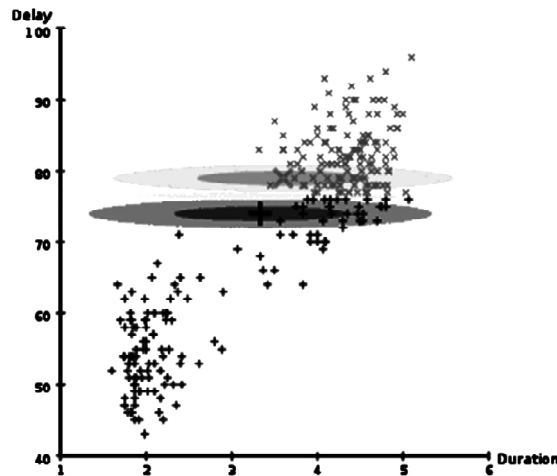
EM for Gaussian Mixture Models

Expectation maximization for mixture models consists of two steps.

The first step, known as the expectation step or E step, consists of calculating the expectation of the component assignments C_k for each data point in $x_i \in X$ given the model parameters ϕ_k , μ_k , and σ_k .

The second step is known as the maximization step or M step, which consists of maximizing the expectations calculated in the E step with respect to the model parameters. This step consists of updating the values ϕ_k , μ_k , and σ_k .

The entire iterative process repeats until the algorithm converges, giving a maximum likelihood estimate. Intuitively, the algorithm works because knowing the component assignment C_k for each x_i makes solving for ϕ_k , μ_k , and σ_k easy, while knowing ϕ_k , μ_k , and σ_k makes inferring $p(C_k | x_i)$ easy. The expectation step corresponds to the latter case while the maximization step corresponds to the former. Thus, by alternating between which values are assumed fixed, or known, maximum likelihood estimates of the non-fixed values can be calculated in an efficient manner.



The EM algorithm updating the parameters of a two-component bivariate Gaussian mixture model.

Algorithm for Univariate Gaussian Mixture Models

The expectation maximization algorithm for Gaussian mixture models starts with an initialization step, which assigns model parameters to reasonable values based on the data. Then, the model iterates over the expectation (E) and maximization (M) steps until the parameters' estimates converge, i.e. for all parameters θ_t at iteration t , $|\theta_t - \theta_{t-1}| \in I$ for some user-defined tolerance I . A graphic of the EM algorithm in action for a two-component, bivariate Gaussian mixture model is displayed on the right.

The EM algorithm for a univariate Gaussian mixture model with K components is described below. A variable denoted $\hat{\theta}$ denotes an estimate for the value θ . All equations can be derived algebraically by solving for each parameter as outlined in the section above titled EM for Gaussian Mixture Models.

Initialization Step:

Randomly assign samples without replacement from the dataset $X = \{x_1, \dots, x_N\}$ to the component mean estimates $\hat{\mu}_1, \dots, \hat{\mu}_K$. E.g. for $K = 3$ and $N = 100$, set $\hat{\mu}_1 = x_{45}$, $\hat{\mu}_2 = x_{32}$, $\hat{\mu}_3 = x_{10}$

Set all component variance estimates to the sample variance $\hat{\sigma}_1^2, \dots, \hat{\sigma}_K^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2$ where \bar{x} is the sample mean $\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i$.

Set all component distribution prior estimates to the uniform distribution $\hat{\phi}_1, \dots, \hat{\phi}_K = \frac{1}{K}$.

Expectation (E) Step:

Calculate $\forall i, k$

$$\hat{\gamma}_{ik} = \frac{\hat{\phi}_k N(x_i | \hat{\mu}_k, \hat{\sigma}_k)}{\sum_{j=1}^K \hat{\phi}_j N(x_i | \hat{\mu}_j, \hat{\sigma}_j)}$$

where $\hat{\gamma}_{ik}$ is the probability that x_i is generated by component C_k . Thus, $\hat{\gamma}_{ik} = p(C_k | x_i, \hat{\phi}, \hat{\mu}, \hat{\sigma})$

Maximization (M) Step:

Using the $\hat{\gamma}_{ik}$ calculated in the expectation step, calculate the following in that order $\forall k$:

- $\hat{\phi}_k = \frac{\sum_{i=1}^N \hat{\gamma}_{ik}}{N}$
- $\hat{\mu}_k = \frac{\sum_{i=1}^N \hat{\gamma}_{ik} x_i}{\sum_{i=1}^N \hat{\gamma}_{ik}}$
- $\hat{\sigma}_k^2 = \frac{\sum_{i=1}^N \hat{\gamma}_{ik} (x_i - \hat{\mu}_k)^2}{\sum_{i=1}^N \hat{\gamma}_{ik}}$

When the number of components K is not known a priori, it is typical to guess the number of components and fit that model to the data using the EM algorithm. This is done for many different values of K . Usually, the model with the best trade-off between fit and number of components (simpler models have fewer components) is kept.

Unsupervised Learning

Once the EM algorithm has run to completion, the fitted model can be used to perform various forms of inference. The two most common forms of inference done on GMMs are density estimation and clustering.

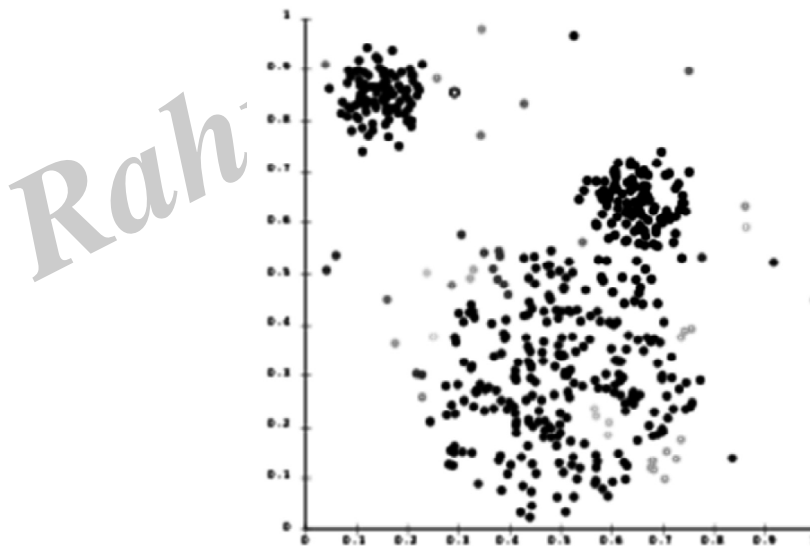


Fig: Clustering using a Gaussian mixture model. Each color represents a different cluster according to the model

Density Estimation

Since the GMM is completely determined by the parameters of its individual components, a fitted GMM can give an estimate of the probabilities of both in-sample and out-of-sample data points, known as density estimation. Furthermore, since numerically sampling from an individual Gaussian distribution is possible, one can easily sample from a GMM to create synthetic datasets.

Sampling from a GMM consists of the following steps:

1. Sample the Gaussian component according to the distribution defined by $p(C_s) = \phi_s$.
2. Sample x from the distribution for component C_s , according to the distribution defined by $N(x | \mu_s, \sigma_s)$.

Clustering

Using Bayes' theorem and the estimated model parameters, one can also estimate the posteriori component assignment probability. Knowing that a data point is likely from one component distribution versus another provides a way to learn clusters, where cluster assignment is determined by the most likely component assignment. Clustering has many uses in machine learning, ranging from tissue differentiation in medical imaging to customer segmentation in market research.

Given a univariate model's parameters, the probability that a data point x belongs to component C_i is calculated using Bayes' theorem:

$$p(C_i | x) = \frac{p(x, C_i)}{p(x)} = \frac{p(C_i)p(x | C_i)}{\sum_j^K p(C_j)p(x | C_j)} = \frac{\phi_i N(x | \mu_i, \sigma_i)}{\sum_{j=1}^K \phi_j N(x | \mu_j, \sigma_j)}$$

5.2 LEARNING THEORY

Q5. What Is Machine Learning? What are the various components in machine learning architecture?

Ans :

(Imp.)

Machine learning is programming computers to optimize a performance criterion using example data or past experience. We have a model defined up to some parameters, and learning is the execution of a computer program to optimize the parameters of the model using the training data or past experience. The model may be predictive to make predictions in the future, or descriptive to gain knowledge from data, or both.

Basic components of learning process The learning process, whether by a human or a machine, can be divided into four components, namely, data storage, abstraction, generalization and evaluation.

Figure 1.1 illustrates the various components and the steps involved in the learning process.

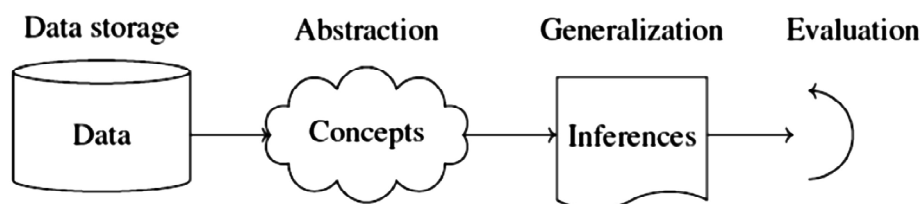


Fig.: Component of learning process

1. Data storage Facilities for storing and retrieving huge amounts of data are an important component of the learning process. Humans and computers alike utilize data storage as a foundation for advanced reasoning.

In a human being, the data is stored in the brain and data is retrieved using electrochemical signals.

Computers use hard disk drives, flash memory, random access memory and similar devices to store data and use cables and other technology to retrieve data.

2. **Abstraction** The second component of the learning process is known as abstraction. Abstraction is the process of extracting knowledge about stored data. This involves creating general concepts about the data as a whole. The creation of knowledge involves application of known models and creation of new models. The process of fitting a model to a dataset is known as training. When the model has been trained, the data is transformed into an abstract form that summarizes the original information.
3. **Generalization** The third component of the learning process is known as generalisation. The term generalization describes the process of turning the knowledge about stored data into a form that can be utilized for future action. These actions are to be carried out on tasks that are similar, but not identical, to those what have been seen before. In generalization, the goal is to discover those properties of the data that will be most relevant to future tasks.
4. **Evaluation** Evaluation is the last component of the learning process. It is the process of giving feedback to the user to measure the utility of the learned knowledge. This feedback is then utilised to effect improvements in the whole learning process

Applications of machine learning

Application of machine learning methods to large databases is called data mining. In data mining, a large volume of data is processed to construct a simple model with valuable use, for example, having high predictive accuracy. The following is a list of some of the typical applications of machine learning.

1. In retail business, machine learning is used to study consumer behaviour.
2. In finance, banks analyze their past data to build models to use in credit applications, fraud detection, and the stock market.
3. In manufacturing, learning models are used for optimization, control, and troubleshooting.
 - Using Probability to classify the instance space. (Probabilistic models)
 - Grouping and Grading

Q6. Explain various learning models in machine learning

Ans :

1. Logical models

Logical models use a logical expression to divide the instance space into segments and hence construct grouping models. A logical expression is an expression that returns a Boolean value, i.e., a True or False outcome. Once the data is grouped using a logical expression, the data is divided into homogeneous groupings for the problem we are trying to solve. For example, for a classification problem, all the instances in the group belong to one class.

There are mainly two kinds of logical models:

Tree models and Rule models.

Rule models consist of a collection of implications or IF-THEN rules. For tree-based models, the 'if-part' defines a segment and the 'then-part' defines the behaviour of the model for this segment. Rule models follow the same reasoning.

Logical models and Concept learning

To understand logical models further, we need to understand the idea of Concept Learning.

Concept Learning involves learning logical expressions or concepts from examples. The idea of Concept Learning fits in well with the idea of Machine learning, i.e., inferring a general function from

specific training examples. Concept learning forms the basis of both tree-based and rule-based models. More formally, Concept Learning involves acquiring the definition of a general category from a given set of positive and negative training examples of the category.

A Formal Definition for Concept Learning is

“The inferring of a Boolean-valued function from training examples of its input and output.” In concept learning, we only learn a description for the positive class and label everything that doesn't satisfy that description as negative.

The following example explains this idea in more detail.

A Concept Learning Task - Enjoy Sport

Training Examples

Example	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
1	Sunny	Warm	Normal	Strong	Warm	Same	YES
2	Sunny	Warm	High	Strong	Warm	Same	YES
3	Rainy	Cold	High	Strong	Warm	Change	NO
4	Sunny	Warm	High	Strong	Warm	Change	YES

ATTRIBUTES

CONCEPT

Concept Learning

Task called “Enjoy Sport” as shown above is defined by a set of data from some example days. Each data is described by six attributes. The task is to learn to predict the value of Enjoy Sport for an arbitrary day based on the values of its attribute values. The problem can be represented by a series of hypotheses. Each hypothesis is described by a conjunction of constraints on the attributes. The training data represents a set of positive and negative examples of the target function. In the example above, each hypothesis is a vector of six constraints, specifying the values of the six attributes – Sky, AirTemp, Humidity, Wind, Water, and Forecast. The training phase involves learning the set of days (as a conjunction of attributes) for which Enjoy Sport = yes. Thus, the problem can be formulated as:

- Given instances X which represent a set of all possible days, each described by the attributes:
 - Sky – (values: Sunny, Cloudy, Rainy),
 - AirTemp – (values: Warm, Cold),
 - Humidity – (values: Normal, High),
 - Wind – (values: Strong, Weak),
 - Water – (values: Warm, Cold),
 - Forecast – (values: Same, Change).

Try to identify a function that can predict the target variable Enjoy Sport as yes/no, i.e., 1 or 0.

2. Geometric models

In Geometric models, features could be described as points in two dimensions (x- and y-axis) or a three-dimensional space (x, y, and z).

Even when features are not intrinsically geometric, they could be modelled in a geometric manner (for example, temperature as a function of time can be modelled in two axes).

In geometric models, there are two ways we could impose similarity.

- We could use geometric concepts like lines or planes to segment (classify) the instance space. These are called Linear models.
- Alternatively, we can use the geometric notion of distance to represent similarity. In this case, if two points are close together, they have similar values for features and thus can be classed as similar. We call such models as Distance-based models.

3. Probabilistic models

The third family of machine learning algorithms is the probabilistic models. We have seen before that the k-nearest neighbour algorithm uses the idea of distance (e.g., Euclidian distance) to classify entities, and logical models use a logical expression to partition the instance space.

Probabilistic models see features and target variables as random variables. The process of modelling represents and manipulates the level of uncertainty with respect to these variables.

There are two types of probabilistic models:

- Predictive and Generative.

Predictive probability models use the idea of a conditional probability distribution $P(Y | X)$ from which Y can be predicted from X. Generative models estimate the joint distribution $P(Y, X)$. Once we know the joint distribution for the generative models, we can derive any conditional or marginal distribution involving the same variables.

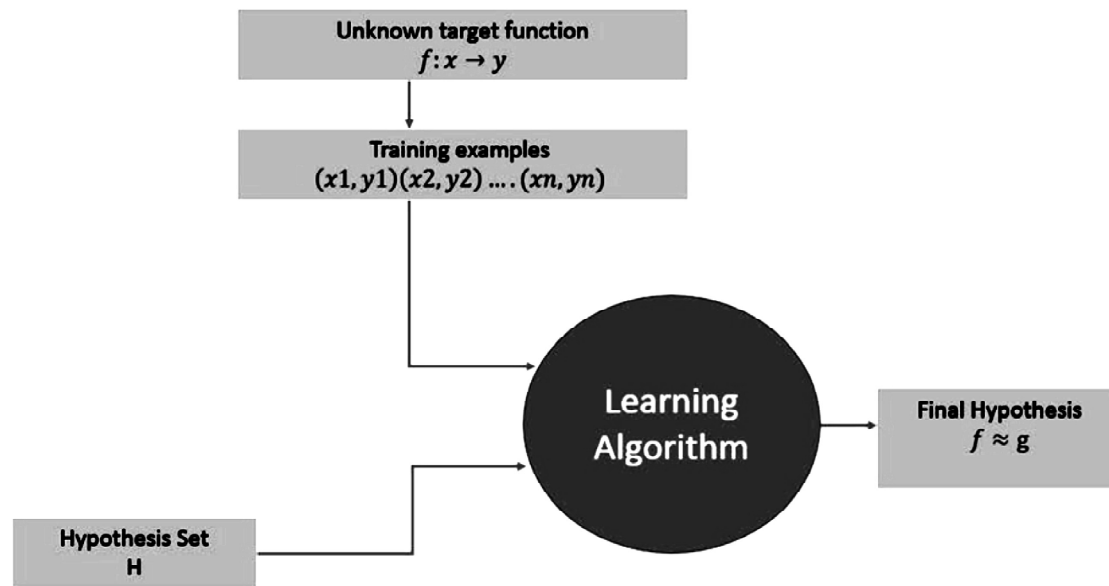
Thus, the generative model is capable of creating new data points and their labels, knowing the joint probability distribution. The joint distribution looks for a relationship between two variables. Once this relationship is inferred, it is possible to infer new data points.

Q7. How to design the learning model ? Explain

Ans :

Designing a Learning System

For any learning system, we must be knowing the three elements – T (Task), P (Performance Measure), and E (Training Experience). At a high level, the process of learning system looks as below.



The learning process starts with task T , performance measure P and training experience E and objective are to find an unknown target function. The target function is an exact knowledge to be learned from the training experience and its unknown.

For example, in a case of credit approval, the learning system will have customer application records as experience and task would be to classify whether the given customer application is eligible for a loan.

So in this case, the training examples can be represented as $(x_1, y_1)(x_2, y_2) \dots (x_n, y_n)$ where X represents customer application details and y represents the status of credit approval.

So the target function to be learned in the credit approval learning system is a mapping function $f: X \rightarrow y$.

This function represents the exact knowledge defining the relationship between input variable X and output variable y .

Design of a learning system

The design choices will be to decide the following key components:

1. Type of training experience
2. Choosing the Target Function
3. Choosing a representation for the Target Function
4. Choosing an approximation algorithm for the Target Function
5. The final Design

We will look into the game - checkers learning problem and apply the above design choices. For a checkers learning problem, the three elements will be,

1. **Task T :** To play checkers
2. **Performance measure P :** Total percent of the game won in the tournament.
3. **Training experience E :** A set of games played against itself

Type of training experience

During the design of the checker's learning system, the type of training experience available for a learning system will have a significant effect on the success or failure of the learning.

1. **Direct or Indirect training experience:** In the case of direct training experience, an individual board states and correct move for each board state are given.

In case of indirect training experience, the move sequences for a game and the final result (win, loss or draw) are given for a number of games. How to assign credit or blame to individual moves is the credit assignment problem.

2. **Teacher or Not :** Supervised – The training experience will be labeled, which means, all the board states will be labeled with the correct move. So the learning takes place in the presence of a supervisor or a teacher.

Unsupervised — The training experience will be unlabeled, which means, all the board states will not have the moves. So the learner generates random games and plays against itself with no supervisor or teacher involvement. Semi-supervised — Learner generates game states and asks the teacher for help in finding the correct move if the board state is confusing.

3. **Is the training experience good:** Do the training examples represent the distribution of examples over which the final system performance will be measured? Performance is best when training examples and test examples are from the same/a similar distribution.

The checker player learns by playing against oneself. Its experience is indirect. It may not encounter moves that are common in human expert play. Once the proper training experience is available, the next design step will be choosing the Target Function.

Choosing the Target Function

When you are playing the checkers game, at any moment of time, you make a decision on choosing the best move from different possibilities. You think and apply the learning that you have gained from the experience. Here the learning is, for a specific board, you move a checker such that your board state tends towards the winning situation. Now the same learning has to be defined in terms of the target function.

Here there are 2 considerations – direct and indirect experience.

Choosing a representation for the Target Function

Now that we have specified the ideal target function V , we must choose a representation that the learning program will use to describe the function \hat{V} that it will learn. As with earlier design choices, we again have many options. We could, for example, allow the program to represent using a large table with a distinct entry specifying the value for each distinct board state. Or we could allow it to represent using a collection of rules that match against features of the board state, or a quadratic polynomial function of predefined board features, or an artificial neural network. In general, this choice of representation involves a crucial tradeoff. On one hand, we wish to pick a very expressive representation to allow representing as close an approximation as possible to the ideal target function V .

Choosing an approximation algorithm for the Target Function

Generating training data — To train our learning program, we need a set of training data, each describing a specific board state b and the training value $V_{\text{train}}(b)$ for b . Each training example is an ordered pair $\langle b, V_{\text{train}}(b) \rangle$

For example, a training example may be $\langle (x_1 = 3, x_2 = 0, x_3 = 1, x_4 = 0, x_5 = 0, x_6 = 0), +100 \rangle$.

This is an example where black has won the game since $x_2 = 0$ or red has no remaining pieces. However, such clean values of $V_{\text{train}}(b)$ can be obtained only for board value b that are clear win, loss or draw.

In above case, assigning a training value $V_{\text{train}}(b)$ for the specific boards b that are clean win, loss or draw is direct as they are direct training experience. But in the case of indirect training experience, assigning a training value $V_{\text{train}}(b)$ for the intermediate boards is difficult. In such case, the training values are updated using temporal difference learning.

Temporal difference (TD) learning is a concept central to reinforcement learning, in which learning happens through the iterative correction of your estimated returns towards a more accurate target return.

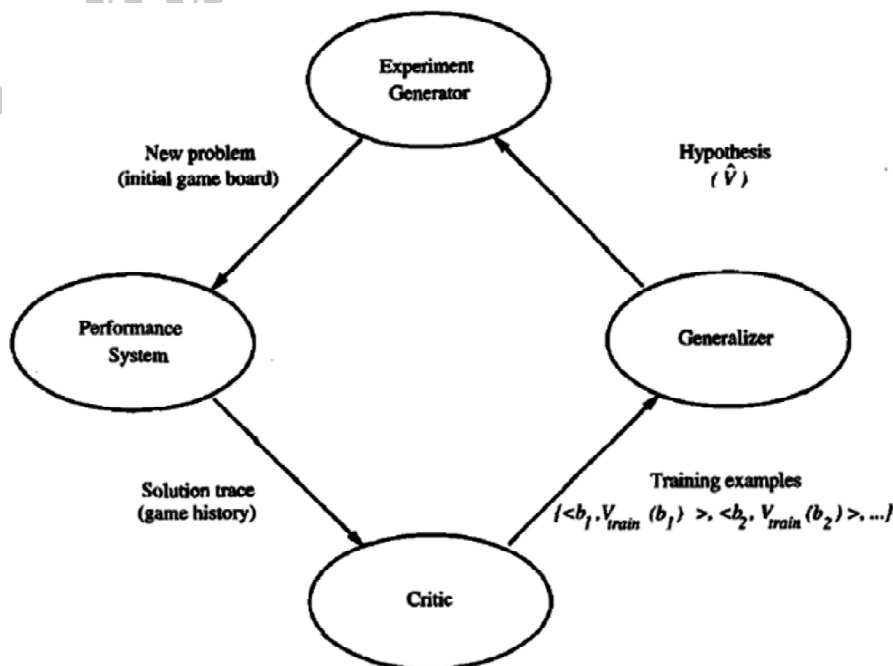
Let $\text{Successor}(b)$ denotes the next board state following b for which it is again the program's turn to move. \hat{V} is the learner's current approximation to V . Using these information, assign the training value of $V_{\text{train}}(b)$ for any intermediate board state b as below :

$$V_{\text{train}}(b) \leftarrow \hat{V}(\text{Successor}(b))$$

Final Design for Checkers Learning system

The final design of our checkers learning system can be naturally described by four distinct program modules that represent the central components in many learning systems.

1. **The performance System:** Takes a new board as input and outputs a trace of the game it played against itself.
2. **The Critic:** Takes the trace of a game as an input and outputs a set of training examples of the target function.
3. **The Generalizer:** Takes training examples as input and outputs a hypothesis that estimates the target function. Good generalization to new cases is crucial.
4. **The Experiment Generator:** Takes the current hypothesis (currently learned function) as input and outputs a new problem (an initial board state) for the performance system to explore.



Q8. What are Version spaces ? How can we use version spaces in representation of knowledge. Explain with example.

Ans :

(Imp.)

Version Spaces

Version space is a hierarchical representation of knowledge that enables you to keep track of all the useful information supplied by a sequence of learning examples without remembering any of the examples.

The version space method is a concept learning process accomplished by managing multiple models within a version space.

Version Space Characteristics

Tentative heuristics are represented using version spaces.

A version space represents all the alternative plausible descriptions of a heuristic.

A plausible description is one that is applicable to all known positive examples and no known negative example.

A version space description consists of two complementary trees:

1. One that contains nodes connected to overly general models, and
2. One that contains nodes connected to overly specific models.

Node values/attributes are discrete.

Fundamental Assumptions

1. The data is correct; there are no erroneous instances.
2. A correct description is a conjunction of some of the attributes with values.

Diagrammatical Guidelines

There is a generalization tree and a specialization tree.

Each node is connected to a model.

Nodes in the generalization tree are connected to a model that matches everything in its subtree.

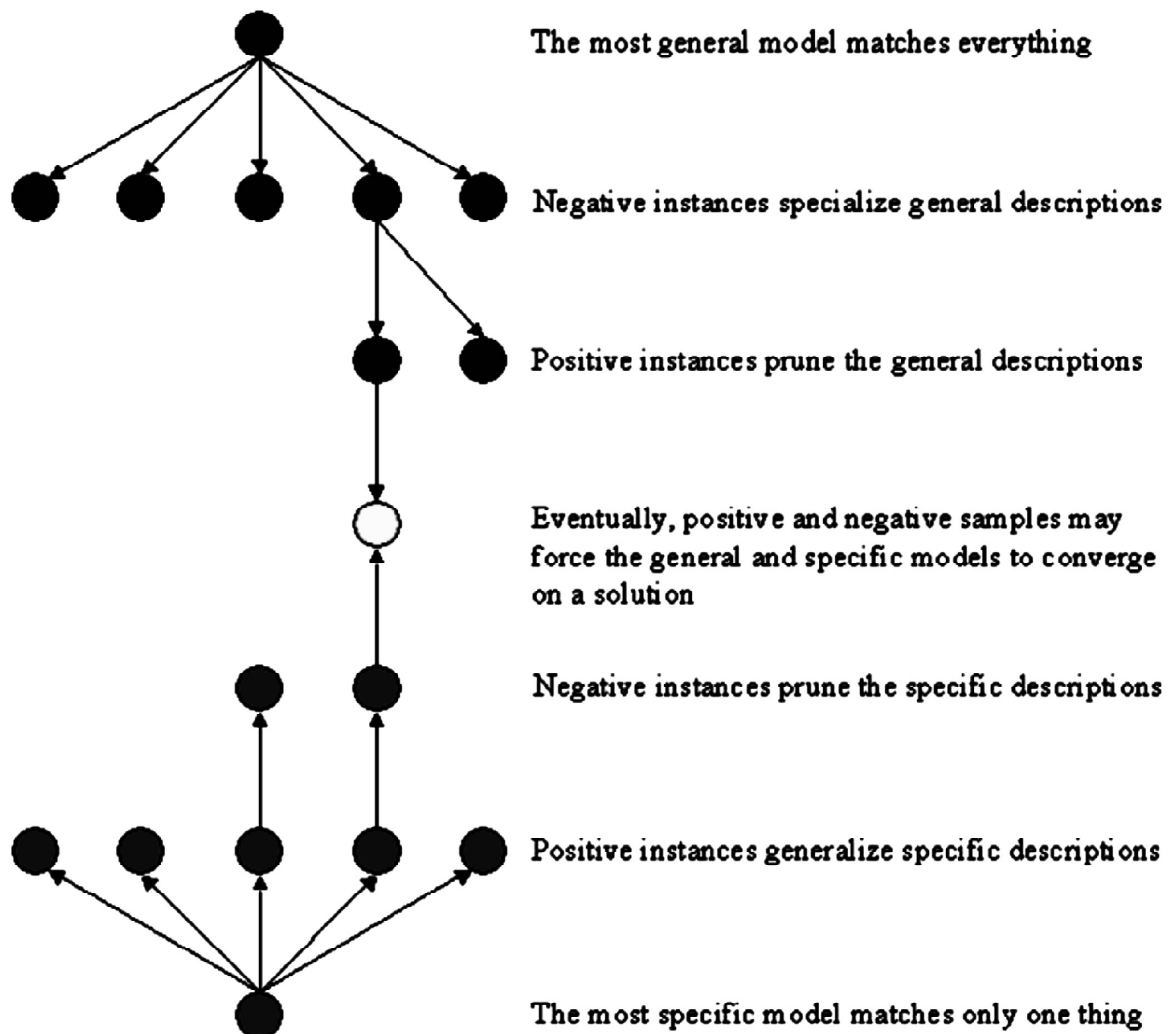
Nodes in the specialization tree are connected to a model that matches only one thing in its subtree.

Links between nodes and their models denote

- generalization relations in a generalization tree, and
- specialization relations in a specialization tree.

Diagram of a Version Space

In the diagram below, the specialization tree is colored red, and the generalization tree is colored green.



Generalization and Specialization Leads to Version Space Convergence

The key idea in version space learning is that specialization of the general models and generalization of the specific models may ultimately lead to just one correct model that matches all observed positive examples and does not match any negative examples.

That is, each time a negative example is used to specialize the general models, those specific models that match the negative example are eliminated and each time a positive example is used to generalize the specific models, those general models that fail to match the positive example are eliminated. Eventually, the positive and negative examples may be such that only one general model and one identical specific model survive.

Version Space Method Learning Algorithm: Candidate-Elimination

The version space method handles positive and negative examples symmetrically.

Given:

- A representation language.
- A set of positive and negative examples expressed in that language.

Compute a concept description that is consistent with all the positive examples and none of the negative examples.

Method:

- Initialize G, the set of maximally general hypotheses, to contain one element: the null description (all features are variables).
- Initialize S, the set of maximally specific hypotheses, to contain one element: the first positive example.
- Accept a new training example.

If the example is positive:

1. Generalize all the specific models to match the positive example, but ensure the following:
 - The new specific models involve minimal changes.
 - Each new specific model is a specialization of some general model.
 - No new specific model is a generalization of some other specific model.
2. Prune away all the general models that fail to match the positive example.

If the example is negative:

1. Specialize all general models to prevent match with the negative example, but ensure the following:
 - The new general models involve minimal changes.
 - Each new general model is a generalization of some specific model.
 - No new general model is a specialization of some other general model.
2. Prune away all the specific models that match the negative example.
 - If S and G are both singleton sets, then:
 - if they are identical, output their value and halt.
 - If they are different, the training cases were inconsistent. Output this result and halt.
 - else continue accepting new training examples.

The algorithm stops when:

1. It runs out of data.
2. The number of hypotheses remaining is:
 - 0 - no consistent description for the data in the language.
 - 1 - answer (version space converges).
 - 2⁺ - all descriptions in the language are implicitly included.

Comments on the Version Space Method

The version space method is still a trial and error method.

The program does not base its choice of examples, or its learned heuristics, on an analysis of what works or why it works, but rather on the simple assumption that what works will probably work again.

Unlike the decision tree ID3 algorithm,

- Candidate-elimination searches an incomplete set of hypotheses (ie. only a subset of the potentially teachable concepts are included in the hypothesis space).
- Candidate-elimination finds every hypothesis that is consistent with the training data, meaning it searches the hypothesis space completely.
- Candidate-elimination's inductive bias is a consequence of how well it can represent the subset of possible hypotheses it will search. In other words, the bias is a product of its search space.
- No additional bias is introduced through Candidate-elimination's search strategy.

Advantages of the version space method:

- Can describe all the possible hypotheses in the language consistent with the data.
- Fast (close to linear).

Disadvantages of the version space method:

- Inconsistent data (noise) may cause the target concept to be pruned.
- Learning disjunctive concepts is challenging.

Example Problem**Problem 1:**

Learning the concept of "Japanese Economy Car"

Features: (Country of Origin, Manufacturer, Color, Decade, Type)

<i>Origin</i>	<i>Manufacturer</i>	<i>Color</i>	<i>Decade</i>	<i>Type</i>	<i>Example Type</i>
Japan	Honda	Blue	1980	Economy	Positive
Japan	Toyota	Green	1970	Sports	Negative
Japan	Toyota	Blue	1990	Economy	Positive
USA	Chrysler	Red	1980	Economy	Negative
Japan	Honda	White	1980	Economy	Positive

Sol.:

1. Positive Example: (Japan, Honda, Blue, 1980, Economy)

Initialize G to a singleton set that includes everything. Initialize S to a singleton set that includes the first positive example.	$G = \{ (?, ?, ?, ?, ?) \}$ $S = \{ (Japan, Honda, Blue, 1980, Economy) \}$
--	--

●(?, ?, ?, ?, ?)

(Japan, Honda, Blue, 1980, Economy)●

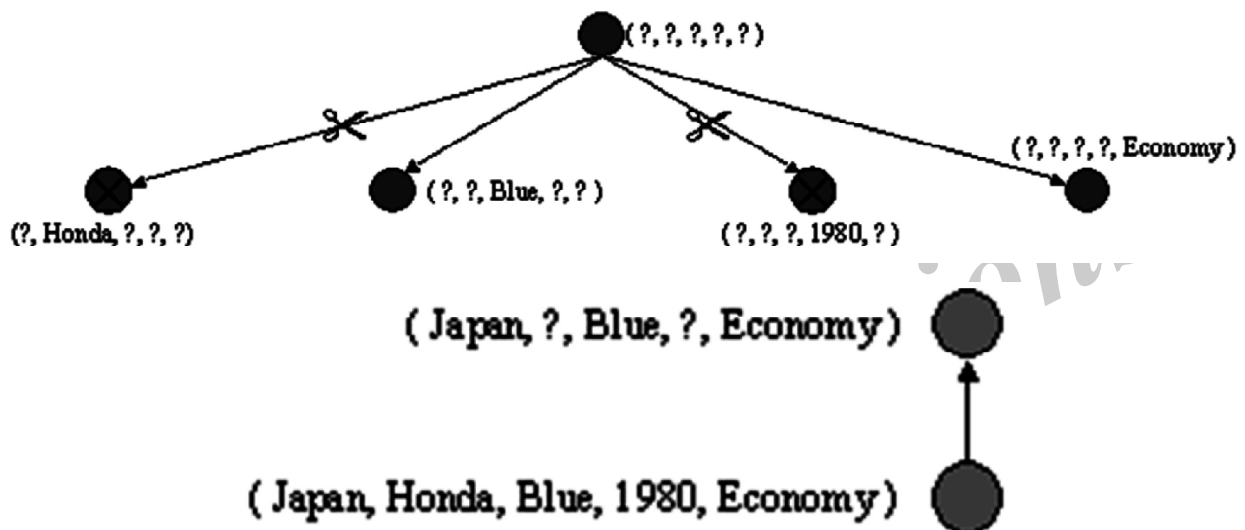
These models represent the most general and the most specific heuristics one might learn.

The actual heuristic to be learned, "Japanese Economy Car", probably lies between them somewhere within the version space.

2. **Negative Example:** (Japan, Toyota, Green, 1970, Sports)

Specialize G to exclude the negative example.

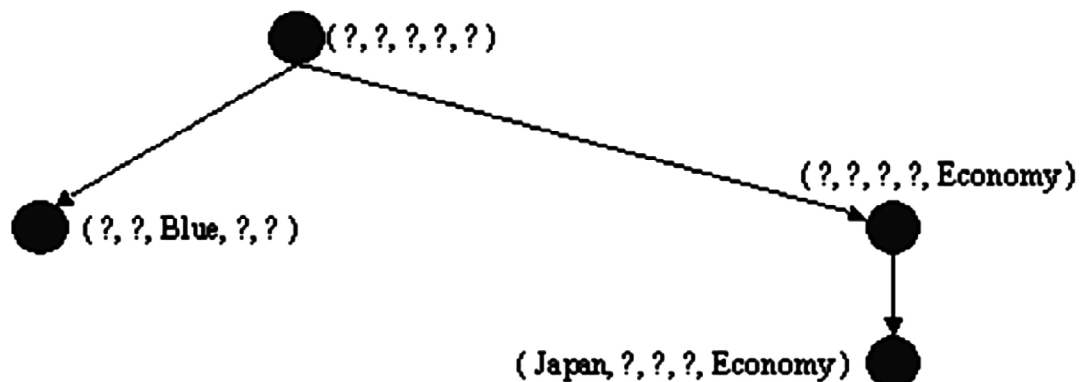
G =	{ (?, Honda, ?, ?, ?), (?, ?, Blue, ?, ?), (?, ?, ?, 1980, ?), (?, ?, ?, ?, Economy) }
S =	{ (Japan, Honda, Blue, 1980, Economy) }

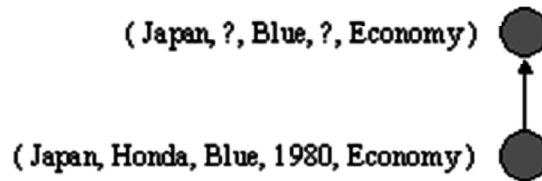


4. **Negative Example:** (USA, Chrysler, Red, 1980, Economy)

Specialize G to exclude the negative example (but stay consistent with S)

G =	{ (?, ?, Blue, ?, ?), (Japan, ?, ?, ?, Economy) }
S =	{ (Japan, ?, Blue, ?, Economy) }





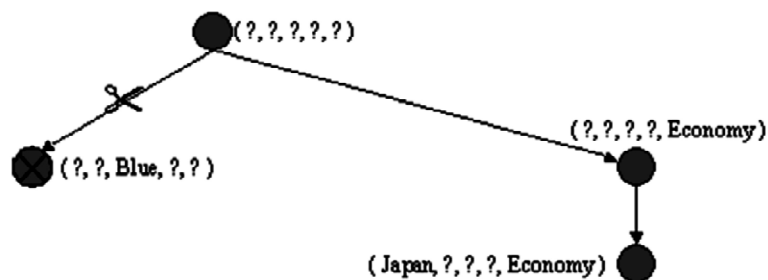
5. **Positive Example:** (Japan, Honda, White, 1980, Economy)

Prune G to exclude descriptions inconsistent with positive example.

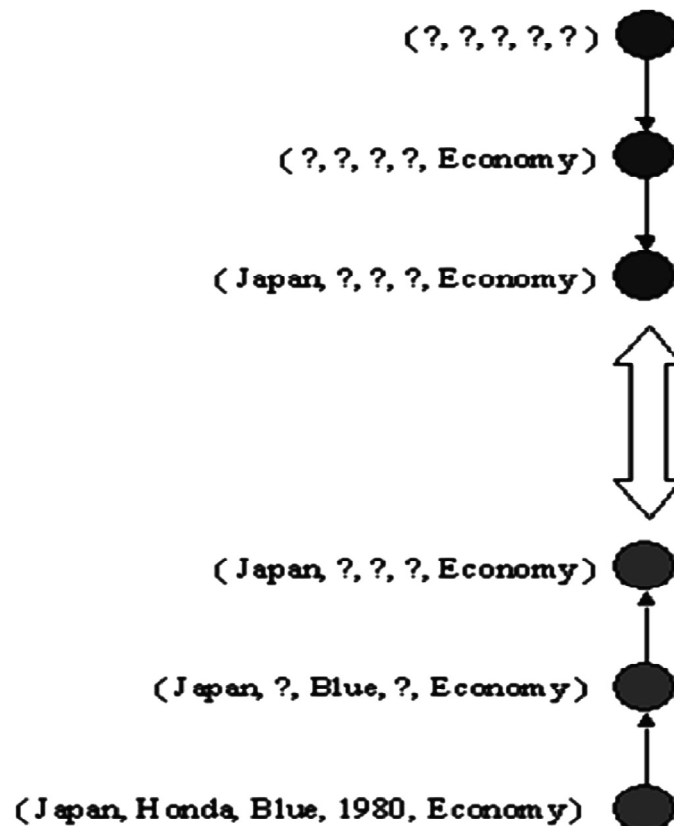
Generalize S to include positive example.

$$G = \{ (Japan, ?, ?, ?, Economy) \}$$

$$S = \{ (Japan, ?, ?, ?, Economy) \}$$



G and S are singleton sets and $S = G$. Converged. No more data, so algorithm stops.



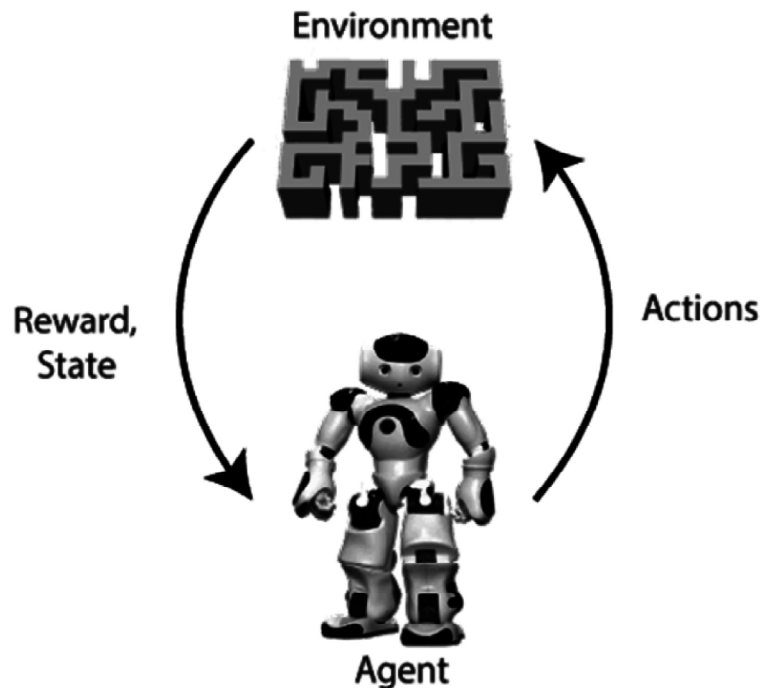
5.3 INTRODUCTION TO REINFORCEMENT LEARNING GRAPHICAL MODELS

Q9. Give the brief introduction of Reinforcement Learning and working of Reinforcement Learning.

Ans :

(Imp.)

- Reinforcement Learning is a feedback-based Machine learning technique in which an agent learns to behave in an environment by performing the actions and seeing the results of actions. For each good action, the agent gets positive feedback, and for each bad action, the agent gets negative feedback or penalty.
- In Reinforcement Learning, the agent learns automatically using feedbacks without any labeled data, unlike supervised learning.
- Since there is no labeled data, so the agent is bound to learn by its experience only.
- RL solves a specific type of problem where decision making is sequential, and the goal is long-term, such as game-playing, robotics, etc.
- The agent interacts with the environment and explores it by itself. The primary goal of an agent in reinforcement learning is to improve the performance by getting the maximum positive rewards.
- The agent learns with the process of hit and trial, and based on the experience, it learns to perform the task in a better way. Hence, we can say that "Reinforcement learning is a type of machine learning method where an intelligent agent (computer program) interacts with the environment and learns to act within that." How a Robotic dog learns the movement of his arms is an example of Reinforcement learning.
- It is a core part of Artificial intelligence, and all AI agent works on the concept of reinforcement learning. Here we do not need to pre-program the agent, as it learns from its own experience without any human intervention.
- Example: Suppose there is an AI agent present within a maze environment, and his goal is to find the diamond. The agent interacts with the environment by performing some actions, and based on those actions, the state of the agent gets changed, and it also receives a reward or penalty as feedback.
- The agent continues doing these three things (take action, change state/remain in the same state, and get feedback), and by doing these actions, he learns and explores the environment.
- The agent learns that what actions lead to positive feedback or rewards and what actions lead to negative feedback penalty. As a positive reward, the agent gets a positive point, and as a penalty, it gets a negative point.



Terms used in Reinforcement Learning

- **Agent()**: An entity that can perceive/explore the environment and act upon it.
- **Environment()**: A situation in which an agent is present or surrounded by. In RL, we assume the stochastic environment, which means it is random in nature.
- **Action()**: Actions are the moves taken by an agent within the environment.
- **State()**: State is a situation returned by the environment after each action taken by the agent.
- **Reward()**: A feedback returned to the agent from the environment to evaluate the action of the agent.
- **Policy()**: Policy is a strategy applied by the agent for the next action based on the current state.
- **Value()**: It is expected long-term return with the discount factor and opposite to the short-term reward.
- **Q-value()**: It is mostly similar to the value, but it takes one additional parameter as a current action (a).

Key Features of Reinforcement Learning

- In RL, the agent is not instructed about the environment and what actions need to be taken.
- It is based on the hit and trial process.
- The agent takes the next action and changes states according to the feedback of the previous action.
- The agent may get a delayed reward.
- The environment is stochastic, and the agent needs to explore it to reach to get the maximum positive rewards.

Approaches to implement Reinforcement Learning

There are mainly three ways to implement reinforcement-learning in ML, which are:

1. Value-based

The value-based approach is about to find the optimal value function, which is the maximum value at a state under any policy. Therefore, the agent expects the long-term return at any state(s) under policy $\bar{\theta}$.

2. Policy-based:

Policy-based approach is to find the optimal policy for the maximum future rewards without using the value function. In this approach, the agent tries to apply such a policy that the action performed in each step helps to maximize the future reward.

The policy-based approach has mainly two types of policy:

- **Deterministic:** The same action is produced by the policy ($\bar{\theta}$) at any state.
- **Stochastic:** In this policy, probability determines the produced action.

3. Model-based:

In the model-based approach, a virtual model is created for the environment, and the agent explores that environment to learn it. There is no particular solution or algorithm for this approach because the model representation is different for each environment.

Elements of Reinforcement Learning

There are four main elements of Reinforcement Learning, which are given below:

1. Policy
2. Reward Signal
3. Value Function
4. Model of the environment

1. Policy

A policy can be defined as a way how an agent behaves at a given time. It maps the perceived states of the environment to the actions taken on those states. A policy is the core element of the RL as it alone can define the behavior of the agent. In some cases, it may be a simple function or a lookup table, whereas, for other cases, it may involve general computation as a search process. It could be deterministic or a stochastic policy:

For deterministic policy: $a = \pi(s)$

For stochastic policy: $\pi(a|s) = P[A_t = a | S_t = s]$

2. Reward Signal

The goal of reinforcement learning is defined by the reward signal. At each state, the environment sends an immediate signal to the learning agent, and this signal is known as a reward signal. These rewards are given according to the good and bad actions taken by the agent. The agent's main objective is to maximize the total number of rewards for good actions. The reward signal can change the policy, such as if an action selected by the agent leads to low reward, then the policy may change to select other actions in the future.

3. Value Function

The value function gives information about how good the situation and action are and how much reward an agent can expect. A reward indicates the immediate signal for each good and bad

action, whereas a value function specifies the good state and action for the future. The value function depends on the reward as, without reward, there could be no value. The goal of estimating values is to achieve more rewards.

4. Model

The last element of reinforcement learning is the model, which mimics the behavior of the environment. With the help of the model, one can make inferences about how the environment will behave. Such as, if a state and an action are given, then a model can predict the next state and reward.

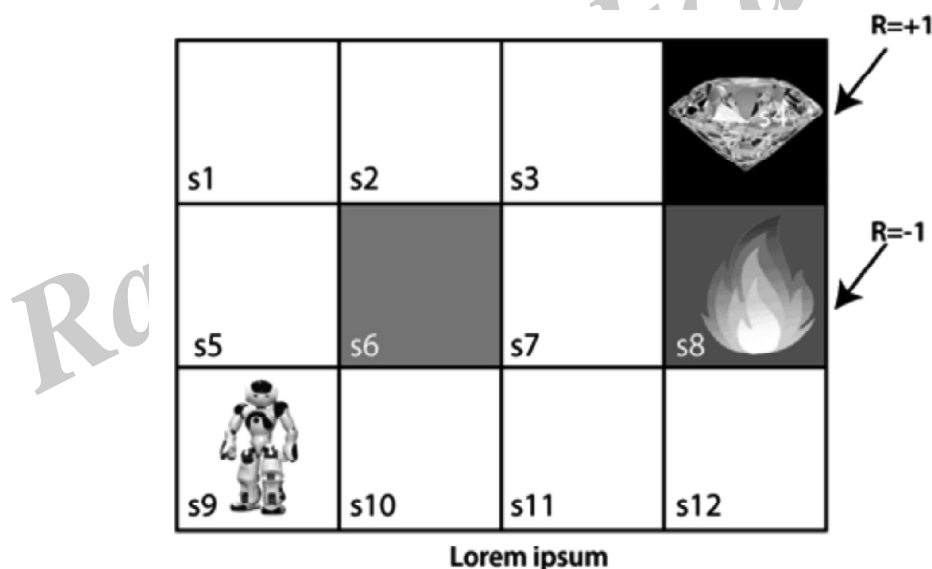
The model is used for planning, which means it provides a way to take a course of action by considering all future situations before actually experiencing those situations. The approaches for solving the RL problems with the help of the model are termed as the model-based approach. Comparatively, an approach without using a model is called a model-free approach.

How does Reinforcement Learning Work?

To understand the working process of the RL, we need to consider two main things:

- **Environment:** It can be anything such as a room, maze, football ground, etc.
- **Agent:** An intelligent agent such as AI robot.

Let's take an example of a maze environment that the agent needs to explore. Consider the below image:

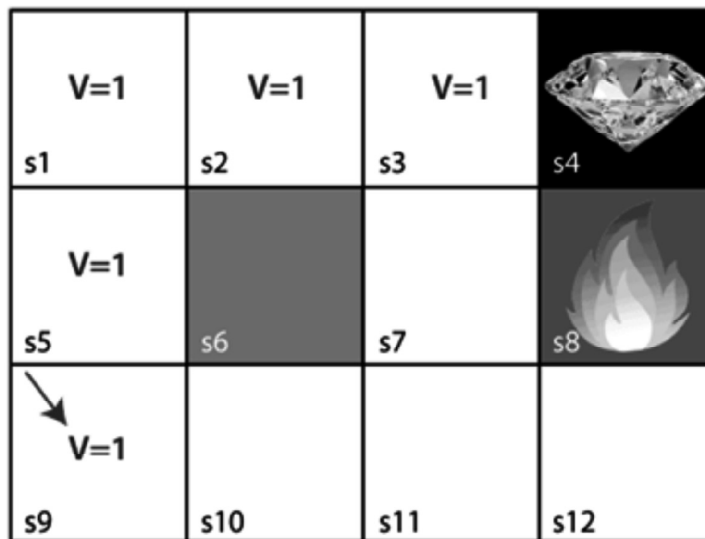


In the above image, the agent is at the very first block of the maze. The maze is consisting of an S_6 block, which is a wall, S_8 a fire pit, and S_4 a diamond block.

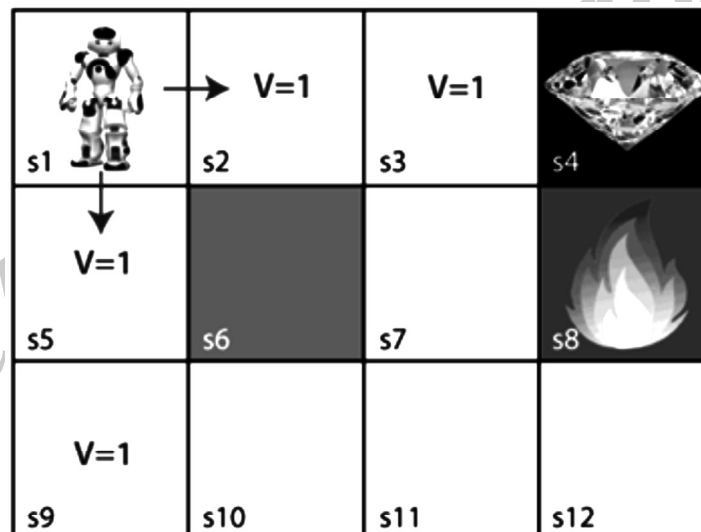
The agent cannot cross the S_6 block, as it is a solid wall. If the agent reaches the S_4 block, then get the +1 reward; if it reaches the fire pit, then gets -1 reward point. It can take four actions: move up, move down, move left, and move right.

The agent can take any path to reach to the final point, but he needs to make it in possible fewer steps. Suppose the agent considers the path S_9 - S_5 - S_1 - S_2 - S_3 , so he will get the +1-reward point.

The agent will try to remember the preceding steps that it has taken to reach the final step. To memorize the steps, it assigns 1 value to each previous step. Consider the below step:



Now, the agent has successfully stored the previous steps assigning the 1 value to each previous block. But what will the agent do if he starts moving from the block, which has 1 value block on both sides? Consider the below diagram:



It will be a difficult condition for the agent whether he should go up or down as each block has the same value. So, the above approach is not suitable for the agent to reach the destination. Hence to solve the problem, we will use the Bellman equation, which is the main concept behind reinforcement learning.

Q10. What is the use of Bellman Equation in machine learning? Explain with example.

Ans :

The Bellman Equation

The Bellman equation was introduced by the Mathematician Richard Ernest Bellman in the year 1953, and hence it is called as a Bellman equation. It is associated with dynamic programming and used to calculate the values of a decision problem at a certain point by including the values of previous states.

It is a way of calculating the value functions in dynamic programming or environment that leads to modern reinforcement learning.

The key-elements used in Bellman equations are:

- Action performed by the agent is referred to as "a"
- State occurred by performing the action is "s."
- The reward/feedback obtained for each good and bad action is "R."
- A discount factor is Gamma " γ "

The Bellman equation can be written as:

$$V(s) = \max [R(s,a) + \gamma V(s')]$$

Where,

$V(s)$ = value calculated at a particular point.

$R(s,a)$ = Reward at a particular state s by performing an action.

γ = Discount factor

$V(s')$ = The value at the previous state.

In the above equation, we are taking the max of the complete values because the agent tries to find the optimal solution always.

So now, using the Bellman equation, we will find value at each state of the given environment. We will start from the block, which is next to the target block.

For 1st block:

$V(s_3) = \max [R(s,a) + \gamma V(s')]$, here $V(s') = 0$ because there is no further state to move.

$$V(s_3) = \max[R(s,a)] \Rightarrow V(s_3) = \max[1] \Rightarrow V(s_3) = 1.$$

For 2nd block:

$V(s_2) = \max [R(s,a) + \gamma V(s')]$, here $\gamma = 0.9$ (lets), $V(s') = 1$, and $R(s, a) = 0$, because there is no reward at this state.

$$V(s_2) = \max[0.9(1)] \Rightarrow V(s_2) = \max[0.9] \Rightarrow V(s_2) = 0.9$$

For 3rd block:

$V(s_1) = \max [R(s,a) + \gamma V(s')]$, here $\gamma = 0.9$ (lets), $V(s') = 0.9$, and $R(s, a) = 0$, because there is no reward at this state also.

$$V(s_1) = \max[0.9(0.9)] \Rightarrow V(s_1) = \max[0.81] \Rightarrow V(s_1) = 0.81$$

For 4th block:

$V(s_5) = \max [R(s,a) + \gamma V(s')]$, here $\gamma = 0.9$ (lets), $V(s') = 0.81$, and $R(s, a) = 0$, because there is no reward at this state also.




$$V(s_5) = \max[0.9(0.81)] \Rightarrow V(s_5) = \max[0.73] \Rightarrow V(s_5) = 0.73$$

For 5th block:



$V(s_9) = \max [R(s,a) + \gamma V(s')]$, here $\gamma = 0.9$ (lets), $V(s') = 0.73$, and $R(s, a) = 0$, because there is no reward at this state also.

$$V(s_9) = \max[0.9(0.73)] \Rightarrow V(s_9) = \max[0.66] \Rightarrow V(s_9) = 0.66$$



Consider the below image:

$V=0.81$ s1	$V=0.9$ s2	$V=1$ s3	 s4
$V=0.73$ s5	s6	s7	 s8
 $V=0.66$ s9	s10	s11	s12

Now, we will move further to the 6th block, and here agent may change the route because it always tries to find the optimal path. So now, let's consider from the block next to the fire pit.

$V=0.81$ s1	$V=0.9$ s2	$V=1$ s3	 s4
$V=0.73$ s5	s6	s7	 s8
$V=0.66$ s9	s10	s11	s12

Now, the agent has three options to move; if he moves to the blue box, then he will feel a bump if he moves to the fire pit, then he will get the -1 reward. But here we are taking only positive rewards, so for this, he will move to upwards only. The complete block values will be calculated using this formula. Consider the below image:

V=0.81 s1	V=0.9 s2	V=1 s3	 s4
V=0.73 s5		V=0.9 s7	 s8
V=0.66 s9	V=0.73 s10	V=0.81 s11	V=0.73 s12

Q11. Explain Q-Learning algorithm of Reinforcement Learning Algorithms.

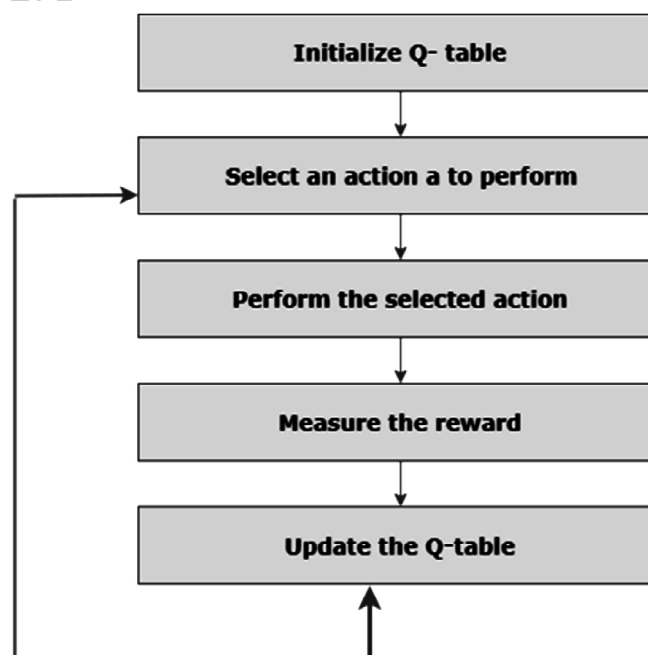
Ans :

Reinforcement Learning Algorithms

Reinforcement learning algorithms are mainly used in AI applications and gaming applications. The main used algorithms are:

Q-Learning

- Q-learning is an Off policy RL algorithm, which is used for the temporal difference Learning. The temporal difference learning methods are the way of comparing temporally successive predictions.
- It learns the value function $Q(S, a)$, which means how good to take action "a" at a particular state "s."
- The below flowchart explains the working of Q-learning:



State Action Reward State action (SARSA)

- SARSA stands for State Action Reward State action, which is an on-policy temporal difference learning method. The on-policy control method selects the action for each state while learning using a specific policy.
- The goal of SARSA is to calculate the $Q_{\bar{\theta}}(s, a)$ for the selected current policy $\bar{\theta}$ and all pairs of (s-a).
- The main difference between Q-learning and SARSA algorithms is that unlike Q-learning, the maximum reward for the next state is not required for updating the Q-value in the table.
- In SARSA, new action and reward are selected using the same policy, which has determined the original action.
- The SARSA is named because it uses the quintuple $Q(s, a, r, s', a')$. Where,
s: original state
a: Original action
r: reward observed while following the states
s' and a': New state, action pair.

Deep Q Neural Network (DQN)

- As the name suggests, DQN is a Q-learning using Neural networks.
- For a big state space environment, it will be a challenging and complex task to define and update a Q-table.
- To solve such an issue, we can use a DQN algorithm. Where, instead of defining a Q-table, neural network approximates the Q-values for each action and state.

Now, we will expand the Q-learning.

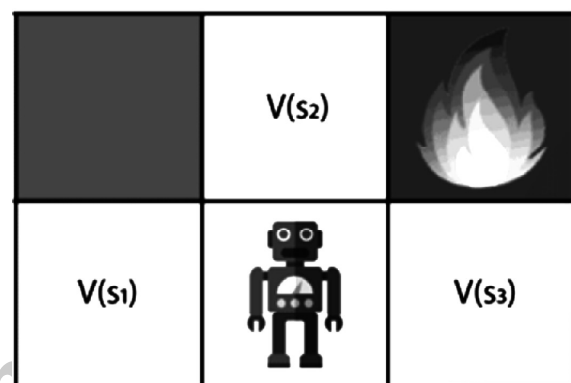
Q-Learning Explanation

- Q-learning is a popular model-free reinforcement learning algorithm based on the Bellman equation.
- The main objective of Q-learning is to learn the policy which can inform the agent that what actions should be taken for maximizing the reward under what circumstances.
- It is an off-policy RL that attempts to find the best action to take at a current state.

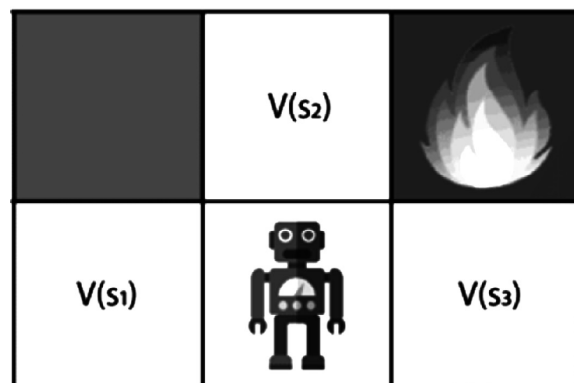
- The goal of the agent in Q-learning is to maximize the value of Q.
- The value of Q-learning can be derived from the Bellman equation. Consider the Bellman equation given below:

$$V(s) = \max_a [R(s, a) + \gamma \sum_{s'} P(s, a, s') V(s')]$$

In the equation, we have various components, including reward, discount factor ($\bar{\alpha}$), probability, and end states s'. But there is no any Q-value is given so first consider the below image:



In the above image, we can see there is an agent who has three values options, $V(s_1)$, $V(s_2)$, $V(s_3)$. As this is MDP, so agent only cares for the current state and the future state. The agent can go to any direction (Up, Left, or Right), so he needs to decide where to go for the optimal path. Here agent will take a move as per probability bases and changes the state. But if we want some exact moves, so for this, we need to make some changes in terms of Q-value. Consider the below image:



Q- represents the quality of the actions at each state. So instead of using a value at each state, we will use a pair of state and action, i.e., $Q(s, a)$. Q-value specifies that which action is more lubricative than others, and according to the best Q-value, the agent takes his next move. The Bellman equation can be used for deriving the Q-value.

To perform any action, the agent will get a reward $R(s, a)$, and also he will end up on a certain state, so the Q -value equation will be:

$$Q(S, a) = R(s, a) + \sum_{s'} P(s, a, s') V(s')$$

Hence, we can say that,

$$V(s) = \max [Q(s, a)]$$

$$Q(S, a) = R(s, a) + \gamma \sum_{s'} (P(s, a, s') \max_{a'} Q(S', a'))$$

The above formula is used to estimate the Q-values in Q-Learning.

What is 'Q' in Q-learning?

The Q stands for quality in Q-learning, which means it specifies the quality of an action taken by the agent.

Q-table:

A Q-table or matrix is created while performing the Q-learning. The table follows the state and action pair, i.e., $[s, a]$, and initializes the values to zero. After each action, the table is updated, and the q-values are stored within the table.

The RL agent uses this Q-table as a reference table to select the best action based on the q-values.

5.3.1 Bayesian Networks

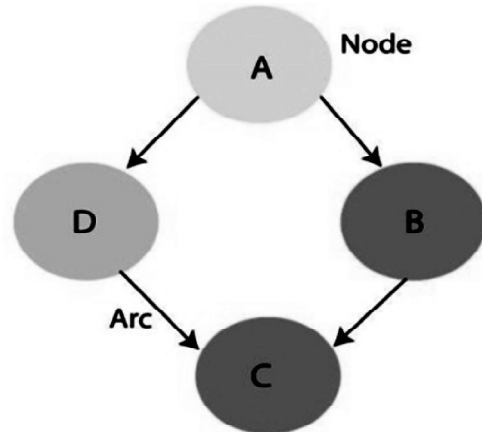
Q12. What are Bayesian Networks? How can we use Bayesian Networks in probability computations? Explain with an example.

Ans :

(Imp.)

By definition, Bayesian Networks are a type of Probabilistic Graphical Model that uses the Bayesian inferences for probability computations. It represents a set of variables and its conditional probabilities with a Directed Acyclic Graph (DAG). They are primarily suited for considering an event

that has occurred and predicting the likelihood that any one of the several possible known causes is the contributing factor.



As mentioned above, by making use of the relationships which are specified by the Bayesian Network, we can obtain the Joint Probability Distribution (JPF) with the conditional probabilities. Each node in the graph represents a random variable and the arc (or directed arrow) represents the relationship between the nodes. They can be either continuous or discrete in nature.

In the above diagram A, B, C and D are 4 random variables represented by nodes given in the network of the graph. To node B, A is its parent node and C is its child node. Node C is independent of Node A.

Before we get into the implementation of a Bayesian Network, there are a few probability basics that have to be understood.

Local Markov Property

The Bayesian Networks satisfy the property known as the Local Markov Property. It states that a node is conditionally independent of its non-descendants, given its parents. In the above example, $P(D|A, B)$ is equal to $P(D|A)$ because D is independent of its non-descendent, B. This property aids us in simplifying the Joint Distribution. The Local Markov Property leads us to the concept of a Markov Random Field which is a random field around a variable that is said to follow Markov properties.

Conditional Probability

In mathematics, the Conditional Probability of event A is the probability that event A will occur given that another event B has already occurred. In simple terms, $p(A | B)$ is the probability of event A occurring, given that event, B occurs. However, there are two types of event possibilities between A and B. They may be either dependent events or independent events. Depending upon their type, there are two different ways to calculate the conditional probability.

- Given A and B are dependent events, the conditional probability is calculated as $P(A | B) = P(A \text{ and } B) / P(B)$
- If A and B are independent events, then the expression for conditional probability is given by, $P(A | B) = P(A)$

Joint Probability Distribution

Before we get into an example of Bayesian Networks, let us understand the concept of Joint Probability Distribution. Consider 3 variables a_1 , a_2 and a_3 . By definition, the probabilities of all different possible combinations of a_1 , a_2 , and a_3 are called its Joint Probability Distribution.

If $P[a_1, a_2, a_3, \dots, a_n]$ is the JPD of the following variables from a_1 to a_n , then there are several ways of calculating the Joint Probability Distribution as a combination of various terms such as,

$$\begin{aligned} P[a_1, a_2, a_3, \dots, a_n] &= P[a_1 | a_2, a_3, \dots, a_n] * P[a_2, a_3, \dots, a_n] \\ &= P[a_1 | a_2, a_3, \dots, a_n] * P[a_2 | a_3, \dots, a_n] \dots P[a_{n-1} | a_n] * P[a_n] \end{aligned}$$

Generalizing the above equation, we can write the Joint Probability Distribution as,

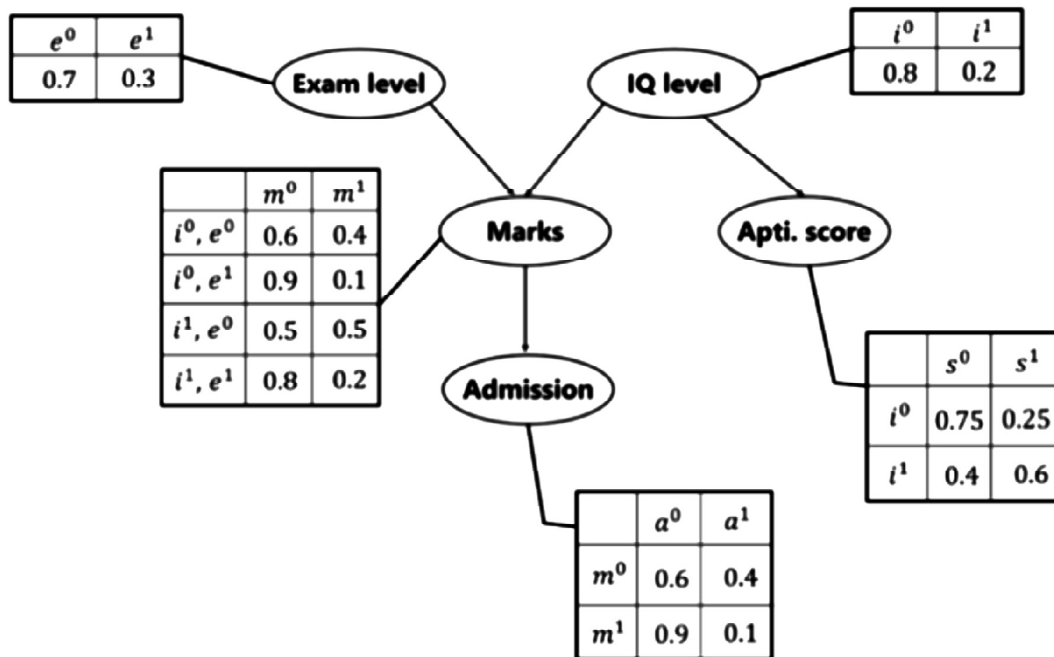
$$P(X_i | X_{i-1}, \dots, X_n) = P(X_i | \text{Parents}(X_i))$$

Example of Bayesian Networks

Let us now understand the mechanism of Bayesian Networks and their advantages with the help of a simple example. In this example, let us imagine that we are given the task of modeling a student's marks (m) for an exam he has just given. From the given Bayesian Network Graph below, we see that the marks depend upon two other variables. They are,

- **Exam Level (e):** This discrete variable denotes the difficulty of the exam and has two values (0 for easy and 1 for difficult)
- **IQ Level (i):** This represents the Intelligence Quotient level of the student and is also discrete in nature having two values (0 for low and 1 for high)

Additionally, the IQ level of the student also leads us to another variable, which is the Aptitude Score of the student (s). Now, with marks the student has scored, he can secure admission to a particular university. The probability distribution for getting admitted (a) to a university is also given below.



In the above graph, we see several tables representing the probability distribution values of the given 5 variables. These tables are called the Conditional Probabilities Table or CPT. There are a few properties of the CPT given below :

- The sum of the CPT values in each row must be equal to 1 because all the possible cases for a particular variable are exhaustive (representing all possibilities).
- If a variable that is Boolean in nature has k Boolean parents, then in the CPT it has 2^k probability values.

Coming back to our problem, let us first list all the possible events that are occurring in the above-given table.

1. Exam Level (e)
2. IQ Level (i)
3. Aptitude Score (s)
4. Marks (m)
5. Admission (a)

These five variables are represented in the form of a Directed Acyclic Graph (DAG) in a Bayesian Network format with their Conditional Probability tables. Now, to calculate the Joint Probability Distribution of the 5 variables the formula is given by,

$$P[a, m, i, e, s] = P(a | m) \cdot P(m | i, e) \cdot P(i) \cdot P(e) \cdot P(s | i)$$

From the above formula,

- $P(a | m)$ denotes the conditional probability of the student getting admission based on the marks he has scored in the examination.
- $P(m | i, e)$ represents the marks that the student will score given his IQ level and difficulty of the Exam Level.

- $P(i)$ and $P(e)$ represent the probability of the IQ Level and the Exam Level.
- $P(s | i)$ is the conditional probability of the student's Aptitude Score, given his IQ Level.

With the following probabilities calculated, we can find the Joint Probability Distribution of the entire Bayesian Network.

Calculation of Joint Probability Distribution

Let us now calculate the JPD for two cases.

Case 1:

Calculate the probability that in spite of the exam level being difficult, the student having a low IQ level and a low Aptitude Score, manages to pass the exam and secure admission to the university.

From the above word problem statement, the Joint Probability Distribution can be written as below,

$$P[a=1, m=1, i=0, e=1, s=0]$$

From the above Conditional Probability tables, the values for the given conditions are fed to the formula and is calculated as below.

$$\begin{aligned} P[a=1, m=1, i=0, e=0, s=0] &= P(a=1 | m=1) \cdot P(m=1 | i=0, e=1) \cdot P(i=0) \cdot P(e=1) \cdot P(s=0 | i=0) \\ &= 0.1 * 0.1 * 0.8 * 0.3 * 0.75 \\ &= \mathbf{0.0018} \end{aligned}$$

Case 2:

In another case, calculate the probability that the student has a High IQ level and Aptitude Score, the exam being easy yet fails to pass and does not secure admission to the university.

The formula for the JPD is given by

$$P[a=0, m=0, i=1, e=0, s=1]$$

Thus,

$$\begin{aligned} P[a=0, m=0, i=1, e=0, s=1] &= P(a=0 | m=0) \cdot P(m=0 | i=1, e=0) \cdot P(i=1) \cdot P(e=0) \cdot P(s=1 | i=1) \\ &= 0.6 * 0.5 * 0.2 * 0.7 * 0.6 \\ &= \mathbf{0.0252} \end{aligned}$$

Hence, in this way, we can make use of Bayesian Networks and Probability tables to calculate the probability for various possible events that occur.

**5.4 USE CASES OF VARIOUS ML ALGORITHMS
IN MANUFACTURING, RETAIL, TRANSPORT,
HEALTHCARE, WEATHER, INSURANCE SECTORS**

Q13. Explain the uses of machine learning in manufacturing sector.

Ans :

Machine learning solutions have been developed for various applications in the manufacturing industry, including data analytics, quality control, and others. Here are some of the top machine learning applications in manufacturing operations that are helping to revolutionize the sector.

1. Predictive maintenance

Predictive maintenance is one of the key use cases for ML in manufacturing because it can preempt the failure of vital machinery or components using algorithms. By analyzing data from previous maintenance cycles, machine learning can identify patterns that can be used to predict equipment failures and when future maintenance will be needed. This information can then be used to schedule maintenance before problems occur. This, in turn, could save manufacturers significant time and money since it allows them to tackle specific issues exactly when needed—and in a highly focused way. This benefits manufacturers by:

- Significantly reducing planned and unplanned downtime and, thus, costs.
- Providing technicians with focused inspection, repair and tool requirements.
- Prolonging the remaining useful life (RUL) of machinery by preventing any secondary damage during repairs.
- Reducing the size of the technical team needed to make repairs.

However, even with the best algorithm, predictive quality analytics will only be as effective as the data that is used to train it. In order to be successful, manufacturers must have a well-designed data collection strategy

that captures all relevant information about their process.

2. Predictive quality and yield

As consumer demand grows in line with an expanding population, process-based losses are becoming harder for manufacturers to tolerate. AI and machine learning can enable businesses to get to the root cause of losses related to quality, yield, energy efficiency and so on, thereby protecting their bottom line and enabling them to remain competitive. It does so using continuous, multivariate analysis via process-tailored ML algorithms, and also through machine learning-enabled Root Cause Analysis (RCA).

ML and AI-driven RCA, in particular, is a powerful tool for tackling process-based wastage and is far more effective than manual RCA for the following reasons:

- With automated RCA, machine learning algorithms harness historical data models to identify patterns in new data and make predictions on where losses may be occurring—preempting issues ahead of time.
- This method, over manual RCA, is entirely data-driven and completely unbiased.
- It's also unclouded by daily admin and other manual tasks performed by process experts, so the focus is purely on optimizing processes.

3. Digital twins

A digital twin - a real - time digital representation of a physical object or, indeed, a process can be used by manufacturers to carry out instant diagnostics, evaluate production processes, and make performance predictions. But more than this, digital twins can help manufacturers revolutionise their engineering practices while offering full design, production and operational customisation. So, in other words, manufacturing companies can create a virtual representation of their products and processes, which can be used to test and optimise them before they are built. The benefits of ML-enabled digital twins in manufacturing include:

- Significant cost reductions
- Improved reliability of production lines
- Optimised performance and productivity
- Reduced risks on the shop floor
- Improved quality and full customisation
- Streamlined maintenance

4. **Generative design/smart manufacturing**

According to Reportlinker, the global smart manufacturing market is predicted to be worth \$314 billion by 2026. AI and machine learning have the capability to create an almost infinite number of design solutions to match any problem/product based on preset factors like size, materials, weight, etc. This allows engineers to find the very best design solution for a product before it goes into production. Machine learning uses generator and discriminator models to:

- Create new designs for specified products
- Distinguish between generated and real products
- Train deep learning algorithms to recognise and define every possible design solution, thus optimising the design for a specified task
- Make the computer a “design partner.”

5. **Energy consumption forecasting**

Manufacturers can now use machine learning algorithms that process data on factors like temperature, lighting, activity levels within a facility and more to build predictive models of likely energy consumption in the future. Machine learning algorithms can analyse large data sets to identify patterns and relationships that would be difficult to find using traditional methods. They do this using:

- Sequential data measurements.
- Autoregressive data models that identify cyclical/seasonal trends - data scientists will often pair this approach with feature engineering, which turns raw and

unordered data into “features” for algorithms to define and build predictive analytics models on.

- Deep neural networks - which can process vast quantities of data and rapidly identify patterns.

Forecasting energy consumption is important for manufacturing for a number of reasons. First, it can help factory owners and operators plan for future energy needs. This planning is essential to ensuring that factories have the necessary resources to meet production demands. Additionally, forecasting energy consumption can help factories avoid disruptions in production due to unexpected changes in energy costs or availability.

6. **Cognitive supply chain management**

With the proliferation of IIoT technologies, it's only a matter of time before smart supply chains completely redefine how manufacturers carry out their operations. Automation is the first rung on the ladder, but soon entire supply chains could be “cognitive”. This means that they can use AI and machine learning algorithms to perform automatic analysis of datasets, including inbound and outbound shipments, inventory, consumer preferences, market trends, and even weather forecasts for predicting optimal shipping conditions. Key areas enhanced by cognitive supply chain management will be:

- **Warehouse control:** Stock control facilitated by deep learning-based computer vision systems, enabling the rapid replenishment of supplies.
- **Demand forecasting:** The analysis of customer behaviours and preferences using time series analysis, feature engineering, and NLP techniques.
- **Logistics route optimisation:** manufacturers can review and allocate the most optimal routes for shipping goods using machine learning algorithms.
- **Transport optimisation:** Assessing impacts on shipments and deliverables using machine and deep learning algorithms to optimise transportation solutions.

Q14. Describe the use of machine learning in retail sector.*Ans :***(Imp.)****Retail**

Machine learning in retail relies on self-improving computer algorithms created to process data, spot recurring patterns and anomalies among variables, and autonomously learn how such relations affect or determine the industry's trends, phenomena, and business scenarios.

1. Targeted ads

While mostly used in ecommerce, targeted marketing represents a powerful tool to route potential customers both towards online platforms and traditional stores. This involves segmenting users based on an ensemble of behavioral, psychographic, demographic and geographic parameters (such as their purchase and browsing history, age, gender, interests, region, etc.) and targeting them with fully personalized ads and promotions.

2. Contextual shopping

A different, more interactive solution to catch users' attention and lead them towards your ecommerce platform is contextual shopping. This marketing tool taps into machine learning and computer vision to identify and point out the merchandise shown in videos and pics on social media while offering a "shortcut" to reach the related product page in an online shop.

3. Recommender engines

Once users land on an online platform, they may feel lost among a massive selection of merchandise. Recommendation engines are powerful tools designed to drive customers towards the products they may actually need.

To provide tailored suggestions, these systems can adopt a content-based filtering approach, namely recommend items with similar features to those purchased in the past, or opt for collaborative filtering, which implies suggesting products ordered by other customers with similar purchase patterns, personal traits, and interests.

4. Dynamic pricing

Product recommendation and ads are not the only things dynamically changing thanks to machine learning. Nowadays, most online stores and ecommerce platforms constantly adjust prices depending on the fluctuations in product demand and supply, competitors' promotions and pricing strategies, broader sales trends, and more.

5. Chatbots

Chatbots and virtual assistants are highly interactive tools powered by machine learning and NLP and capable of providing customers with 24/7 user support (including information about available products and shipping options) while sending reminders, coupons, and personalized suggestions to upscale your sales.

6. Supply chain management

Product replenishment and other inventory management operations should never be left to chance. To better match product supply and demand, optimize space utilization in warehouses, and avoid food spoilage, it's worth relying on the analytical and forecasting capabilities of machine learning algorithms. This means considering several variables, such as price fluctuations or seasonality-based purchase patterns, predicting future sales trends, and therefore planning proper restocking initiatives.

7. Delivery optimization

Another aspect of logistics which can be enhanced through machine learning is product delivery. ML-powered systems, fuelled with traffic and weather data collected through networks of IoT sensors and cameras, can easily calculate the fastest delivery routes. By processing user data, instead, they may recommend suitable delivery methods to better meet customers' needs.

But the apotheosis of this approach is probably the ML-based anticipatory shipping technique implemented by Amazon, which allows to forecast future deliveries based on customers' purchase patterns, move products to a closer warehouse, and therefore be able to dispatch them faster and inexpensively when the actual order is placed.

8. Autonomous vehicles

This embodiment of machine learning and computer vision for product delivery is still far from being perfected and implemented on a vast scale. However, companies like Amazon and Kroger are betting on this technology and soon we may rely on self-driving vehicles to speed up product distribution.

9. Video surveillance

ML-powered computer vision systems can drive vehicles... and spot thieves. The major difference between these tools and traditional video surveillance solutions is that the latter identify intruders based on a rather inaccurate rule-based approach that suffers from lots of false positives. Machine learning systems, on the other hand, can recognize more subtle behavioral patterns and alert the management if something suspicious happens.

10. Fraud detection

When it comes to online retailers and ecommerce platforms, thieves are more likely to steal money from credit cards than products from a shelf. Since machine learning algorithms are designed to identify recurring patterns, they can also pinpoint any event deviating from the norm, including anomalous transaction frequency or account data inconsistencies, and flag it as suspicious for further inspection.

Q15. Explain the machine learning algorithm in transport sector.

Ans :

Transportation

Machine learning had great applicability in the transport industry. In recent years, ML techniques have become a part of smart transportation. Through deep learning, ML explored the complex interactions of roads, highways, traffic, environmental elements, crashes, and so on. ML has also great potential in daily traffic management and the collection of traffic data.

Machine learning can also help back-office operations as well. Let's take, for instance, a transport company. Daily, they can receive dozens if not hundreds of orders, depending on how big the company is. Imagine that all those transport orders are manually processed. These operations take a huge amount of time to do it and also is considered to be a boring and error-prone task. To ensure the flow of the transports, the order processing must be done at a certain time

Navigation

Navigation is being used in most of vehicles these days. Where one can drive to a destination with voice assistant and accuracy. Real-time vehicle tracking is possible because of ML and data analytics.

App support to Vehicles

Isn't it amazing that you can regulate the temperature of the car while sitting inside your home? With the help of ML- powered app, you can various such controls.

Self-driving Cars

Self-driving cars or autonomous vehicles look like sci-fi. But it is practical with the help of ML. A car can drive itself to a destination, although a driver needs to be inside the car for emergencies.

Drone Taxi

Many Ecommerce companies are using this to deliver their parcels to the buyer. Many businesses, defense is using this as a logistic and locomotive option. These taxis reduce the travel time, carbon emission, and cost incurred.

Traffic Management

It is one of the best applications of ML in real life. It helps people as well as transportation companies to optimize their routes and prevents traffic congestion.

Q16. Discuss the machine learning algorithm in healthcare sector.

Ans :

Health care

Machine learning in the healthcare sector helps improve decision-making, optimized innovation, improved efficiency of research/clinical trials, and new tool creation for physicians, consumers, insurers, and regulators. Research and Development in Healthcare are directed towards identifying the risk of developing sepsis and diagnosing breast cancer.

These have to be carefully planned and implemented so that it helps the patients and brings returns on investment. Using open-source data sciences, developers can build systems in such a way that it addresses the requirements of the healthcare sector and improves patient's conditions. Here are four machine learning use cases for the healthcare sector that can be developed with open-source data science tools and adapted for different functions.

1. Patient-Risk Identification

Machine learning models have several advantages over classic linear models when it comes to patient-level predictions that are important in a value-based care framework. Provider organizations can use machine learning models to help identify the members who would benefit most from active care management to improve patient outcomes and reduce costs under value-based contracting.

Advances in machine learning, particularly in the healthcare sector have helped the doctors across the globe to use algorithms to detect patterns and subsequently predict heart attacks, sepsis, and diseases. If doctors are aware of the condition beforehand, they can start taking preventive measures and practice risk management. The Sepsis Sniffer Algorithm uses demographic signals to alert the individual staff whenever the risk of developing the disease increases. This algorithm was developed by Mayo Clinic and has decreased the manual screening time by 72%.

Another such example is El Camino Hospital. Their researchers used electronic health records, bed alarm data, and nurse call data to develop a tool for predicting patient falls. This new tool alerts staff when a patient is at high risk for falling so they can take action to reduce the risk. They managed to reduce falls by 39%.

2. Oncology

Deep learning can be customized to algorithms that can identify early development of tumours in the lungs, breasts, brains, etc. This happens because the algorithms can be designed to provide information about the patterns in patients and the development of the disease and alert an early detection. Algorithms can be trained to recognize intricate patterns in radiographic imaging data. Researchers all over the world have tried to benefit from these tools to detect essential features from gene expression data, to identify brain cells of breast cancer. Houston Methodist Research developed such a device, and this resulted in the information generation getting 30 times faster than that of a human report.

A research paper published by Stanford University researchers has shown that their convolutional neural network (CNN) achieves performance on par with all tested experts when classifying skin cancer. Google's CNN system has demonstrated the ability to identify deadly skin cancers at an accuracy rate on par with practitioners, potentially extending diagnosis reach outside the clinic and into service-based apps that have been popping up as mobile access expands worldwide.

Convolutional Neural Networks can be used for the diagnosis of skin cancer that otherwise relies on procedures including clinical screening, dermoscopic analysis, and histopathological examination. This process usually takes up a lot of time. Still, machine learning models are trained using thousands of images of malignant and benign skin lesions, which increases the accuracy of detection by 87-95%.

3. Pharma

AI and machine learning can potentially help develop and improve the pharma sector too. The applications of DNNs in drug discovery have been numerous and include bioactivity prediction, de novo molecular design, synthesis prediction, and biological image analysis. One advantage of DNNs is that they have several different flexible architectures and are thus used to answer a variety of questions. The use of machine learning can be put to use from initial screening of drug compounds and their consequent results based on biological factors.

The NLP tool can successfully read and interpret thousands of reports and get us years ahead in the field. A team of researchers from the U.S. and Ireland worked together to conduct a study on Adverse Drug Events (ADEs) using text mining, predictive analytics, and neural networks to analyze vast databases of medical literature and social media posts for comments related to drug side effects. They were able to list the side effects of the respective drugs with the help of these machine learning tools.

4. NLP in Administration

Natural Language Processing can help doctors and other medical staff to avoid filing reports manually, and let the software do it. More than 80% of respondents reported a physician's burnout problem. By leveraging NLP tools that use algorithms to identify and categorize words and phrases, physicians can dictate notes directly to EHRs during patient visits. This will facilitate the administrative work of hospitals and increase efficiencies and clarity in the reports. These tools will help the doctors to spend more time with the patients.

Q17. Discuss the use of machine learning algorithm in weather.

Ans :

Weather

Weather forecasting is the prediction of weather and conditions of the atmosphere for a specific time, weather conditions include rain, snow, temperature, fog, wind etc, they are various techniques for

the prediction of weather which includes persistence forecast, climatological forecast, synoptic forecasting, statistical forecasting, computational forecast etc. Due to the erratic, uncertain complex nature of the weather, forecasting has been a tedious, challenging task that involves expensive, complex, and computational processes

Weather forecast has a big impact on the global economy, these impacts can be categorized into four general categories, they include low-impact, moderate-impact, high-impact, and extreme-impact humans depend on weather forecast because it affects many aspects of our livelihood and lifestyle, precise weather forecast plays a critical role in decision making for severe weather management and for primary and secondary sectors like agriculture, transportation, tourism, and industry because they rely on good weather conditions for production and operations.

Weather forecast is made by collecting quantitative, atmospheric data about the past and current state of the atmosphere known as determinates, they include temperature, pressure, the humidity of the air, precipitations, wind, and other meteorological elements, these abiotic determinates are inputted into a model base, early success in forecasting was in terms of Numeric weather predictions

Numerical weather prediction (NWP) such as Weather Research and Forecasting (WRF) model uses mathematical equations or models to compute current weather observations collected from the weather station to produce weather predictions either for short term weather forecasts or long-term climate change

These current weather prediction models depend on the complex physical model and require large computational power to run, whose forecasts are often inaccurate more recently with the advancement of technology and the availability of metrological big data, researchers had begun utilizing data-driven approaches in metrology, data-driven approaches include using machine learning methods and other technological advance methods which have achieved considerable success in weather forecast

Machine learning approach to weather forecast uses quantitative metrological data to build models and tries to improve the performance of the model by learning from the dataset.

Q18. Explain the machine learning algorithm in insurance industry.

Ans :

(Imp.)

Insurance**1. Claims processing**

The first application of machine learning to the insurance industry is claims management. It can help companies get rid of any manual processing and, hence, provide end-users with better and faster service. Apart from this, automated claims processing means improved decision-making and reduced risks.

Here are more specific applications of machine learning in claims management:

- **Claims registration:** Typical claims registration process takes lots of time and is data intensive. ML can provide insurers with analytical insights on how to remove these operation inefficiencies.
- **Claims triage:** ML can also be useful in scoring and triaging risks. If an ML system in insurance learns based on past experience, it will be able to prioritize insurance claims faster and more accurately.
- **Claims volume forecast:** A typical stumbling block in an insurance practice is to set premiums before signing any insurance contract. An insurance agent, in this case, has to go through lots of manual work and make predictions about the number of claims occurrences and approximate

claims amounts. With an ML system in place, the forecast for individual claims will be less error-prone and probably take less time. As a result, this can decrease the overall claims settlement time and improve customer experience.

- **Smart audit:** Using ML algorithms in claims audit improves the quality of such audits. Technology helps to identify only those claims that are indeed incorrect and need review.

2. Fraud detection

Fraudulent claims represent one of the most critical challenges in the industry. Coalition Against Insurance Fraud states that insurance fraud costs businesses \$80 billion annually. This makes insurers add these costs to premiums and increase pricing from 10 to 20% on average.

Since ML algorithms work great for anomaly detection and classification of large datasets, machine learning is a good fit for fraud detection and prevention. An ML system in insurance detects patterns and analyzes consumers' behaviors, for example, transaction methods. If it notices any abnormal activity, it warns the insurer immediately.

3. Customer Service

Customer service makes up one more interesting application of machine learning. For instance, you can use machine learning in insurance for automatic customer segmentation to get insights about customers that your marketers cannot discover by themselves. This way, an insurer doesn't have to manually analyze large datasets to seek patterns an ML model will do this for you.

Insurance companies have two ways in this case:

- Use supervised ML and alter rules and settings based on their operations
- Choose unsupervised ML and allow the model to build datasets and find patterns on its own

4. Underwriting

The use of ML-enabled risk management systems allows insurers to speed up and facilitate underwriters' work. Of course, AI and ML cannot entirely replace manual risk assessment in the insurance sector. Still, new technologies can contribute to operational efficiency and intelligent decision-making in underwriting.

FACULTY OF INFORMATICS

M.C.A. I Year II Semester Examination

Model Paper - I

MACHINE LEARNING

Time : 3 Hrs]	Max. Marks : 70
---------------	-----------------

Answer all the question according to the internal choice	(5 × 14 = 70)
--	---------------

ANSWERS

- | | | |
|----|--|----------------------|
| 1. | (a) Explain the probability rules with example problems. | (Unit-I, Q.No. 3) |
| | (b) Explain about vectors and implementation of vectors with the help of some examples. | (Unit-I, Q.No. 16) |
| | (OR) | |
| 2. | (a) Explain normal distribution with an example | (Unit-I, Q.No. 13) |
| | (b) Explain Bayesian Learning in terms of ML, MAP, Bayes Estimates and Conjugate Priors | (Unit-I, Q.No. 23) |
| 3. | (a) What is Regression Analysis? What are the various types of regressions used in regression analysis | (Unit-II, Q.No. 1) |
| | (b) Write about Dimensionality Reduction Technique. | (Unit-II, Q.No. 6) |
| | (OR) | |
| 4. | (a) What is Regularization? How does Regularization Work? | (Unit-II, Q.No. 4) |
| | (d) Explain Principle Component Analysis algorithm with example. | (Unit-II, Q.No. 9) |
| 5. | (a) Write about Logistic Regression. | (Unit-III, Q.No. 3) |
| | (b) What is Backpropagation? Explain about the working of backpropagation with example. | (Unit-III, Q.No. 11) |
| | (OR) | |
| 6. | (a) Write about Quardratic Discriminant analysis. | (Unit-III, Q.No. 5) |
| | (b) Give the brief introduction about classification and regression trees. | (Unit-III, Q.No. 16) |
| 7. | (a) What is hypothesis testing? Explain about it with help of example. | (Unit-IV, Q.No. 4) |
| | (b) Explain Hierarchical Clustering in Machine Learning. | (Unit-IV, Q.No. 4) |
| | (OR) | |
| 8. | (a) Write about the importance of AdaBoosting. | (Unit-IV, Q.No. 7) |
| | (b) What is Clustering ? Write about various types of clustering Methods and algorithms. | (Unit-IV, Q.No. 4) |

9. (a) Write about Gaussian mixture models. **(Unit-V, Q.No. 3)**
(b) Explain the uses of machine learning in manufacturing sector. **(Unit-V, Q.No. 13)**
(OR)
10. (a) What Is Machine Learning? What are the various components in machine learning architecture. **(Unit-V, Q.No. 5)**
(b) Describe the use of machine learning in retail sector. **(Unit-V, Q.No. 14)**

FACULTY OF INFORMATICS**M.C.A. I Year II Semester Examination*****Model Paper - II*****MACHINE LEARNING**

Time : 3 Hrs]

Max. Marks : 70

Answer all the question according to the internal choice**(5 × 14 = 70)****ANSWERS**

1. (a) What are Independent Events in Probability? Explain, How to calculate the probability of independent events? **(Unit-I, Q.No. 4)**
 (b) What is convex optimisation? Explain how can we solve convex optimization problem. **(Unit-I, Q.No. 21)**
 (OR)
2. (a) Explain about Random variables, and how it is used in different context. **(Unit-I, Q.No. 9)**
 (b) Write about tensors and the use of tensors in machine language. **(Unit-I, Q.No. 20)**
3. (a) Elaborate Simple and Multiple Linear Regression Models. **(Unit-II, Q.No. 3)**
 (b) What is PCA? Explain PCA algorithm **(Unit-II, Q.No. 7)**
 (OR)
4. (a) What are the various techniques used for regularization ? write about them. **(Unit-II, Q.No. 5)**
 (b) How PLS techniques is used in linear decomposition? Explain. **(Unit-II, Q.No. 9)**
5. (a) What is the Classification Algorithm in Machine Learning? Explain. **(Unit-III, Q.No. 1)**
 (b) Write about C4.5 algorithm **(Unit-III, Q.No. 15)**
 (OR)
6. (a) Explain Support Vector Machine Algorithm **(Unit-III, Q.No. 7)**
 (b) Why is it called Naïve Bayes? Explain Naïve Bayes Algorithm with example. **(Unit-III, Q.No. 18)**
7. (a) Write about different kinds of ensemble methods used in machine learning for predicting the data. **(Unit-IV, Q.No. 2)**
 (b) What is K-Means Algorithm? How does the K-Means Algorithm Work? **(Unit-IV, Q.No. 12)**
 (OR)
8. (a) Explain, how random forest algorithm works. **(Unit-IV, Q.No. 4)**
 (d) Explain DBSCAN Algorithm with example. **(Unit-IV, Q.No. 14)**

9. (a) What is known as Expectation minimization. Explain EM algorithm (Unit-V, Q.No. 1)
(b) Explain the machine learning algorithm in transport sector. (Unit-V, Q.No. 15)
- (OR)
10. (a) Explain various learning models in machine learning. (Unit-V, Q.No. 6)
(d) Discuss the machine learning algorithm in healthcare sector. (Unit-V, Q.No. 16)

FACULTY OF INFORMATICS**M.C.A. I Year II Semester Examination*****Model Paper - III*****MACHINE LEARNING**

Time : 3 Hrs]

Max. Marks : 70

Answer all the question according to the internal choice**(5 × 14 = 70)****ANSWERS**

1. (a) Define and prove Baye's theorem of probability. **(Unit-I, Q.No. 6)**
 (b) What is statistical decision theory? Explain its framework. **(Unit-I, Q.No. 22)**
 (OR)
2. (a) Write about Binomial Distribution with an example. **(Unit-I, Q.No. 11)**
 (b) What is Linear Algebra ? How many ways we can define the linear algebra functions. Explain some linear algebra functions with examples. **(Unit-I, Q.No. 15)**
3. (a) Explain about Linear Regression model. **(Unit-II, Q.No. 2)**
 (b) How does principal component analysis help in dimension reduction? **(Unit-II, Q.No. 8)**
 (OR)
4. (a) Write about Ridge and Lasso Regressions. **(Unit-II, Q.No. 5)**
 (d) Explain, how dynamically reduction technique is used for Lasso reduction. **(Unit-II, Q.No. 6)**
5. (a) How can we use LDA technique for dimensionally Reduction. **(Unit-III, Q.No. 4)**
 (b) How does the Decision Tree algorithm Work? **(Unit-III, Q.No. 13)**
 (OR)
6. (a) What are kernels? Write about how Kernels function for SVMs. **(Unit-III, Q.No. 8)**
 (d) What is the use of Bayes optimal classifier in the machine learning? **(Unit-III, Q.No. 17)**
7. (a) Explain different kinds of voting techniques used for prediction. **(Unit-IV, Q.No. 3)**
 (b) Explain K-Medoids Algorithm with example. **(Unit-IV, Q.No. 13)**
 (OR)
8. (a) What is Bagging technique? Why it is used for training data sets in ML. **(Unit-IV, Q.No. 5)**
 (b) What is the use of Gradient Boosting ? Explain. **(Unit-IV, Q.No. 8)**

9. (a) Explain, How can we present EM algorithm using probabilistic model. **(Unit-V, Q.No. 2)**
(b) Discuss the use of machine learning algorithm in weather. **(Unit-V, Q.No. 17)**
(OR)
10. (a) Explain EM algorithms for Gaussian Mixture Models. **(Unit-V, Q.No. 10)**
(b) Explain the machine learning algorithm in insurance industry. **(Unit-V, Q.No. 18)**

FACULTY OF INFORMATICS
M.C.A II-Semester(CBCS) Examination
December - 2021
MACHINE LEARNING

Time : 2 Hours]

[Max. Marks : 70

PART - A - ($4 \times 17^{1/2} = 70$ Marks)

Note : Answer any four Questions.

1. (a) Explain the concept of MAP.
(b) Analyze gradient descent optimization algorithm.
2. (a) Give an overview of Bayesian learning.
(b) Describe the basics of linear algebra for machine learning.
3. (a) Explain the Lasso regression method.
(b) Differentiate between linear and ridge regression.
4. (a) How to perform dimensionality reduction. Explain.
(b) Discuss how to work with multiple variable outcome.
5. (a) Describe about the logistic regression.
(b) Analyze the concept of perception.
6. (a) How are decisions taken in decision tree with an example.
(b) What are the applications of naive bayes.
7. (a) What is the need of ensemble methods brief.
(b) Illustrate K-means with an example.
8. (a) Elaborate density based hierarchical methods.
(b) Differentiate between bagging and boosting .
9. (a) How EM perform maximum likelihood estimation.
(b) Write notes on learning theory.
10. (a) Tell the working of Gaussian mixture models.
(b) Describe how to maximize rewards in reinforcement learning.

FACULTY OF INFORMATICS
MCA II - Semester Examinations,
April - 2022
MACHINE LEARNING
(Missing data, if any, may be suitably assumed)

Time : 3 Hours]**[Max. Marks : 70**

Note : Answer any five questions from the following.

All questions carry equal marks.

1. (a) Describe statistical decision theory.
(b) Illustrate joint probability with an example.
2. (a) Explain Bayes estimates.
(b) Elaborate dot product of two matrices with an example.
3. (a) Discuss linear regression method.
(b) Write about partial least squares.
4. (a) Describe the technique of ridge regression.
(b) Explain the steps of PCA.
5. (a) Give an overview of the working of LDA.
(b) Describe artificial neural networks.
6. (a) Discuss the functionality of SVM.
(b) Analyze bayes optimal classifier.
7. (a) How to perform hypothesis testing? Discuss.
(b) Describe Adaboost method.
8. (a) Tell about K-medoids in detail.
(b) Why machine learning in spectral domain? Explain.
9. (a) State the expectation maximization technique.
(b) Write notes on reinforcement learning.
10. (a) List the applications of GMMs and brief each one.
(b) Describe the graphical models for machine learning.