As Per The Revised Syllabus of
Gujarat Technological University

Fundamentals of Machine Learning

Karan P. Bhatt

For Diploma Students
Semester-IV • Information Technology

ATUL PRAKASHAN ™
GANDHI ROAD, AHMEDABAD.

As per the revised syllabus of the Gujarat Technological University (GTU),
This book is specially written for 4th Semester students of Diploma Programmes in Information Technology with

Course Title : Fundamentals of Machine Learning
(Course Code: 4341603)

Semester - IV

Fundamentals of Machine Learning

: Author :

Karan P. Bhatt
Assistant Professor
Computer Engineering Department
Vishwakarma Government Engineering College,
Ahmedabad.

First Edition : 2023-2024 **Price : ₹ 100=00**

ATUL PRAKASHAN
GANDHI ROAD, AHMEDABAD.

PUBLISHERS :

ATUL PRAKASHAN
Under Farnandis Bridge,
Gandhi Road, Ahmedabad - 380 001.
Phone : (079) 26426677, 22160475

EDITION :

First Edition : 2023-2024

PRICE : ₹ 100=00

COPYRIGHT
Copyright © 2022 by the Authors
All rights reserved. No part of this book shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without written permission from the publisher. No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained herein.

Disclaimer : Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an "as is" basis. The authors and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book.

Type Setting :

"BHAVANI GRAPHICS"
1st Floor, Ambica Complex,
Nr. Saibaba Temple,
Bala Hanuman Lane,
Gandhi Road, Ahmedabad-380 001.

Printed by :

ASHISH OFFSET
5-B, Mahalaxmi Compound,
Near Kalpana Bhumi Estate,
Opp. Rustom Mills,
Dhudheshwar, Ahmedabad.

Dedicated to

my Mother Smt. Sima Bhatt

&

my Father Shri Parimal Bhatt

PREFACE

Machine learning is a rapidly growing field that is changing the way we interact with technology. From recommendation systems that suggest products based on our past behavior to voice assistants that understand our natural language, machine learning is everywhere. The ability of computers to learn from data has opened up new possibilities in areas such as healthcare, finance, transportation, industries and many more.

This book is designed to introduce readers to the fundamentals of machine learning. It is written for diploma students who want to gain a basic understanding of the principles and techniques of this field. The book covers the foundational concepts of machine learning, including supervised and unsupervised learning, classification, clustering as well as it includes the practical approach for some problems using various machine learning tools.

I hope that this book will serve as a valuable resource for the diploma level students in learning more about machine learning. I wish that readers will find this book both informative and engaging, and I welcome feedback on how I can improve this book for future editions.

First and foremost, I would like to thank the almighty and my family members for their support and encouragement throughout the writing process. I could not have completed this book without their patience and understanding. I express my special thanks to my daughter **Dharvi** and my wife **Unnati Bhatt**. I would like to thank all the people who have given their valuable support knowingly or unknowingly.

I am grateful to the team at the publisher **Atul Prakashan** and **Mr. Hemendrabhai Shah** for their guidance and support throughout the publication process. In particular, I would like to thank our editor for his insightful comments and suggestions, and the production team for their hard work in bringing this book to fruition.

– Karan P. Bhatt

GUJARAT TECHNOLOGICAL UNIVERSITY (GTU)

Competency-focused Outcome-based Green Curriculum-2021 (COGC-2021)

Semester - IV

Course Title : Fundamentals of Machine Learning

(Course Code: 4341603)

Teaching and Examination Scheme :

Teaching Scheme (In Hours)			Credits (L + T + 2/P/2)	Examination Marks				Total Marks
				Theory Marks		Practical Marks		
L	T	P	C	CA	ESE	CA	ESE	
3	–	4	5	30	70	25	25	150

- * Out of 30 marks under the theory CA, 10 marks are for assessment of the micro-project to facilitate integration of COs and the remaining 20 marks is the average of 2 tests to be taken during the semester for the assessing the attainment of the cognitive domain UOs required for the attainment of the COs.
- **Legends: L**-Lecture; **T** – Tutorial/Teacher Guided Theory Practice; **P** -Practical; **C** – Credit, **CA** - Continuous Assessment; **ESE** -End Semester Examination.
- **Underpinning Theory :**

  The major underpinning theory is given below based on the higher level UOs of Revised Bloom's taxonomy that are formulated for development of the COs and competency. If required, more such UOs could be included by the course teacher to focus on attainment of COs and competency.

Unit	Unit Outcomes (UOs) (4 to 6 UOs at Application and above level)	Topics and Sub-topics
Unit – I Introduction to machine learning	1.1. Describe basic concept of machine learning and its applications.	1.1.1 Overview of Human Learning and Machine Learning 1.1.2 Types of Machine Learning 1.1.3 Applications of Machine Learning 1.1.4 Tools and Technology for Machine Learning
Unit – II Preparing to Model	2.1 Describe different types of Machine learning Activities. 2.2 Explain types of data and data preprocessing.	2.1.1 Machine Learning activities 2.1.2 Types of data in Machine Learning 2.1.3 Structures of data 2.1.4 Data quality and remediation 2.1.5 Data Pre-Processing • Dimensionality reduction • Feature subset selection
Unit - III Modeling and Evaluation	3.1 Selecting a machine learning model. 3.2 Train the model supervised learning. 3.3 Evaluate the prepared model.	3.1.1 Selecting Model • Predictive/Descriptive 3.2.1 Training a Model for supervised learning : • Holdout method • k-fold Cross-validation method 3.3.1 Model representation and interpretability 3.3.2 Evaluating performance of a model • Confusion Matrix 3.3.3 Improving Performance of a model

Unit– IV Supervised Learing - Classification and Regression	4.1 Describe supervised learning 4.2 Explain classification Algorithms. 4.3 Explain Regression.	4.1.1 Introduction to supervised learning 4.1.2 Classification Model 4.1.3 Learning steps 4.2.1 Classification Algorithms • k-Nearest Neighbor (KNN) • Support Vector Machine 4.3.1 Regression • Simple linear regression • Multiple linear regression • Logistic regression
Unit – V Unsupervised Learning	5.1 Explain unsupervised learning 5.2 Describe Clustering 5.3 Describe pattern finding using association rule.	5.1.1 Supervised vs. Unsupervised Learning 5.1.2 Applications of ussupervised learning 5.2.1 Clustering • K-means Clustering algorithm 5.3.1 Finding pattern using Association Rule • Apriori algorithm
Unit-VI Python libraries for Machine learning	6.1 Explain the use of existing machine learning python libraries.	6.1.1 Pandas 6.1.2 Numpy 6.1.3 Matplotlib 6.1.4 Scikit-learn

- **Suggested Specification Table for Question Paper Design :**

Unit No.	Unit Title	Teaching Hours	Distribution of Theory Marks			
			R Level	U Level	A Level	Total Marks
I	Introduction to machine learning	04	02	04	02	8
II	Preparing to Model	06	02	06	04	12
III	Modeling and Evaluation	08	02	04	06	12
IV	Supervised Learning - Classification and Regression	10	02	04	08	14
V	Unsupervised Learning	08	02	06	04	12
VI	Python libraries for Machine learning	06	04	04	04	12
	Total	**42**	**14**	**28**	**28**	**70**

PO-COMPETENCY-CO MAPPING :

Semester - IV	Fundamentals of Machine Learning (Course Code: 4341603)						
	POs and PSOs						
Competency & Course Outcomes	PO 1 Basic & Discipline specific knowledge	PO 2 Problem Analysis	PO 3 Design/ development of solution	PO 4 Engineering Tools, Experimentation & Testing	PO 5 Engineering Practics for society, sustainability & environment	PO 6 Project Management	PO 7 Life long learning
Competency Develop a machine learning model to solve real world problem							
Course Outcomes CO a) To understand the need of machine learning for various problem solving.	3	2	3	2	–	–	3
CO b) Prepare machine learning model and learning the evaluation methods.	3	2	3	2	–	2	3
CO c) Evaluate various supervised learning algorithms using appropriate dataset	3	3	3	3	–	3	3
CO d) Evaluate various unsupervised learning algorithms using appropriate dataset	3	3	3	3	–	3	3
CO e) To understand the use of various existing machine learning libraries	3	2	3	3	–	–	3

Contents

Unit-1 : Introduction to Machine Learning 1-11

1.1 Overview of Human Learning and Machine Learning 1
1.2 Types of Machine Learning 4
1.3 Applications of Machine Learning 7
1.4 Tools and Technology for Machine Learning 9
■ **Question Bank** 10

Unit-2 : Preparing to model 12-22

2.1 Machine Learning activities 12
2.2 Types of data in Machine Learning 13
2.3 Structures of data 14
2.4 Data quality and remediation 17
2.5 Data Pre-Processing 18
2.5.1 Dimensionality reduction 19
2.5.2 Feature subset selection 20
■ **Question Bank** 21

Unit-3 : Modeling and evaluation 23-37

3.1 Selecting a Model (Predictive/Descriptive) 23
3.2 Training a Model for supervised learning: 24
3.2.1 Holdout method 24
3.2.2 K-fold Cross-validation method 25
3.3 Model representation and interpretability 27
3.4 Evaluating performance of a model 29
3.4.1 Confusion Matrix 29
3.5 Improving Performance of a model 31
■ **Question Bank** 35

Unit-4 : Supervised Learning: Classification and Regression 38-50

4.1 Introduction to supervised learning 38
4.2 Classification Model 38
4.3 Learning Steps 40
4.4 Classification Algorithms 41
4.4.1 k-Nearest Neighbor (kNN) 41
4.4.2 Support Vector Machines 43
4.5 Regression 44
4.5.1 Simple linear regression 44
4.5.2 Multiple linear regression 46
4.5.3 Logistic Regression 47
■ **Question Bank** 49

Unit-5 : Unsupervised Learning **51-62**

5.1 Supervised vs. Unsupervised Learning 51
5.2 Applications of unsupervised learning 52
5.3 Clustering 53
5.3.1 K-means Clustering algorithm 54
5.4 Finding pattern using Association Rule 57
5.4.1 Apriori algorithm 57
■ **Question Bank** 61

Unit-6 : Python libraries for Machine Learning **63-73**

6.1 Pandas 63
6.2 Numpy 65
6.3 Matplotlib 66
6.4 Scikit-learn 69
■ **Question Bank** 72

• **Lab Manual for Machine Learning** **74-107**

PRACTICAL : 1 75
PRACTICAL : 2 81
PRACTICAL : 3 84
PRACTICAL : 4 88
PRACTICAL : 5 90
PRACTICAL : 6 91
PRACTICAL : 7 93
PRACTICAL : 8 95
PRACTICAL : 9 96
PRACTICAL : 10 98
PRACTICAL : 11 101
PRACTICAL : 12 103

• **ABBREVIATION** **108**

• **Model Paper for Mid Semester Exam** **109-110**

PAPER : 1 109
PAPER : 2 109
PAPER : 3 110

• **Model Paper for Semester Annual Exam** **111-116**

PAPER : 1 111
PAPER : 2 113
PAPER : 3 115

❖ ❖ ❖

1 Introduction to Machine Learning

1.1 Overview of Human Learning and Machine Learning

1.1.1 Human Learning :

1.1.2 Types of Human Learning:

1.1.3 Machine Learning:

1.1.4 How do machines learn?

1.1.5 Well-posed learning problem

1.2 Types of Machine Learning

1.2.1 Supervised Learning

1.2.2 Unsupervised Learning

1.2.3 Reinforcement Learning

1.2.4 Difference between Supervised, unsupervised and reinforcement learning

1.3 Applications of Machine Learning

1.4 Tools and Technology for Machine Learning

■ Question Bank

1.1 Overview of Human Learning and Machine Learning

1.1.1 Human Learning :

Human Learning is the process of gaining information through observations.

In our day to day life, we perform several activities. To do the activity in a proper way, we must have proper information related to that activity. Also, we always try to improve our activity by gaining more information or by previous experience related to that one. So, with more learning, we can make our activity more efficient.

1.1.2 Types of Human Learning:

1. Learning under expert guidance

A kid shows her head when we asked, "Where is your head?" Because, this thing she learns from her parents.

Guided learning is the process of gaining information from a person who has sufficient knowledge due to the past experience.

2. Learning guided by knowledge gained from experts

A kid can group together only those toys which belong to the group of "cars". Here, the kid has the only knowledge regarding which toy is a car. No one taught her to make a group of cars.

It is some past knowledge shared on some different way which is used as a learning to make decisions.

3. Learning by self

The kids learn to walk through obstacles is the appropriate example for learning by self. Using past experience, observations; one can learn by self.

It is direct learning.

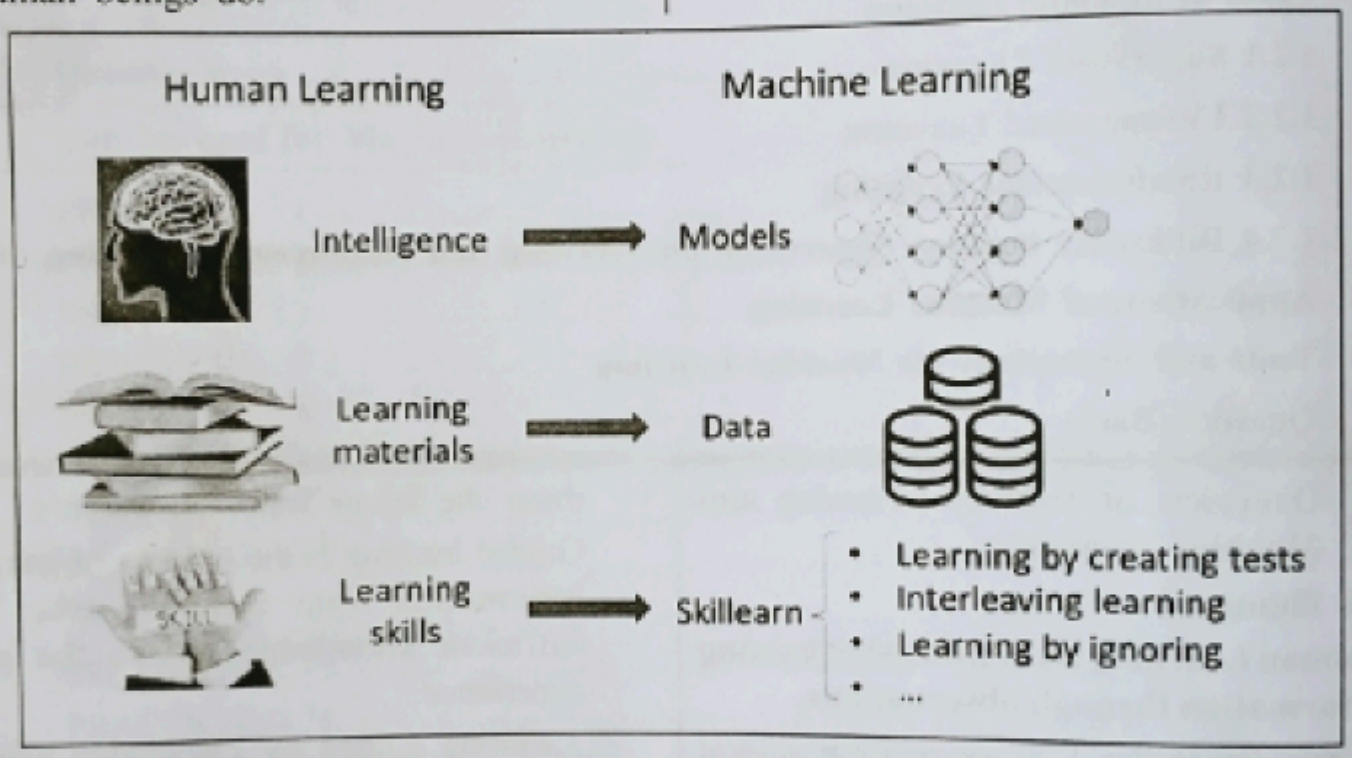
1.1.3 Machine Learning:

Machine Learning (ML) is basically that field of computer science with the help of which computer systems can provide sense to data in much the same way as human beings do.

Machine learning is a subfield of Artificial Intelligence (AI) which extracts patterns out of raw data by using an appropriate algorithm.

Its goal is to enable computers to learn on their own from experience.

A machine's learning algorithm enables it to identify patterns in observed data, build models that explain the world, and predict things without having explicit pre-programmed rules and models.



Machine Learning (ML) is a field of AI consisting of learning algorithms that improve their performance (P) at executing some task (T) over time with experience (E).

As per Tom M. Mitchell, Machine Learning is defined as , "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

For example, **Robot Driving in a maze.**

- Class of task: To reach at the end point
- Performance Measure: Time taken to reach at the end point
- Source of experience: Navigating the maze from start to end by the robot

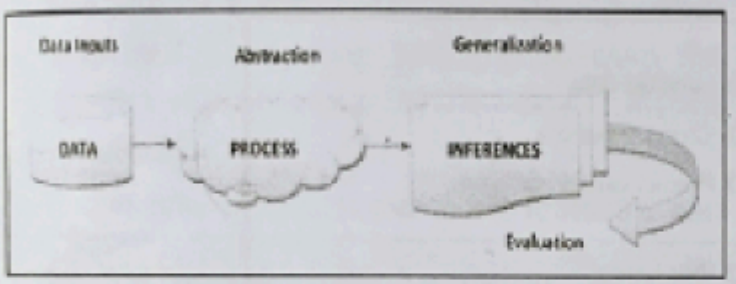
Remember that,

Task (T): It is a real world problem. Classification, clustering, regression are the examples of ML based task.

Experience (E): It is the knowledge gained from data points provided to the algorithm or model. The experience gained by out ML model or algorithm will be used to solve the task T.

Performance (P): The measure which tells whether ML algorithm is performing as per expectation or not is its performance. An ML algorithm is supposed to perform task and gain experience with the defined time period.

1.1.4 How do machines learn ?



The basic machine learning process can be divided in four parts:

1. **Data Inputs**:

   Past data is used as a base for future prediction.

   The vast pool of knowledge is available from the input data. The machine can use the past data for decision-making process.

2. **Abstraction**:

   The input data is represented in a broader way using appropriate algorithms.

Abstraction helps to derive a model which is a summarized knowledge representation of the raw data. The model may be a mathematical equation, rule-based computational blocks, logical combinations of same observations or well-defined data structures.

The decision related to choice of a model is based on the type of a problem, nature of input data, domain of the problem, and many more.

In this part, we get the finite set of data which is known as **training data** set.

Once model is chosen then the next step is to fit the model based on the input received.

3. **Generalization:**

   The abstracted representation is generalized to create a framework for making decisions.

Generalization is the most critical part in ML process.

In this part, the abstracted knowledge will be tuned up to a form which can be used to take decisions.

We want to apply the model to take decision on a set of unknown data which is called **test data** set. For decision-making, approximate/heuristic approach has to be taken.

4. **Evaluation:**

   Evaluation is the process of giving feedback to the user to measure the utility of the learned knowledge. This feedback is then utilized to effect improvements in the whole learning process.

1.1.5 Well-posed learning problem

A (machine learning) problem is well-posed if a solution to it exists, if that solution is unique, and if that solution depends on the data / experience but it is not sensitive to (reasonably small) changes in the data / experience.

To have a well-defined learning problem, three features needs to be identified:

1. The class of tasks (T)
2. The measure of performance to be improved (P)

3. The source of experience (E)

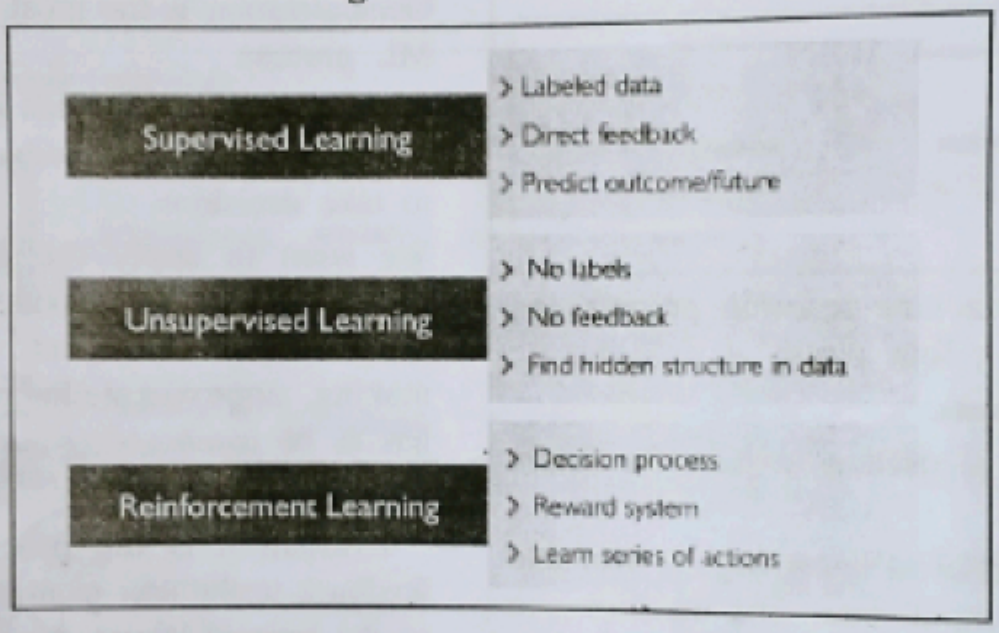
Example, **Face Recognition Problem**

- Class of Task: predicting different types of faces
- Performance Measure: able to predict maximum types of faces
- Source of Experience: training machine with maximum amount of datasets of different face images

To solve well-posed problem using Machine Learning, a simple framework this involves three questions to answer:

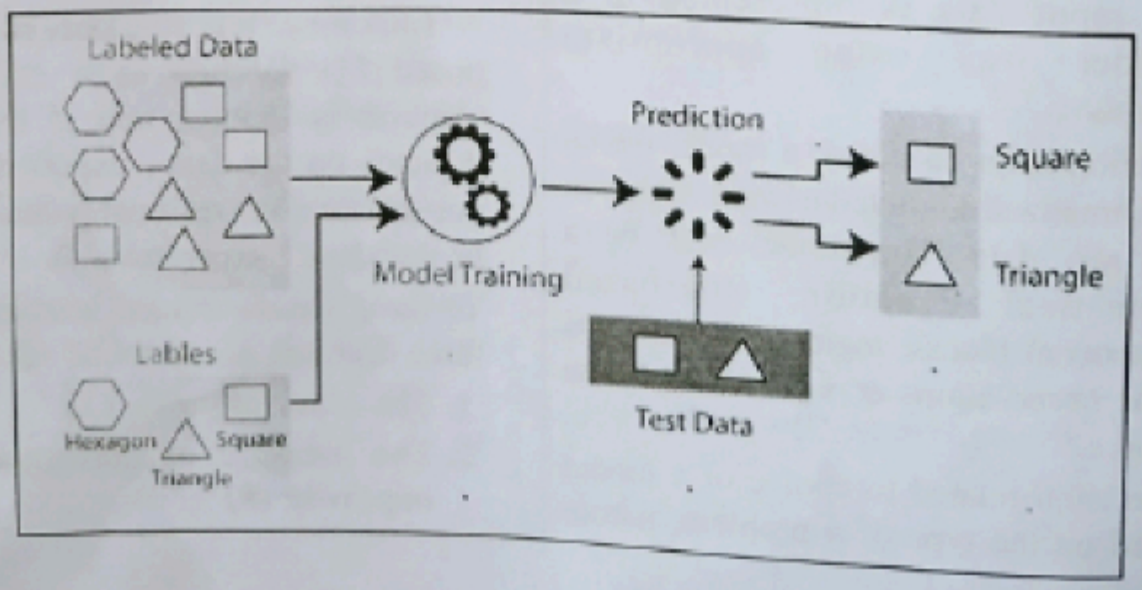
1. What is the problem? (Formalism the problem based on Task, Experience and Performance)
2. Why does the problem need to be solved? (Motivation and benefits to solve the problem)
3. How to solve the problem? (Creating the model/algorithm using knowledge)

1.2 Types of Machine Learning



1.2.1 Supervised Learning

Supervised learning is the types of machine learning in which machines are trained using well "labelled" training data, and on basis of that data, machines predict the output.



The labelled data means some input data is already tagged with the correct output. In supervised learning, models are trained using labelled dataset, where the model learns about each type of data. Once the training process is completed, the model is tested on the basis of test data and then it predicts the output.

Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to map the input variable(x) with the output variable(y).

In the real-world, supervised learning can be used for **Risk Assessment, image classification, Fraud Detection, spam filtering, etc.**

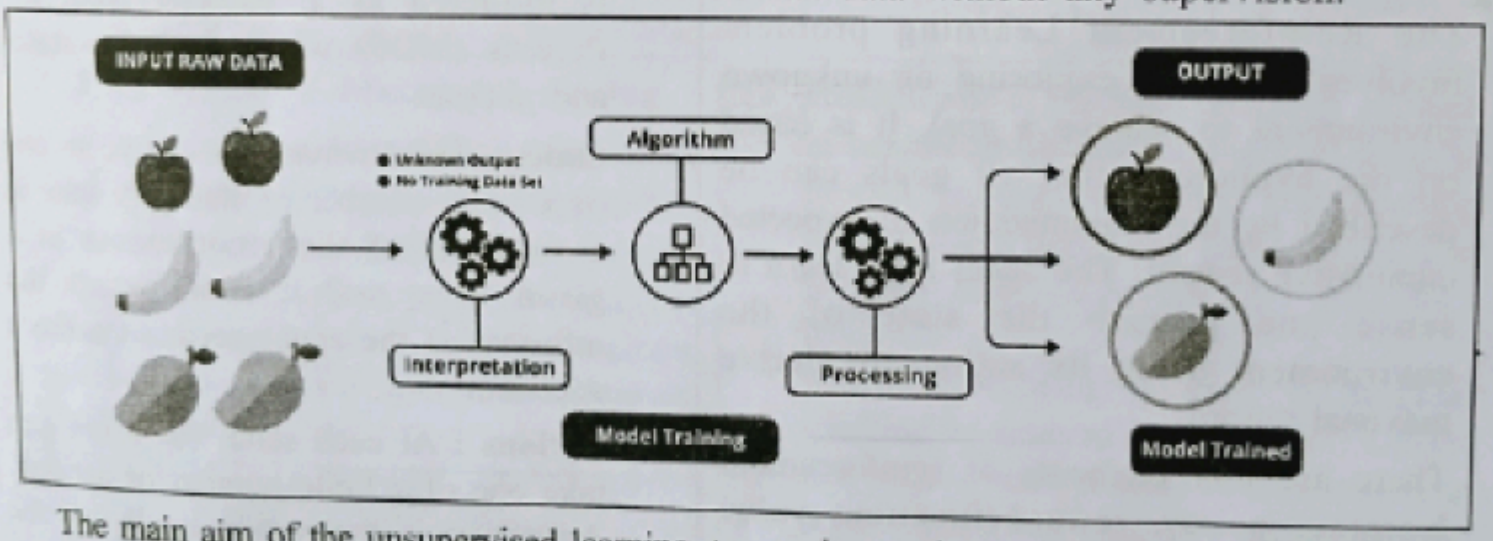
Regression and classification are two main areas of Supervised Learning.

Regression algorithms are used if there is a relationship between the input variable and the output variable. It is used for the prediction of continuous variables, such as Weather forecasting, Market Trends, etc.

Classification algorithms are used when the output variable is categorical, which means there are two classes such as Yes-No, Male-Female, True-false, etc.

1.2.2 Unsupervised Learning

Unsupervised learning is a type of machine learning in which models are trained using unlabeled dataset and are allowed to act on that data without any supervision.



The main aim of the unsupervised learning algorithm is to group or categories the unsorted dataset according to the similarities, patterns, and differences.

In the real world, unsupervised learning can be used for pattern recognition, inventory management, data exploration, recommender system, etc.

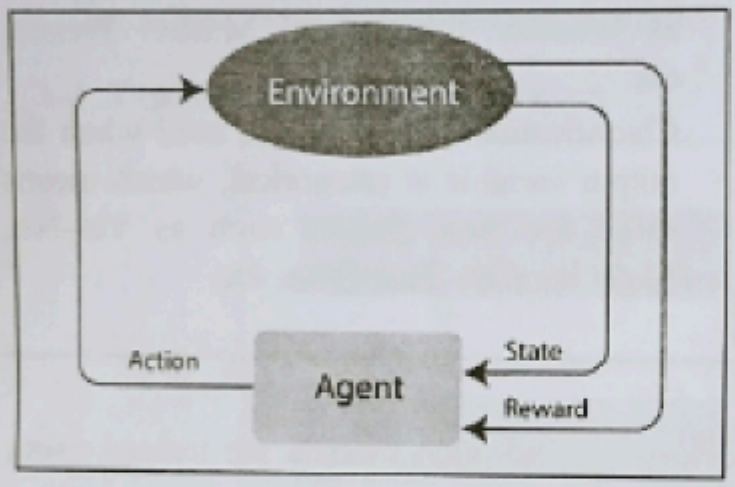
Unsupervised learning is classified into two categories of algorithms : Clustering and association rule.

- **Clustering**: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
- **Association**: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

1.2.3 Reinforcement Learning :

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.

In general, a reinforcement learning agent is able to perceive and interpret its environment, take actions and learn through trial and error.



The Reinforcement Learning problem involves an agent exploring an unknown environment to achieve a goal. It is based on the hypothesis that all goals can be described by the maximization of expected cumulative reward. The agent must learn to sense and perturb the state of the environment using its actions to derive maximal reward.

There are four elements of reinforcement learning: policy, reward function, value function and model of the environment.

1. **Policy :** A set of rules or strategies that the agent uses to decide what action to take in a particular state. The policy is updated based on the rewards received by the agent.
2. **Reward Function :** A reward function is a function that provides a numerical score based on the state of the environment. If it makes the right decision, it gets a reward. If it makes mistakes, it receives a penalty. A reward is a positive reinforcement that the agent receives for taking an action that brings it closer to its goal. A penalty is a negative reinforcement that the agent receives for taking an action that moves it further away from its goal.
3. **Value function :** A function that estimates the expected cumulative reward that the agent will receive if it follows a particular policy from a particular state. The value function is used to evaluate different policies and select the best one.
4. **Model of the environment :** In reinforcement learning, the environment is modeled as a Markov Decision Process (MDP) which includes states and actions.

   States : The environment can be in one of a finite number of states. A state is a snapshot of the environment at a given time, and it contains all the information the agent needs to make a decision.

   Actions : At each state, the agent can take one of a finite number of actions. An action is a decision made by the agent that causes the environment to transition to a new state.

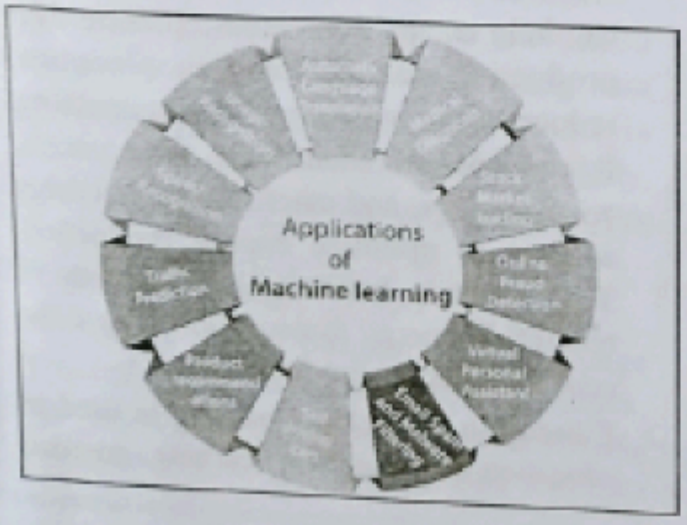
   In the real world, reinforcement learning can be used for self driving cars, gaming, robotics, etc.

1.2.4 Difference between Supervised Learning, Unsupervised Learning and Reinforcement Learning

Criteria	Supervised ML	Unsupervised ML	Reinforcement ML
Definition	Machine Learns by using labelled data	Machine is trained using unlabelled data without any guidance.	Agent interacts with the environment by performing action. Learns by errors and rewards.
Type of data	Labelled data	Unlabelled data	No – predefined data.
Type of problems	Regression and classification	Association and Clustering	Reward and error based.
Supervision	External supervision	No supervision	No supervision
Algorithms	Linear Regression, Logistic Regression, Naive Byes Decision trees	K – Means clustering, KNN (K-nearest neighbours) Principle Component Analysis Neural Networks	Monte Carlo, Q-Learning, SARSA
Aim	Calculate outcomes	Discover underlying patterns	Learn a series of action
Approach	Maps labelled inputs to the known outputs	Understands patterns & discover the output	Follow the trial and error method
Application	Risk Evaluation, Forecast Sales	Recommendation System, Anomaly Detection	Self-Driving Cars, Gaming, Healthcare

1.3 Applications of Machine Learning

Machine learning approach is adopted in multiple forms in every business domain. The major areas are Banking and finance, Insurance, Healthcare, e-Commerce, Marketing and sales, Search engines, Social Media, Virtual assistant, Transport, Security, etc.



- **Healthcare :** Machine learning is used in healthcare to analyze medical images, predict patient outcomes, and develop personalized treatment plans. It is also used in drug discovery and genomics research.

  Machine learning is used in medical imaging to analyze images from X-rays, CT scans, and MRIs. Algorithms can detect patterns and anomalies in the images, which can help with diagnosis and treatment planning.

  Machine learning can be used to analyze patient data and develop personalized treatment plans based on factors such as genetics, lifestyle, and medical history.

- **Banking and Finance :** Machine learning is used in banking and finance for fraud detection, risk assessment, credit scoring, and investment analysis.

Machine learning is used to analyze large datasets to identify fraudulent transactions. Algorithms can detect patterns and anomalies in transaction data, which can help prevent financial crimes such as credit card fraud and money laundering.

Machine learning is used to assess credit risk when evaluating loan applications. Algorithms can analyze a range of data points, such as credit history, income, and employment status, to determine the likelihood of default.

- **Insurance** : Machine learning (ML) is increasingly being used in the insurance industry to improve risk assessment, automate underwriting, and enhance the customer experience.

  Machine learning is used to automate claims processing. Algorithms can analyze claims data and make decisions about coverage, processing claims more quickly and accurately than human adjusters.

- **Marketing** : Machine learning is used in marketing to personalize advertising, improve customer segmentation, and predict customer behavior.

  Machine learning algorithms can analyze customer data to identify patterns and segments, allowing marketers to create targeted campaigns that resonate with specific customer groups. It can analyze customer data to create personalized recommendations and content. This can help improve customer engagement and loyalty.

- **Manufacturing** : Machine learning is used in manufacturing to optimize production processes, predict equipment failure, and improve quality control. It is the heart of Industry 4.0.

  Machine learning algorithms can analyze data from sensors and other sources to predict when maintenance is needed. This can help reduce downtime and improve the reliability of manufacturing equipment. ML is critical for the development of advanced robotics and automation systems. These systems can perform complex tasks and adapt to changes in the manufacturing environment.

- **Transportation** : Machine learning is used in transportation for route optimization, demand forecasting, and autonomous vehicles.

  It can analyze data on traffic patterns, road conditions, and weather to optimize routes for transportation systems. This can help reduce fuel consumption and emissions, as well as improve delivery times.

- **Energy** : Machine learning is used in the energy sector for predictive maintenance, grid optimization, and renewable energy forecasting.

  Machine learning can be used to predict energy demand based on historical data, weather patterns, and other factors. This can help energy companies optimize their production and distribution processes, reduce costs, and improve sustainability. It can analyze data from solar panels, wind turbines, and other renewable energy sources to optimize energy production. This can help improve the efficiency of renewable energy systems and reduce the cost of energy production.

- **Education** : Machine learning is used in education to personalize learning, improve

student engagement, and assess student performance.

Machine learning algorithms can analyze student data to identify their strengths, weaknesses, and learning preferences. This information can be used to create personalized learning experiences that are tailored to the needs of each student. It can be used to create intelligent tutoring systems that provide personalized feedback and guidance to students. These systems can adapt to the student's pace of learning and provide additional support when needed.

- **Agriculture :** Machine learning is used in agriculture for crop yield prediction, disease detection, and soil analysis.

1.4 Tools and Technology for Machine Learning

1. Python

Python is a high-level, general-purpose, object-oriented programming language. **It has a huge number of libraries and frameworks :** The Python language comes with many libraries and frameworks that make coding easy. This also saves a significant amount of time.

The most popular libraries are NumPy, which is used for scientific calculations; SciPy for more advanced computations; and scikit, for learning data mining and data analysis. Python is consistent and is anchored on simplicity, which makes it most appropriate for machine learning.

2. R Programming

R is a procedural programming language created by statisticians for statistics, specifically for working with data. It is a language for statistical computing and data visualizations used widely by business analysts, data analysts, data scientists, and scientists.

The R language has become popular because it lets researchers easily combine different machine learning techniques into a single program.

3. MATLAB

MATLAB (matrix laboratory) makes machine learning easy. With tools and functions for handling big data, as well as apps to make machine learning accessible, MATLAB is an ideal environment for applying machine learning to your data analytics.

In MATLAB it takes fewer lines of code and builds a machine learning or deep learning model, without needing to be a specialist in the techniques. MATLAB provides the ideal environment for machine learning, through model training and deployment.

4. SAS

SAS (Statistical Analysis System) is an enterprise-grade statistical software suite developed by SAS Institute. It is used for data management, advanced analytics, and predictive analytics, and has been one of the leading analytics platforms for a long time.

The SAS software contains all the necessary features required to design, develop, train, and evaluate machine learning models.

5. Julia

Julia is a general purpose programming language well suited for numerical analysis and computational science.

It is an open-source. It has ability to implement high performance machine learning algorithms.

Question Bank

■ Short-Answer Questions (3 or 4 marks)

1. Explain human learning in brief.
2. Define Machine Learning. Also, compare it with Human Learning.
3. Explain classification and regression in brief.
4. Explain clustering and association in brief.
5. Give the difference between supervised learning and unsupervised learning.
6. Explain the concept of penalty and rewards in reinforcement learning.
7. Write a short note on machine learning in finance and banking.
8. Give the difference between Python and R.
9. Identify main three features for following well-posed problem:
   a. Fruit prediction Problem
   b. Handwriting recognition Problem

■ Long-Answer Questions (7 marks)

1. Define well-posed problem. Explain important features that are required to well-define a learning problem.
2. Compare the different types of machine learning.
3. Explain types of machine learning in detail.
4. Define machine learning. Explain any two business applications of Machine Learning in detail.
5. Explain different tools and technology used in Machine Learning.
6. Describe basic concept of Machine Learning and its application.

■ Check your knowledge :

1. What is Machine Learning ?
   a. A field of computer science b. A method of data analysis
   c. A type of algorithm **d. All of the above**
2. A type of learning where the model learns by interacting with an environment and receiving rewards or penalties is called ____.
   a. Supervised Learning b. Unsupervised Learning
   c. Reinforcement Learning d. Machine Learning

3. Which type of machine learning involves no target variable and the model learns by finding patterns in the data ?

   a. Supervised Learning **b. Unsupervised Learning**

   c. Reinforcement Learning d. Machine Learning

4. Which type of machine learning involves a target variable that is used to train the model ?

   a. Supervised Learning b. Unsupervised Learning

   c. Reinforcement Learning d. Machine Learning

5. Which of the following is an application of machine learning in transportation?

   a. Face recognition **b. Predictive maintenance**

   c. Sentiment analysis d. Language translation

■

2 Preparing to Model

2.1 Machine Learning Activities

2.2 Types of Data in Machine Learning

2.2.1 Qualitative Data:

2.2.2 Quantitative Data:

2.3 Structure of Data

2.3.1 Exploring numerical data Understanding Measure of Central Tendency

2.3.2 Plotting Numerical data

2.4 Data Quality and Remediation

2.4.1 Data Quality

2.4.2 Data Remediation

2.5 Data Pre-processing

2.5.1 Dimensionality reduction

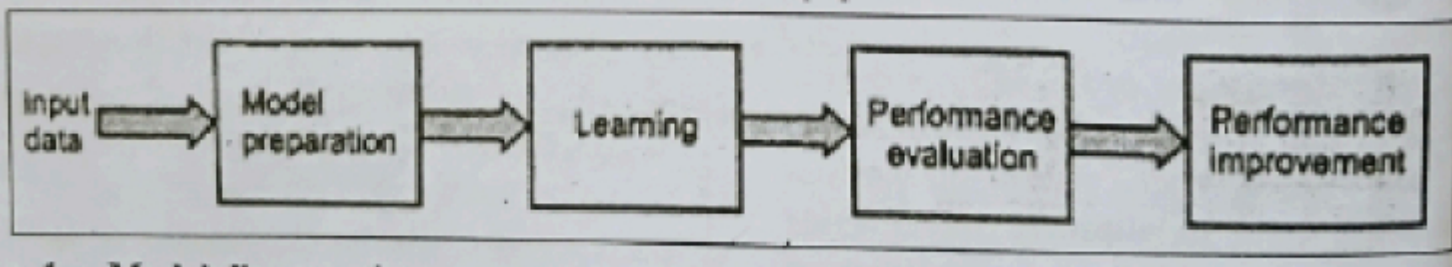
2.5.2 Feature Subset Selection

■ Question Bank

2.1 Machine Learning Activities

In machine learning, the detailed view of understanding the nature of input data is very important. Using the detailed view of input data, we can select the ML model and apply it on the data.

The machine learning activities have the four-step process.



Step 1 : Model Preparation

In this step, activities like understanding the type, nature and quality of data are involve. Inter-feature relationship between data, data pre-processing and finding any issues in data have been involved through Step 1.

Step 2 : Learning

In this step, activities like data partitioning, model selection and cross-validation are involved.

If it is supervised learning approach then the data is divided into two parts: Training

Data and Test Data in Step-2 and then the appropriate model will be applied.

If it is unsupervised learning approach then the unsupervised learning model will be directly applied to the input data.

Step 3: Performance Evaluation

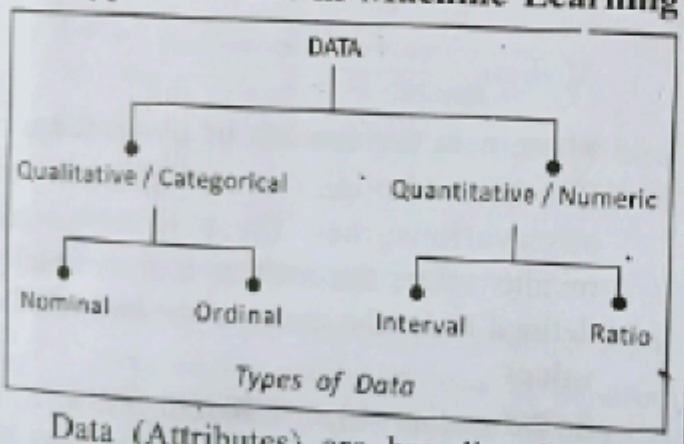
In this step, the performance of the model is evaluated. If it is a classification then the performance will be examined using confusion matrix. Also, performance tade-offs should be visualized using ROC curves.

Evaluation will be done using different criteria like accuracy, F-Score, etc.

Step 4: Performance Improvement

This is the last step of Machine Learning Process. In this step, tuning the model, bagging, boosting activities are involved to improve the performance of the model.

2.2 Types of Data in Machine Learning



Data (Attributes) are broadly classified into Qualitative Data and Quantitative Data.

2.2.1 Qualitative Data : (04)

The data collected on grounds of categorical variables are **qualitative data**. It is also known as categorical data.

Qualitative data are more descriptive and conceptual in nature.

It measures the data on basis of the type of data, collection, or category.

Qualitative data talks about the experience or quality and explains the questions like 'why' and 'how'.

For example, if we consider the performance of employee in terms of Good, Average or Poor then it is called qualitative data.

The qualitative data are subdivided into two parts: Nominal Data and Ordinal Data.

Nominal data is the data which has no numeric value, but a named value.

Example : Nationality: Indian, American, British, etc.

On nominal data, mathematical functions like addition, multiplication or statistical functions like mean, median cannot be applied. Only basic count should be applied. So we can get mode of the nominal data.

Ordinal data is the data which has named value which can be naturally ordered in increasing or decreasing.

Example: Students Satisfaction: Very Happy, Happy, Unhappy

On ordinal data, the ordered data is available so we can perform mode, median and quartiles operations. But, we cannot perform operations like mean.

2.2.2 Quantitative Data :

The data collected on the grounds of the numerical variables are **quantitative data**. It is also known as numeric data.

Quantitative data are more objective and conclusive in nature.

It measures the values and is expressed in numbers. The data collection is based on "how much" is the quantity.

Quantitative data talks about the quantity and explains the questions like 'how much', 'how many'.

For example, if we consider marks of Mid Semester Exam like '30', '25', '12' then it is called quantitative data.

The quantitative data are subdivided into two parts: Interval Data and Ratio Data.

Interval Data is numeric data for which exact difference between values is known.

Such data do not have something called a 'true zero' value.

Example: Celsius Temperature, Date, Time

Ratio Data is numeric data for which exact value can be measured.

Such data have the 'true zero' value, too.

Example: Height, Weight, Age, Marks

2.3 Structure of Data

2.3.1.1 Exploring numerical data Understanding Measure of Central Tendency

The central tendency measure is defined as the number used to represent the center or middle of a set of data values. The three commonly used measures of central tendency are the mean, median, mode.

The purpose of the central tendency is to provide an exact representation of the entire collected data. It is often defined as the single value that is representative of the data.

Mean: In statistics, arithmetic mean is the average of the given set of numbers or observations.

$$A = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i$$

Where,

A is the arithmetic mean.

n is the number of items or numbers.

x_i is the value of every individual item being averaged

Example: The mean of first 10 natural numbers are

Arithmetic mean = Sum of all values / Total number of values

$= (1+2+3+4+5+6+7+8+9+10) / 10$

$= 55 / 10$

$= 5.5$

Median: Median is the middle value of the given list of data when arranged in an order. The arrangement of data or observations can be made either in ascending order or descending order.

If the total number of observations given is odd, then the formula to calculate the median is:

$$Median = \left(\frac{n+1}{2}\right)^{th} term$$

where n is the number of observations.

For example, the median of 2, 4, 7, 9, 12 is 7.

If the total number of observation is even, then the median formula is:

$$Median = \frac{\left(\frac{n}{2}\right)^{th} term + \left(\frac{n}{2}+1\right)^{th} term}{2}$$

where n is the number of observations

If there is an even number of observations, then there is no single middle value; the median is then usually defined to be the mean of the two middle values.

So the median of 3, 6, 8, 10 is (6+8)/2 = 7.

Mode: The mode is the value that is repeatedly occurring in a given set.

For example, in the data set 2, 4, 5, 5, 6, 8, the mode of the data set is 5 since it has appeared in the set twice.

When there are two modes in a data set, then the set is called bimodal.

When there are three modes in a data set, then the set is called trimodal.

When there are four or more modes in a data set, then the set is called multimodal

Understanding Measure of Dispersion

Measures of dispersion are statistical measures that describe how spread out the data is around the central tendency measures. They provide information on the variability or diversity of the dataset.

The four commonly used measures of dispersion are the range, variance, standard deviation and Interquartile range (IQR).

Range: The range is the difference between the largest and smallest values in a dataset. It provides a simple measure of how spread out the data is.

For example, suppose we have the following dataset:

5, 6, 8, 10, 12, 14, 20

The largest value is 20 and the smallest value is 5.

Therefore, the range is:

Range = 20 - 5 = 15

So, the range of this dataset is 15.

Variance : Variance is a measure of how data points vary from the mean. It is denoted as 'σ^2'.

Properties of Variance:

- It is always non-negative since each term in the variance sum is squared and therefore the result is either positive or zero.
- Variance always has squared units.

The variance formula is given by:

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Here,

s^2 = Sample variance

n = Number of observations in sample

x_i = i^{th} observation in the sample

$\bar{x}$ = Sample mean

Standard Deviation:

Standard deviation is the measure of the distribution of statistical data. It is denoted as 'σ'.

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Here,

s = Population standard deviation

Properties of Standard Deviation

- It describes the square root of the mean of the squares of all values in a data set and is also called the root-mean-square deviation.
- The smallest value of the standard deviation is 0 since it cannot be negative.
- When the data values of a group are similar, then the standard deviation will be very low or close to zero. But when the data values vary with each other, then the standard variation is high or far from zero.

Example : If a die is rolled, then find the variance and standard deviation of the possibilities.

When a die is rolled, the possible outcome will be 6. So the sample space, n = 6 and the data set = { 1;2;3;4;5;6}.

To find the variance, first, we need to calculate the mean of the data set.

Mean, $\bar{x} = (1+2+3+4+5+6)/6 = 3.5$

We can put the value of data and mean in the formula to get;

$S^2 = \Sigma\ (xi - \bar{x})^2/n$

$S^2 = \frac{1}{6}\ (6.25+2.25+0.25+0.25+2.25+6.25)$

$S^2 = 2.917$

Now, the standard deviation,

$S = \sqrt{2.917} = 1.708$

Interquartile range (IQR) :

The IQR is the range between the first quartile (25th percentile) and third quartile (75th percentile) of a dataset. It provides a measure of the spread of the middle 50% of the data. For example, suppose we have the following dataset:

4, 5, 6, 8, 10, 12, 14, 18, 20

To calculate the IQR, we first need to find the median, which is:

Median = (10 + 12) / 2 = 11

Next, we need to find the first and third quartiles. The first quartile (Q1) is the median of the lower half of the dataset, and the third quartile (Q3) is the median of the upper half of the dataset. In this case, we have:

Lower half :

4, 5, 6, 8 Upper half: 12, 14, 18, 20

Q1 = (5+6)/2 = 5.5 Q3 = (14+18)/2 = 16

The IQR is then :

IQR = Q3 - Q1 = 16 - 5.5 = 10.5

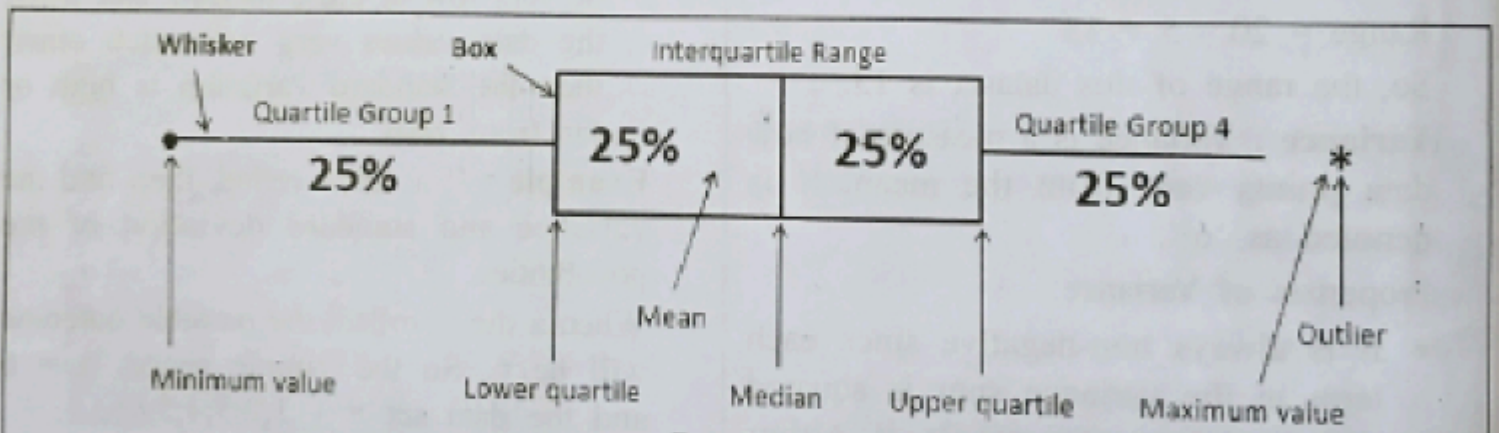
So, the IQR of this dataset is 10.5. This tells us that the middle 50% of the data is spread out over a range of 10.5.

2.3.2 Plotting Numerical data

One of the best ways to analyze any process is to plot the data on a graph or chart. Numerical data can be plotted using box plots or histograms.

Box Plots : A box plot is a graph that shows the frequency of numeric data values. It is mainly used to explore data as well as to present the data in an easy and understandable manner.

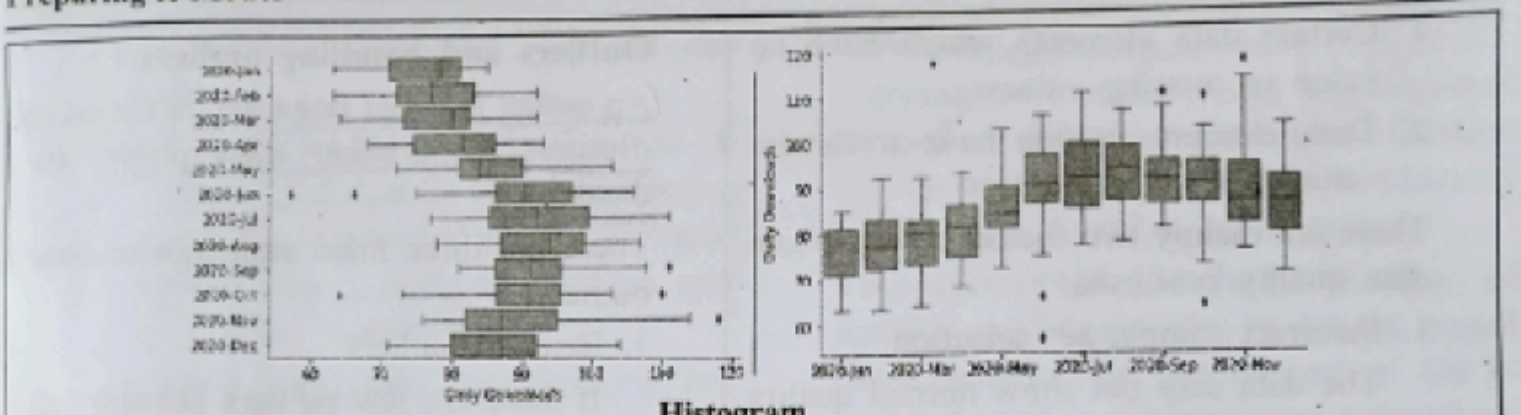
A box plot is a standardized way of displaying the dataset based on the five-number summary: the minimum, the maximum, the sample median, and the first and third quartiles.



Box Plot

- Minimum (Q_0 or 0th percentile): the lowest data point in the data set excluding any outliers
- Maximum (Q_4 or 100th percentile): the highest data point in the data set excluding any outliers
- Median (Q_2 or 50th percentile): the middle value in the data set
- First quartile (Q_1 or 25th percentile): also known as the lower quartile $q_n(0.25)$, it is the median of the lower half of the dataset.
- Third quartile (Q_3 or 75th percentile): also known as the upper quartile $q_n(0.75)$, it is the median of the upper half of the dataset.

We can also find Inter-quartile range (IQR) : the distance between the upper and lower quartiles. IQR = Q_3 - Q_1



Histogram

Histogram :

A **histogram** is a graph which shows the frequency of continuous data values. It is a two-dimensional figure.

Histograms are widely used in statistics, scientific research, economics, in social and human sciences, and it is especially important in process improvement and operational excellence.

Histograms are mainly used to explore data as well as to present the data in an easy and understandable manner. They are often used as the first step to determine the underlying probability distribution of a data set or sample.

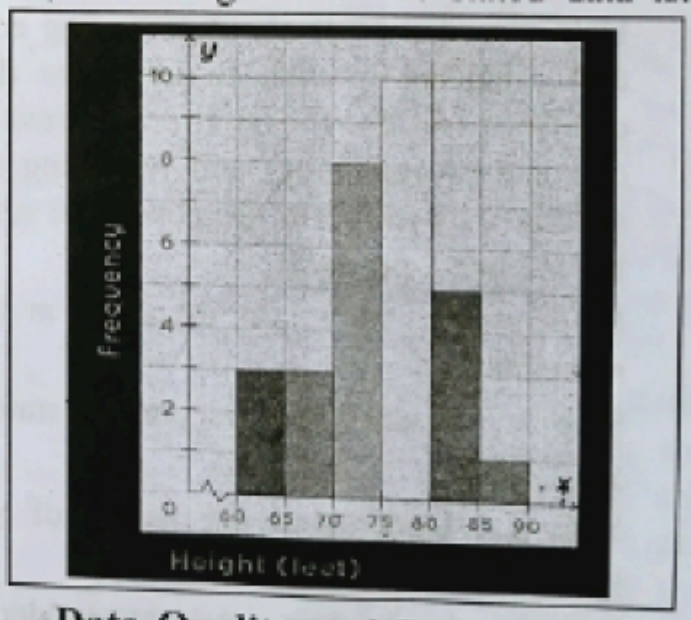
Follow the below mentioned steps to create histogram manually:

1. Collect the data set and prepare it for the analysis.
2. Draw a horizontal line and divide it into equal intervals or bins (12 intervals for example). The total width should be equal to the range of the data.
3. Draw bars above each bin to represent the frequency of the data values within each interval. The bars should be adjacent with no gaps between them.
4. Indicate the mean of the data and other important information such as the standard deviation and the specification limits.

Example : The following table shows the number of trees in a garden with different height range.

Height Range (ft)	Number of Trees (Frequency)
60-75	3
66-70	3
71-75	8
76-80	10
81-85	5
86-90	1

So, the histogram for the stated data is:



2.4 Data Quality and Remediation

2.4.1 Data Quality

Data which have the right quality helps to achieve better performance in Machine Learning Model.

Data quality problems are :

1. Certain data elements which have no value or missing values
2. Data elements which have irrelevant values known as outliers

There are mainly two factors which create data quality problems:

1. Incorrect sample set selection

   The data may not show normal quality due to incorrect selection of sample set. For example, in sales transactions, if we collect the data set of Diwali period then the prediction should be differing from our daily routine.

2. Errors in data collection

   It leads towards outliers and missing values. It may happen that a person is responsible for the wrongly collected data. It may also happen that the data is not recorded at all. So, in record it shows missing values.

2.4.2 Data Remediation

Data remediation is about correcting errors and mistakes in data to eliminate data-quality issues. It is the process of cleansing, organizing, and migrating data so that it is properly protected and serves its intended purpose.

Mainly five steps are involved in data remediation:

Step 1 : Understand the current state of your data.

Step 2 : Understand the nature of your data via classification.

Step 3 : Apply data governance. Identify outliers, if any.

Step 4 : Discuss various data remediation options and select appropriate one.

Step 5 : Implement the selected data remediation strategy.

Outliers and handling outliers :

An outlier is a data point that is remarkably distinct from other data points in a dataset.

There are three main strategies to handle outliers:

1. Remove outliers

   If there are few outliers in record then simply remove it.

2. Imputation (Replace the value)

   Impute the value with mean, median or mode.

3. Capping

   For values that lie outside the 1.5|x| IQR limits, we can replace those observations with the following conditions:

   a. If below the lower limit then replace with the value of 5th percentile

   b. If above the upper limit the replace with the value of 95th percentile

Handling missing values :

One or more elements may have missing values in multiple records of data set.

Following strategies have been used to handle missing values:

1. Ignore the tuple
2. Eliminate records having a missing value of data elements
3. Filling missing values using mean, median or mode.
4. Use a global constant to fill in the missing value:
5. Use the most probable value to fill in the missing value.

2.5 Data Pre-processing (04)

Data pre-processing is a process of preparing the raw data and making it suitable for a machine learning model.

Data Pre-processing Methods/ Techniques :

- **Data Cleaning** routines work to "clean" the data by filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies.
- **Data Integration** which combines data from multiple sources into a coherent data store, as in data warehousing.
- **Data Transformation**, the data are transformed or consolidated into forms appropriate for mining
- **Data Reduction** obtains a reduced representation of the data set that is much smaller in volume, yet produces the same (or almost the same) analytical results.

2.5.1 Dimensionality reduction

Dimensionality reduction is the data reduction technique where encoding mechanisms are used to reduce the data set size.

Dimensionality reduction, or dimension reduction, is the transformation of data from a high-dimensional space into a low-dimensional space so that the low-dimensional representation retains some meaningful properties of the original data, ideally close to its intrinsic dimension.

The number of input variables or features for a dataset is referred to as its dimensionality. Dimensionality reduction refers to techniques that reduce the number of input variables in a dataset. Avoiding overfitting is a major motivation for performing dimensionality reduction.

Example :

Dimensional reduction can be discussed through a simple e-mail classification problem, where we need to classify whether the e-mail is spam or not.

This can involve a large number of features, such as whether or not the e-mail has a generic title, the content of the e-mail, whether the e-mail uses a template, etc.

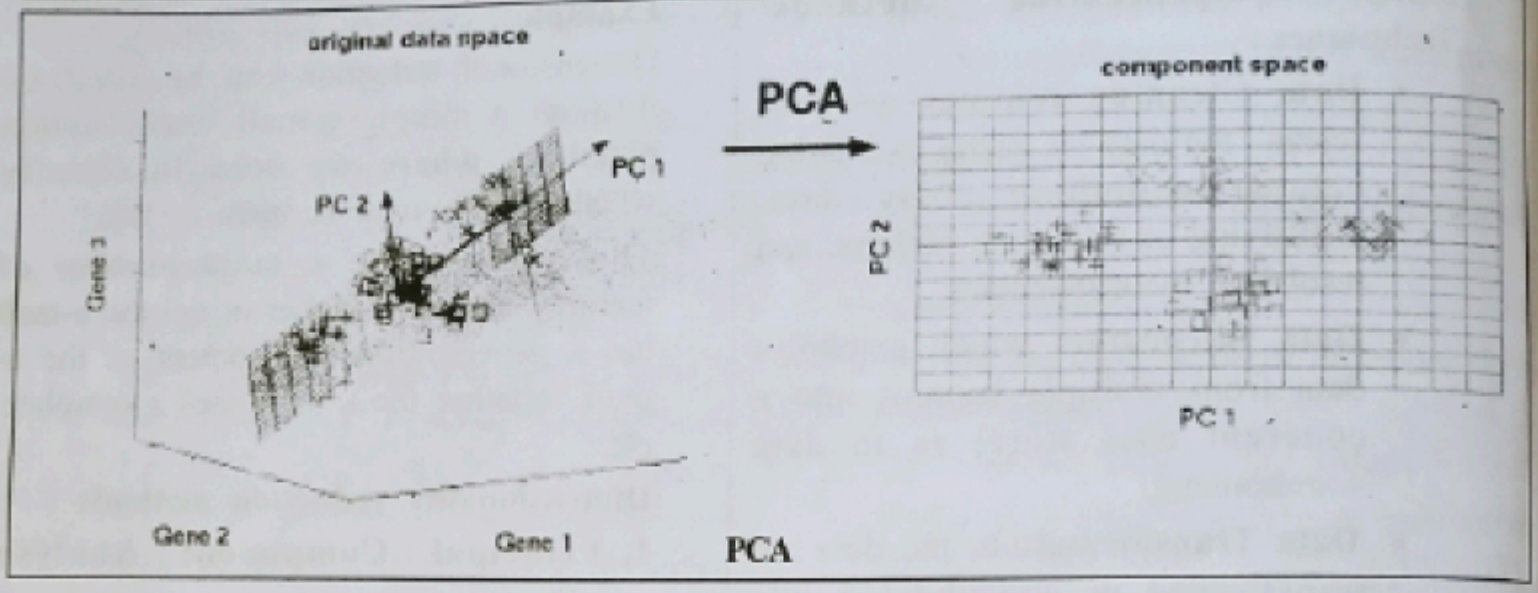
Dimensionality reduction methods :

1. Principal Component Analysis (PCA) :

Frequently used for dimensionality reduction in continuous data, PCA rotates and projects data along the direction of increasing variance. The features with the maximum variance are the principal components.

PCA is a statistical technique used to reduce the dimensionality of a large dataset.

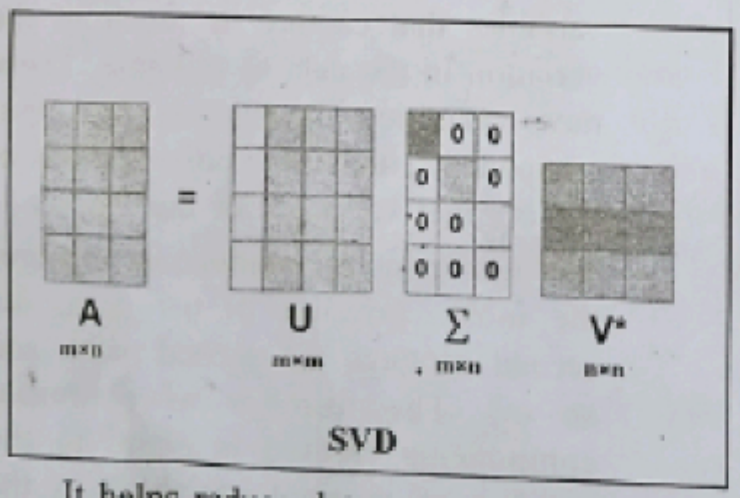
- PCA works by identifying patterns in the data and then creating new variables that capture as much of the variation in the data as possible. These new variables, known as principal components, are linear combinations of the original variables in the dataset.
- The first principal component captures the most variation in the data; the second captures the second most, and so on. The number of principal components created is equal to the number of original variables in the dataset.



PCA

PCA can be used for a variety of purposes, including data visualization, feature selection, and data compression.

2. Singular Value Decomposition (SVD): The main intuition behind Singular Value Decomposition (SVD) is, that Matrix A transforms a set of orthogonal vectors (v) to another set of orthogonal vectors (u) with a scaling factor of σ. So, σ is called the Singular Value corresponding to the respective singular vectors u and v.



SVD

It helps reduce datasets containing a large number of values. Furthermore, this method is also helpful to generate significant solutions for fewer values.

2.5.2 Feature Subset Selection

Feature Subset Selection is the method of reducing the input variable to your model by using only relevant data and getting rid of noise in data.

It is the process of automatically choosing relevant features for your machine learning model based on the type of problem you are trying to solve. We do this by including or excluding important features without changing them.

It helps in cutting down the noise in our data and reducing the size of our input data.

Feature Subset Selection is the most critical pre-processing activity in any machine learning process.

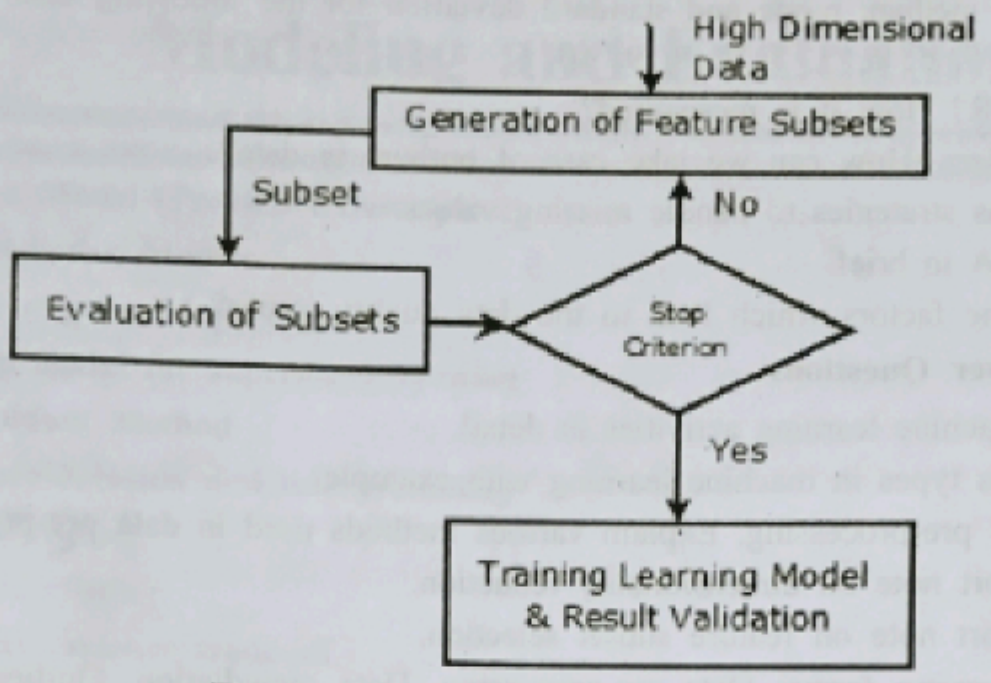
Four basic steps for feature subset selection are as follow:

Step 1: Generation of all possible subset

Step 2: Evaluation of generated subset,

Step 3: Stopping criterion

Step 4: Validation of result



Flow chart for feature subset selection

Generation of all possible subset is a brute force method that generates candidate feature subsets for estimation based on a particular search procedure.

Each generated candidate feature subset is estimated and compared with the preceding most excellent one according to a certain estimating criterion.

If the criteria score of new subset come out to be better, it replaces the previous best subset. The process of generation of subset and estimation is recurring until a given stopping criterion is not satisfied.

Then, the most excellent subset usually needs to be tested by prior knowledge or many separate tests on synthetic datasets and/or real-world datasets.

Feature subset selection models are of two types:

1. Supervised Models: Supervised feature selection refers to the method which uses the output label class for feature selection. They use the target variables to identify the variables which can increase the efficiency of the model. Filter method, wrapper method and intrinsic method are three methods used in supervised model.
2. Unsupervised Models: Unsupervised feature selection refers to the methods which not need the output labels class for feature selection. We use them for unlabelled data.

Question Bank

■ **Short-Answer Questions** **(3 or 4 marks) :**

1. State the activities involved in model preparation stage.
2. Give the difference between qualitative data and quantitative data.
3. Give the difference between histogram and box plot with example.
4. State different measures of central tendency.

5. Find mean, median, mode and standard deviation for the following data:
   1, 1, 2, 4, 5, 5, 6, 7, 7, 7, 8, 9, 10
6. What is IQR? How it is meseaured?
7. Define outliers. How can we take care of outliers in data?
8. State various strategies to handle missing values.
9. Explain PCA in brief.
10. What are the factors which lead to the data quality issues?

■ **Long-Answer Questions** **(7 marks):**

1. Describe machine learning activities in detail.
2. Explain data types in machine learning with example.
3. Define data pre-processing. Explain various methods used in data pre-processing.
4. Write a short note on dimensionality reduction.
5. Write a short note on feature subset selection.
6. Define following terms: Data pre-processing, Data remediation, Outliers, Imputation, Standard Deviation, Ratio Data, Ordinal Data
7. Explain steps to create box plot with suitable example.
8. Explain steps to create histogram with suitable example. Also, state the difference between bar-chart and histogram.

■ **Check your knowledge :**

1. Which of the following types of data is non-numerical and can be classified into distinct categories?
   a. Categorical data b. Ordinal data
   c. Interval data d. All of the above
2. A technique which is used to clean, transform and prepare raw data for analysis is called ____.
   a. Data preparing **b. Data pre-processing**
   c. Data Processing d. Data Mining
3. Which of the following is a data reduction technique ?
   a. Dimensionality reduction b. Data encoding
   c. Dimensionality Imputation d. Data Visualization
4. What is the purpose of feature selection ?
   a. To increase the size of the dataset
   b. To remove irrelevant or redundant features from the dataset
   c. To add new features to the dataset
   d. To convert numerical data into categorical data
5. Which measure of central tendency is most appropriate for nominal or categorical data?
   a) Mean b) Median **c) Mode** d) Range

■

3 Modeling and Evaluation

3.1 Selecting a Model (Predictive/Descriptive)
- 3.1.1 Predictive Models
- 3.1.2 Descriptive Models

3.2 Training a model for supervised learning
- 3.2.1 Holdout Method

3.3 Model representation and interpretability
- 3.3.1 Overfitting
- 3.3.2 Underfitting
- 3.3.3 Bias-variance trade-off

3.4 Evaluating Performance of a Model
- 3.4.1 Confusion Matrix

3.5 Improving performance of a model
- 3.5.1 Ensemble Approach
- 3.5.2 Model Parameter Tuning

■ Question Bank

Model: Structured representation of raw input data to the meaningful pattern is called a model.

Model Training: The process of fitting a specific model to a data set is called model training.

Target Function: Target function of a model is the function defining the relationship between the input (also called predictor or independent) variables and the output (also called response or dependent or target) variable.

It is represented in the general form: $Y = f(X) + e$, where Y is the output variable, X represents the input variables and 'e' is a random error term.

3.1 Selecting a Model (Predictive/ Descriptive)

Machine learning algorithms are broadly of two types:

1. Models for supervised learning, which primarily focus on solving predictive problems
2. Models for unsupervised learning, which solve descriptive problems.

3.1.1 Predictive Models

Models for supervised learning approach are Predictive models. They try to predict certain value using the values in an input data set.

The predictive models have a clear focus on what they want to learn and how they want to learn.

Predictive models may need to predict the value of a category or class to which a data instance belongs to.

Examples :

1. Predicting win/loss in a basketball match
2. Predicting whether a bank transaction is fraud

The models which are used for prediction of target features of categorical value are known as **classification models**. The target feature is known as a class and the categories to which classes are divided into are called levels. Examples of classification models are k-Nearest Neighbor (kNN), Naïve Bayes, and Decision Tree.

Predictive models may also be used to predict numerical values of the target feature based on the predictor features.

Example :

1. Prediction of financial growth in the next coming year
2. Prediction of heat wave in the summer

The models which are used for prediction of the numerical value of the target feature of a data instance are known as **regression models**. Linear Regression and Logistic Regression models are examples of regression models.

3.1.2 Descriptive Models

Models for unsupervised learning approach are Descriptive models. They are used to describe a data set or gain insight from a data set.

There is no target feature or single feature of interest in case of unsupervised learning. Based on the value of all features, interesting patterns or insights are derived about the data set.

Descriptive models which group together similar data instances are called clustering models. The most popular model for clustering is k-Means.

Examples:

1. Customer grouping based on society
2. Movie grouping based on language and age

Descriptive models related to pattern discovery is used for market basket analysis of transactional data.

Market Basket Analysis is a modeling technique based upon the theory that if you buy a certain group of items, you are more (or less) likely to buy another group of items.

For example, if you are in a store and you buy milk and don't buy bread, you are more likely to buy fruits at the same time than somebody who didn't buy bread.

The set of items a customer buys is referred to as an item set, and market basket analysis seeks to find relationships between purchases.

Typically, the relationship will be in the form of a rule like:

IF {milk, butter} THEN {bread}.

The probability that a customer will buy milk without an butter (i.e. that the antecedent is true) is referred to as the **support for the rule**.

The conditional probability that a customer will purchase bread is referred to as the **confidence**.

3.2 Training a model for supervised learning

3.2.1 Holdout Method

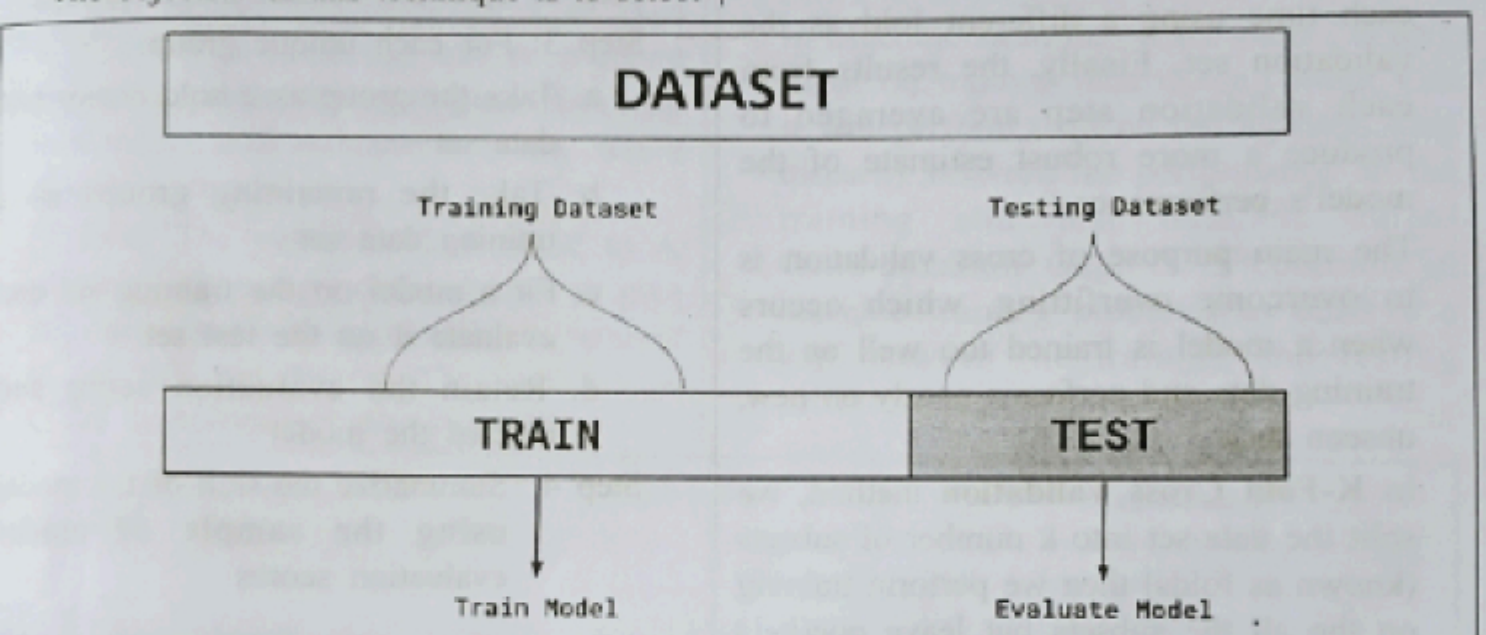
The hold-out method involves splitting the data into multiple parts and using one part for training the model and the rest for

validating and testing it. It can be used for both model evaluation and selection.

Model evaluation using the hold-out method entails splitting the dataset into training and test datasets, evaluating model performance, and determining the most optimal model.

The objective of this technique is to select the best model based on its accuracy on the testing dataset and compare it with other models. Models are trained to improve model accuracy on test datasets based on the assumption that the test dataset represents the population.

This diagram illustrates the hold-out method for model evaluation.



Holdout method

There are two parts to the dataset in the diagram above.

One split is held aside as a training set. Another set is held back for testing or evaluation of the model. The percentage of the split is determined based on the amount of training data available. A typical split of 70–30% is used in which 70% of the dataset is used for training and 30% is used for testing the model.

Follow the steps below for using the hold-out method for model evaluation:

1. Split the dataset in two (preferably 70–30%; however, the split percentage can vary and should be random).
2. Now, we train the model on the training dataset by selecting some fixed set of hyper-parameters while training the model.
3. Use the hold-out test dataset to evaluate the model.
4. Use the entire dataset to train the final model so that it can generalize better on future datasets.

In this process, the dataset is split into training and test sets, and a fixed set of hyper-parameters is used to evaluate the model.

3.2.1 K-fold cross validation method

In machine learning, we couldn't fit the model on the training data and can't say that the model will work accurately for the real data.

For this, we have to assure that our model got the correct patterns from the data, and it is not getting up too much noise. For this purpose, we use the cross-validation technique.

Cross validation is a technique used in machine learning to evaluate the performance of a model on unseen data. It involves dividing the available data into multiple folds or subsets, using one of these folds as a validation set, and training the model on the remaining folds.

This process is repeated multiple times, each time using a different fold as the validation set. Finally, the results from each validation step are averaged to produce a more robust estimate of the model's performance.

The main purpose of cross validation is to overcome overfitting, which occurs when a model is trained too well on the training data and performs poorly on new, unseen data.

In **K-Fold Cross Validation** method, we split the data-set into k number of subsets (known as folds) then we perform training on the all the subsets but leave one(k-1) subset for the evaluation of the trained model. In this method, we iterate k times with a different subset reserved for testing purpose each time.

The general steps to implement K-Fold cross validation are as follows:

Step 1: Shuffle the dataset randomly.

Step 2: Split the dataset into k groups

Step 3: For each unique group:

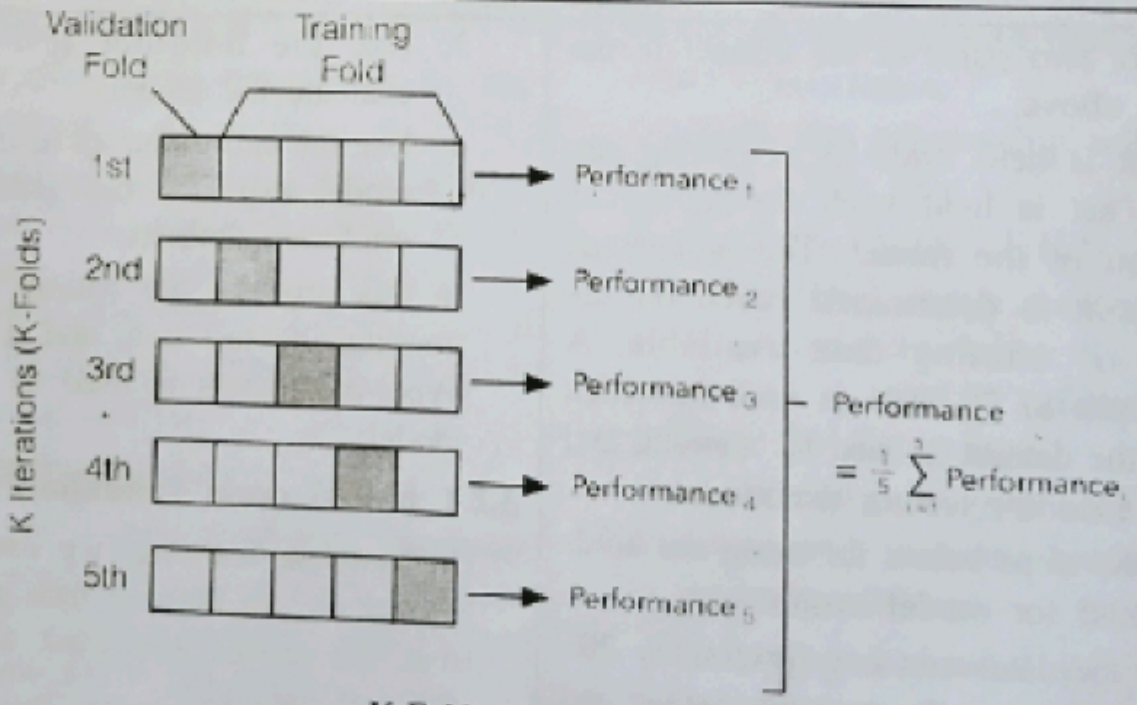
a. Take the group as a hold out or test data set

b. Take the remaining groups as a training data set

c. Fit a model on the training set and evaluate it on the test set

d. Retain the evaluation score and discard the model

Step 4 : Summarize the skill of the model using the sample of model evaluation scores



K-Fold cross validation

Advantages of cross-validation:

- More accurate estimate of out-of-sample accuracy.
- More "efficient" use of data as every observation is used for both training and testing.
- Overcome the issue of overfitting

Three common tactics for choosing a value for k are as follows:

Representative: The value for k is chosen such that each train/test group of data samples is large enough to be statistically representative of the broader dataset.

k=5 or 10: The value for k is fixed to 5 or 10, a value that has been found through experimentation to generally result in a model skill estimate with low bias a modest variance.

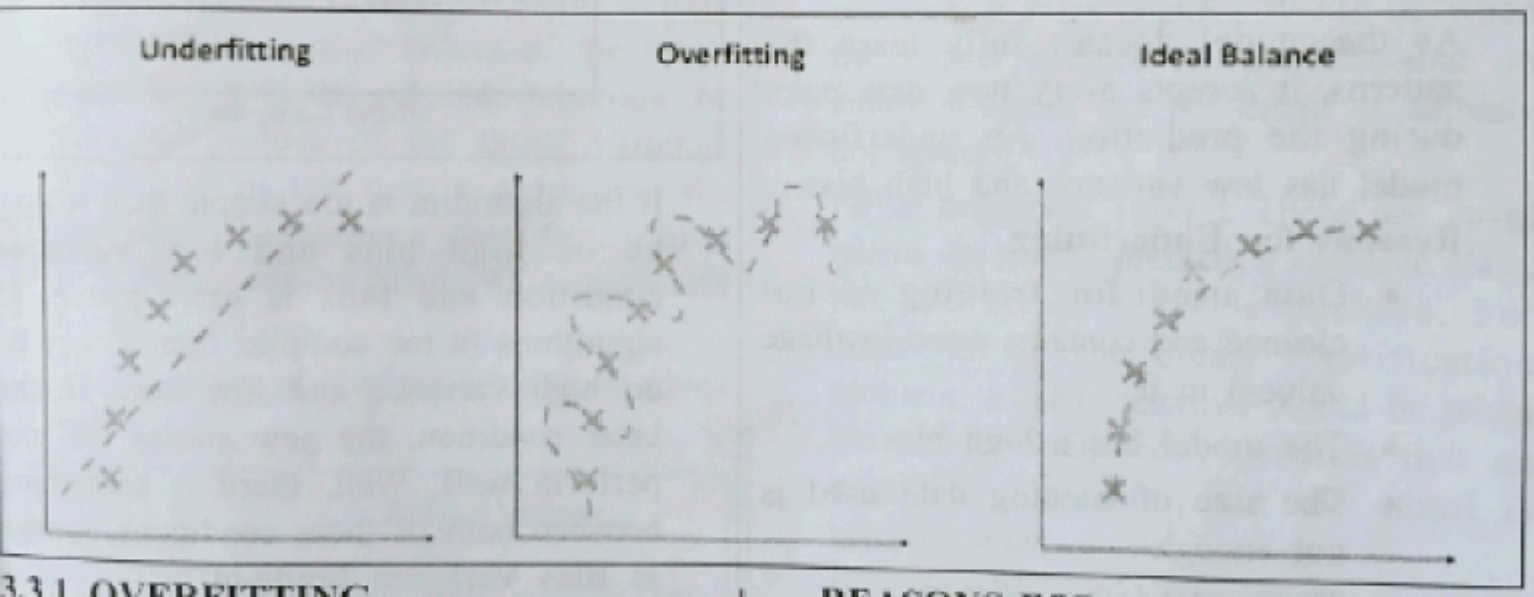
k=n: The value for k is fixed to n, where n is the size of the dataset to give each test sample an opportunity to be used in the hold out dataset. This approach is called leave-one-out cross-validation.

3.3 Model representation and interpretability

Interpretability focuses on the *how*.

Fitness of a target function approximated by a learning algorithm determines how correctly it is able to classify a set of data it has never seen.

In Machine Learning, if the model is able to fit on training data doesn't mean that it will perform well on testing data. This disparity between the performance on the training and test data is called Generalization Gap. It is common in machine learning problems to observe a gap between the training and testing performance.



3.3.1 OVERFITTING

Overfitting is a scenario where the machine learning model tries to learn from the data along with the noise present in the data and tries to fit each data point on the curve.

Overfitting is a fundamental issue in supervised machine learning which prevents us from perfectly generalizing the models to well fit observed data on training data, as well as unseen data on the testing sets. When a model is high on variance, it is then said to as Overfitting of Data.

REASONS FOR OVERFITTING

- Data used for training is not cleaned and contains noise (garbage values) in it
- The model has a high variance
- The size of training data used is not enough
- The model is too complex

METHODS TO AVOID OVERFITTING

- Use re-sampling techniques like k-fold Cross-validation

- Hold back of a validation data set
- Remove the nodes which have no. predictive power

3.3.2 UNDERFITTING

Underfitting is a scenario where a machine learning model can neither learn the relationship between variables in the data nor predict or classify a new data point.

In order to avoid overfitting, we could stop the training at an earlier stage. But it might also lead to the model not being able to learn enough from training data, which may find it difficult to capture the dominant trend. This is known as underfitting.

As the model doesn't fully learn the patterns, it accepts every new data point during the prediction. An underfitting model has low variance and high bias.

Reasons for Underfitting

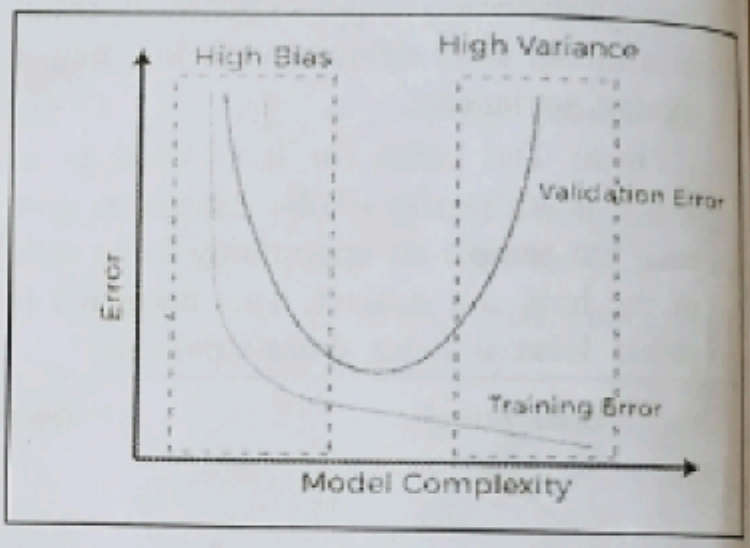
- Data used for training is not cleaned and contains noise (garbage values) in it
- The model has a high bias
- The size of training data used is not enough
- The model is too simple

Methods to avoid Underfitting

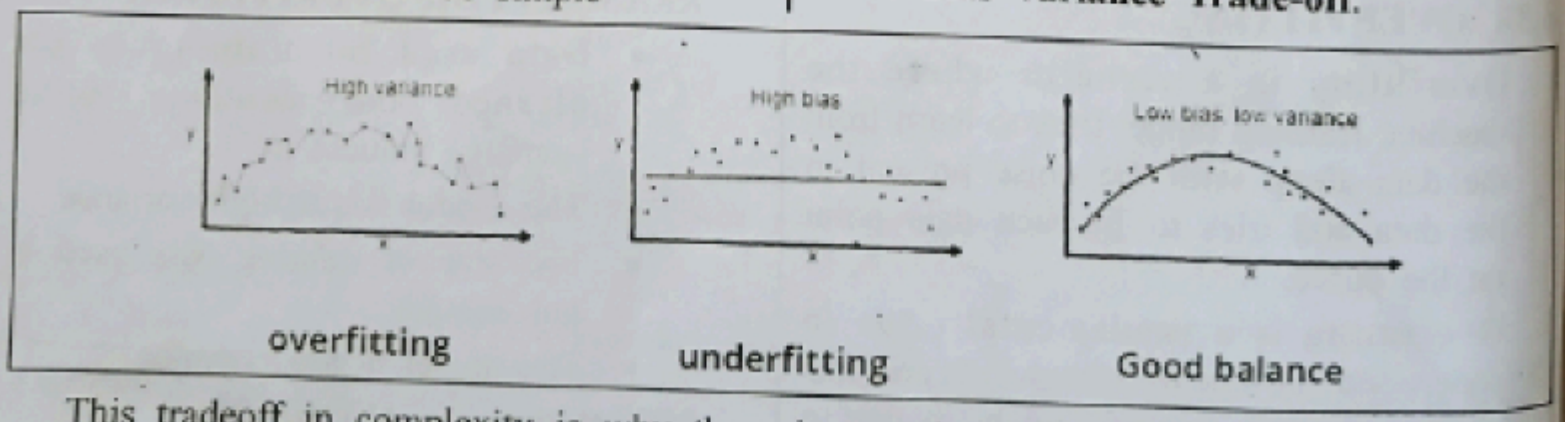
- Use more training data
- Reduce features by effective feature selection method

3.3.3 Bias-variance trade-off

It is important to understand prediction errors (bias and variance) when it comes to accuracy in any machine learning algorithm.



If the algorithm is too simple then it may be on high bias and low variance condition and thus is error-prone. If algorithms fit too complex then it may be on high variance and low bias. In the latter condition, the new entries will not perform well. Well, there is something between both of these conditions, known as **Bias Variance Trade-off.**



This tradeoff in complexity is why there is a tradeoff between bias and variance. An algorithm can't be more complex and less complex at the same time. To build a good model, we need to find a good balance between bias and variance such that it minimizes the total error.

3.4 Evaluating Performance of a Model

Classification is one of the major tasks in Supervised Learning. In classification, the number of correct and incorrect predictions made by the model is evaluated and accuracy of the model calculated based on that evaluation.

3.4.1 Confusion Matrix

A confusion matrix is a table used to evaluate the performance of a machine learning model by comparing its predictions to the true values of the dataset. It is used in classification.

It consists of four possible outcomes: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

The rows of the confusion matrix represent the actual values of the target variable, while the columns represent the predicted values of the target variable. Each cell in the matrix represents the number of instances that fall into a particular combination of actual and predicted values.

The structure of confusion matrix is given below :

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

True positives (TP) : These are cases where the model predicts a positive class, and the actual class is also positive. For example, if we're trying to classify whether an email is spam or not, a true positive would be when the model correctly predicts that an email is spam and the email is actually spam.

True negatives (TN) : These are cases where the model predicts a negative class, and the actual class is also negative. For example, using the same email classification problem, a true negative would be when the model correctly predicts that an email is not spam, and the email is actually not spam.

False positives (FP) : These are cases where the model predicts a positive class, but the actual class is negative. For example, in the email classification problem, a false positive would be when the model incorrectly predicts that an email is spam, but the email is actually not spam.

False negatives (FN) : These are cases where the model predicts a negative class, but the actual class is positive. For example, in the email classification problem, a false negative would be when the model incorrectly predicts that an email is not spam, but the email is actually spam.

In general, we want to minimize false positives and false negatives and maximize true positives and true negatives to have a good performance of a machine learning model.

The confusion matrix summarizes these four possible outcomes and helps us to calculate various performance metrics such as accuracy, error rate, sensitivity, specificity, precision, recall, and F1-score.

- **Accuracy :** This measures the proportion of correctly classified instances out of all the instances in the dataset. It is calculated

by dividing the number of correct predictions by the total number of predictions. The formula is :

- **Accuracy = (TP + TN) / (TP + TN + FP + FN)**

Error rate: This indicates the percentage of misclassification. The formula is :

Error rate = (FP + FN) / (TP + TN + FP + FN)

In other words,

Error rate = 1 – Model Accuracy

- **Sensitivity**: The sensitivity of a model measures the proportion of positive examples which were correctly classified. The formula is:

  Sensitivity = TP / (TP + FN)

- **Specificity**: The specificity of a model measures the proportion of negative examples which were correctly classified. The formula is:

  Specificity = TN / (TN + FP)

- **Precision**: This measures the proportion of true positive predictions out of all the positive predictions made by the model. It is calculated by dividing the number of true positives by the total number of positive predictions. The formula is:

  Precision = TP / (TP + FP)

- **Recall**: This measures the proportion of true positive predictions out of all the actual positive instances in the dataset. It is calculated by dividing the number of true positives by the total number of positive instances. The formula is:

  Recall = TP / (TP + FN)

- **F1 score**: This is a harmonic mean of precision and recall, and provides a balanced measure of a model's performance. It is calculated by taking the harmonic mean of precision and recall. The formula is:

  F1 score = 2 * (Precision * Recall) / (Precision + Recall)

Example 1 :

Suppose we have a binary classification problem where we want to predict if a person has a disease or not based on their medical test results. We have a dataset of 100 patients, where 60 do not have the disease and 40 have the disease. We train a machine learning model on this data to predict the disease status based on test results.

After training the model, we test it on a separate dataset of 20 patients and get the following results:

True positives (TP) : The model correctly predicted that 7 patients have the disease.

True negatives (TN) : The model correctly predicted that 10 patients do not have the disease.

False positives (FP) : The model predicted that 2 patients have the disease, but they do not actually have the disease.

False negatives (FN) : The model predicted that 1 patient does not have the disease, but they actually have the disease.

We can use these results to create a confusion matrix, which would look like this:

	Predicted No Disease	Predicted Disease
Actual No Disease	TN = 10	FP = 2
Actual Disease	FN = 1	TP = 7

From the confusion matrix, we can calculate several performance metrics of the model, including accuracy, error rate, sensitivity, specificity, precision, recall, and F1 score.

For **example :**

Accuracy = (TP + TN) / (TP + TN + FP + FN)
= (7 + 10) / (7 + 10 + 2 + 1)
= 0.85 (85% Accuracy of the model)

Error rate = 1 – Accuracy
= 1 – 0,85 = 0.15

Sensitivity = TP / (TP + FN)
= 7 / (7+1) = 0.875

Specificity = TN / (TN + FP)
= 10 / (10 + 2) = 0.8333

Precision = TP / (TP + FP)
= 7 / (7 + 2) = 0.78

Recall = TP / (TP + FN)
= 7 / (7 + 1) = 0.88

F1 score = 2 * (precision * recall) / (precision + recall)
= 2 * (0.78 * 0.88) / (0.78 + 0.88) = 0.82

These metrics provide insight into how well the model is performing and can help us make improvements or adjustments as necessary.

Example 2 :

Let's consider the example of a binary classification problem where we want to predict whether an email is spam or not spam. We have a dataset of 100 emails, where 60 are not spam (negative class) and 40 are spam (positive class). We build a machine learning model to predict whether each email is spam or not, and the results are as follows :

	Predicted Not Spam	Predicted Spam
Actual Not Spam	50	10
Actual Spam	5	35

This confusion matrix tells us that out of the 60 actual not spam emails, the model correctly predicted 50 as not spam (True Negatives, TN) but incorrectly predicted 10 as spam (False Positives, FP). Out of the 40 actual spam emails, the model correctly predicted 35 as spam (True Positives, TP) but incorrectly predicted 5 as not spam (False Negatives, FN).

Using this confusion matrix, we can calculate several performance metrics of the model.

For the given example, accuracy is (50+35)/100 = 85%,

precision is 35/(10+35) = 77.8%,

recall is 35/(5+35) = 87.5%, and

F1 score is 2*(precision*recall)/(precision+recall) = 82.4%.

These metrics provide insight into how well the model is performing and can help us make improvements or adjustments as necessary.

3.5 Improving performance of a model

Let's see the different avenues to improve the performance of models.

1. Feature engineering: (Feature engineering is the process of selecting and transforming the input variables used in the model.) By selecting the most important features and transforming them into a more suitable form, we can improve the performance of the model.
2. Model Parameter tuning: Most machine learning algorithms have a set of model parameters that can be adjusted to improve performance. Model parameters control the learning rate, regularization, number of trees in a random forest, and so on. Tuning these model parameters can lead to significant performance improvements.

3. Data cleaning and preprocessing: It is important to clean and preprocess the data before training the model. This includes removing missing values, scaling the features, encoding categorical variables, and so on. By carefully preprocessing the data, we can improve the performance of the model.
4. Ensemble methods: Ensemble methods combine multiple models to improve performance. This can include bagging, boosting, and stacking methods.
5. Larger training set: In some cases, the model may be underfitting because the training set is too small. By increasing the size of the training set, we can improve the performance of the model.
6. Regularization: Regularization methods such as L1, L2, and dropout can help prevent overfitting and improve the generalization ability of the model.
7. Different algorithms: If one machine learning algorithm is not performing well, it may be worth trying a different algorithm. There are many different algorithms to choose from, and some may be better suited to the particular problem at hand.

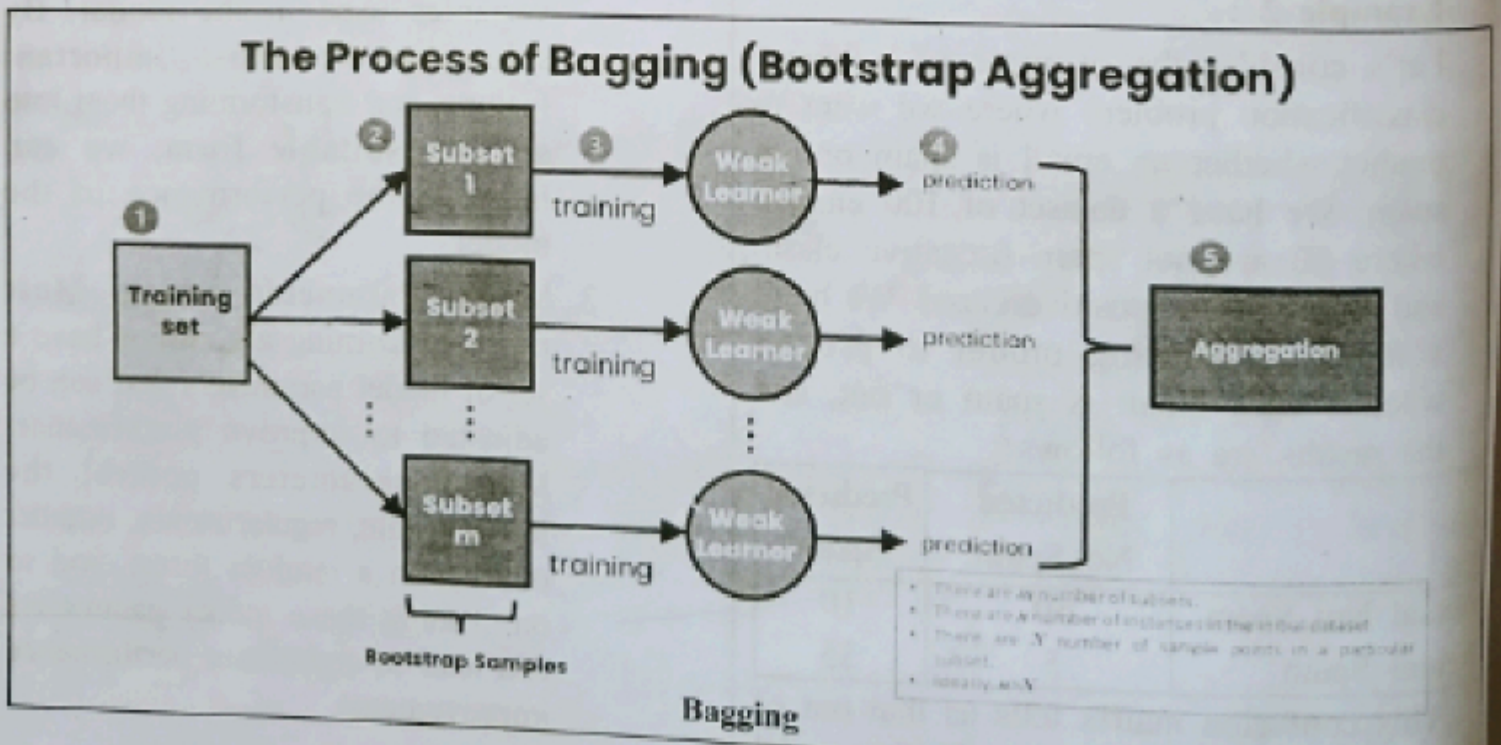
Let's discuss two avenues in detail.

3.5.1 Ensemble Approach

Ensemble learning is a machine learning technique that combines multiple models to improve the overall performance. The basic idea behind ensemble learning is that by combining multiple models, we can reduce the error caused by individual models and improve the overall accuracy.

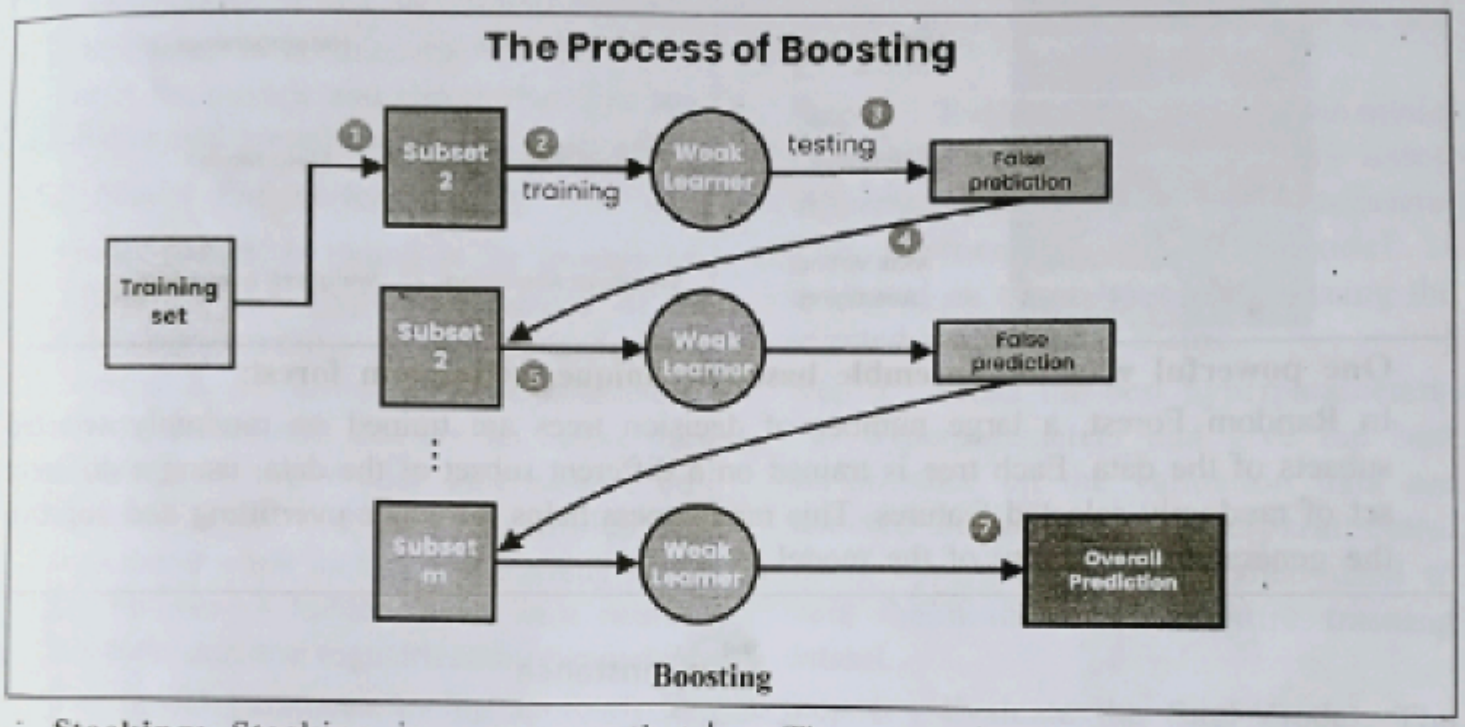
There are several different ways to create an ensemble of models, but some of the most common methods include:

Bagging: Bagging stands for Bootstrap Aggregating. In bagging, we create multiple samples of the training data by sampling with replacement. Then we train a separate model on each of these samples. Finally, we combine the results of all models to make the final prediction.



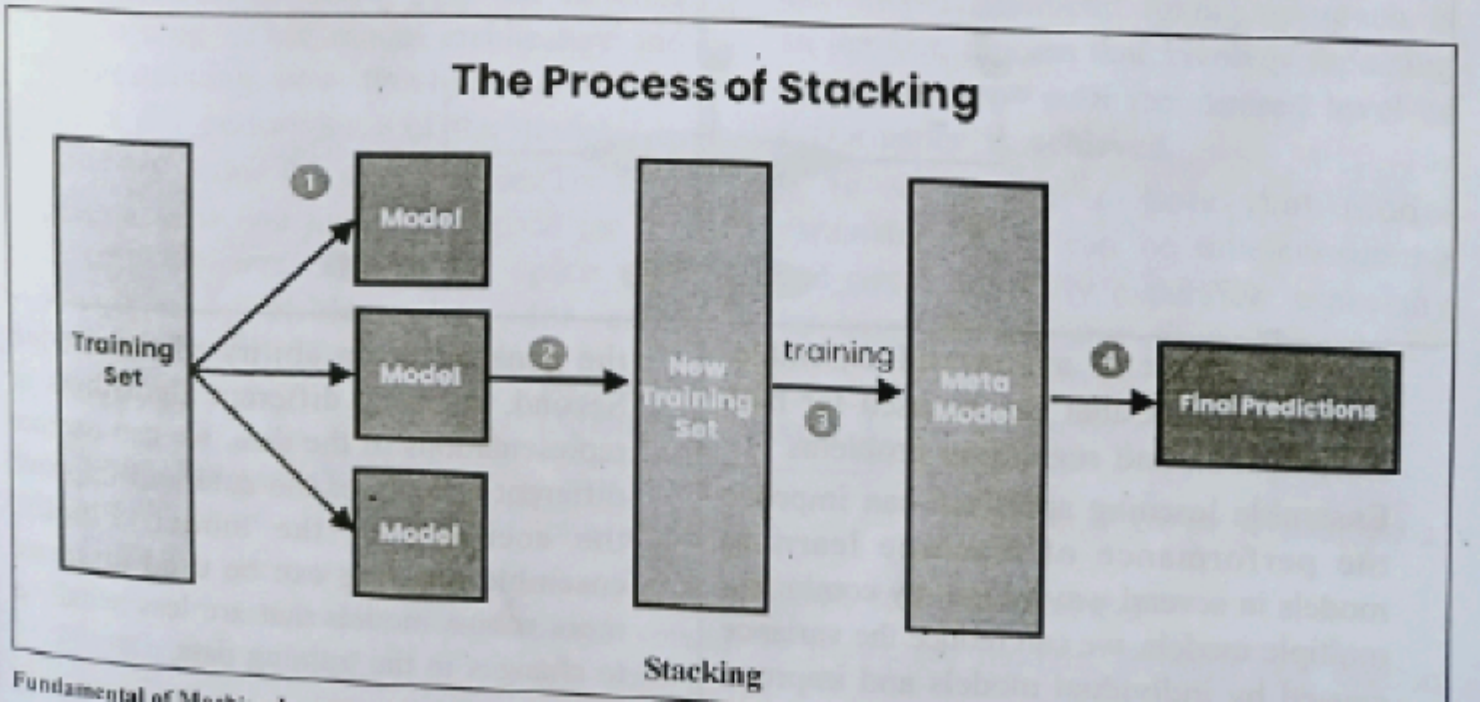
Bagging

Boosting: Boosting is a sequential process that creates an ensemble of models by iteratively improving the performance of a single model. In boosting, we train a base model on the entire training set. Then, we assign weights to each sample in the training set and train a new model on the same dataset. In each iteration, we give more weight to the misclassified samples and less weight to the correctly classified samples. Finally, we combine the results of all models to make the final prediction.



Boosting

Stacking: Stacking is a process that involves combining multiple models that use different algorithms or different representations of the data. In stacking, we train multiple models on the same dataset. Then, we train a meta-model on the output of these models. The meta-model learns how to combine the outputs of the models to make the final prediction.

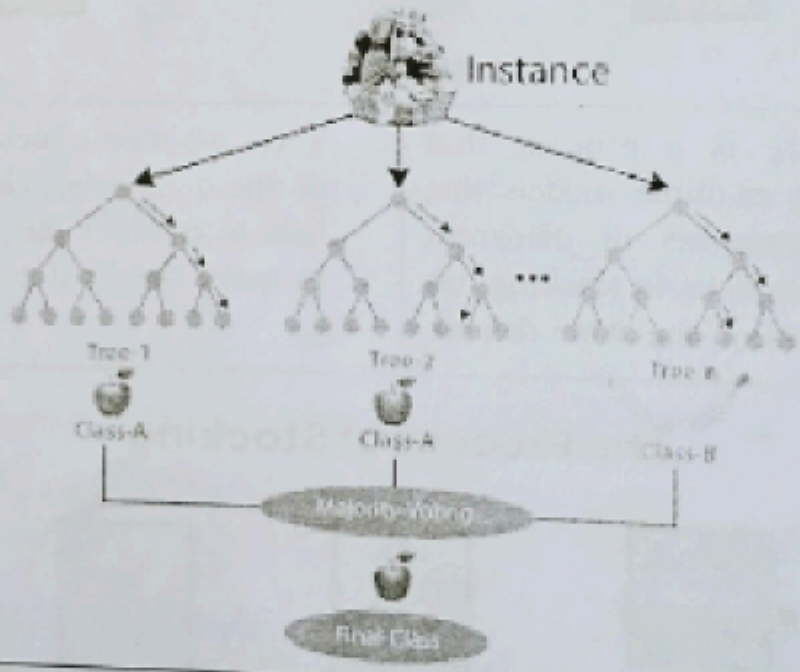


Stacking

	Bagging	Boosting	Stacking
Purpose	Reduce Variance	Reduce Bias	Improve Accuracy
Base Learner Types	Homogeneous	Homogeneous	Heterogeneous
Base Learner Training	Parallel	Sequential	Meta Model
Aggregation	Max Voting, Averaging	Weighted Averaging	Weighted Averaging

One powerful way for ensemble based technique is Random forest:

In Random Forest, a large number of decision trees are trained on randomly selected subsets of the data. Each tree is trained on a different subset of the data, using a different set of randomly selected features. This randomness helps to reduce overfitting and improve the generalization ability of the model.



Random Forest is a powerful ensemble-based technique that can be used for both classification and regression problems.

Ensemble learning approach can improve the performance of machine learning models in several ways. First, by combining multiple models, we can reduce the variance caused by individual models and improve the generalization ability of the model. Second, by using different algorithms or representations of the data, we can capture different aspects of the data and improve the accuracy of the model. Finally, ensemble learning can be used to create more robust models that are less sensitive to changes in the training data.

However, ensemble learning approach also has some drawbacks. It can be computationally expensive and may require a large amount of memory. In addition, the performance of the ensemble may be highly dependent on the performance of the individual models. Therefore, it is important to carefully select the models and ensure that they are diverse and complementary to each other.

3.5.2 Model Parameter Tuning

Model parameter tuning is the process of adjusting the hyperparameters of a machine learning model in order to improve its performance. Hyperparameters are the parameters that are set before the training of the model and cannot be learned from the data. Examples of hyperparameters include the learning rate, the number of hidden layers in a neural network, and the regularization parameter.

The model parameter tuning approach in machine learning involves the following steps:

Step 1 : Define the hyperparameters: The first step is to define the hyperparameters that need to be tuned. This can be done by looking at the model architecture and understanding how the hyperparameters affect the performance of the model.

Step 2 : Define the search space: The next step is to define the search space for the hyperparameters. The search space is a range of possible values for each hyperparameter. The search space can be defined using a grid search or a random search.

Step 3 : Select a performance metric: The performance metric is the metric that will be used to evaluate the performance of the model. The most common performance metrics are accuracy, precision, recall, and F1-score.

Step 4 : Evaluate the model: The model is trained on the training data using different combinations of hyperparameters. The performance of the model is evaluated on the validation data using the selected performance metric.

Step 5 : Select the best hyperparameters: The hyperparameters that give the best performance on the validation data are selected as the final hyperparameters. These hyperparameters are then used to train the model on the entire training dataset.

Step 6 : Evaluate the final model: The final model is evaluated on the test data using the selected performance metric to determine the generalization performance of the model.

The model parameter tuning approach is an iterative process that involves repeating the above steps until the desired level of performance is achieved.

It is important to note that model parameter tuning can be time-consuming and computationally expensive, especially for large datasets and complex models.

Question Bank

■ **Short-Answer Questions** **(3 or 4 marks) :**

1. Define model. How can you train a model ?
2. Give the difference between predictive model and descriptive model.
3. State any four real-world problems solved by predictive models. Explain any one in brief.

4. State any four real-world problems solved by descriptive models. Explain any one in brief.
5. Write down steps to use holdout method for model training.
6. Draw a detailed diagram to show the approach of 10-fold cross validation.
7. Give the difference between Bagging and Boosting.
8. Define overfitting. When does it happen?
9. Define underfitting. When does it happen?
10. Write a short note on bias-variance trade-off in context of model fitting.
11. Explain structure of confusion matrix.
12. Define following terms: Sensitivity, Specificity
13. Write a brief note on stacking.
14. State various ways to improve performance of a model.

■ **Long-Answer Questions** **(7 marks):**

1. Explain different types of model.
2. Explain Holdout method in detail.
3. Describe k-fold cross validation in detail.
4. Explain bagging, boosting and stacking in detail.
5. Describe Ensemble learning approach in detail.
6. Consider the following confusion matrix of the win/loss prediction of cricket match. calculate the accuracy, error rate, sensitivity, specificity, precision, recall and F-measure of the model.

	Actual Win	Actual Loss
Predicted Win	82	7
Predicted Loss	3	8

7. While predicting malignancy of tumour of a set of patients using a classification model, following are the data recorded:
   1. Correct predictions – 20 malignant, 70 benign
   2. Incorrect predictions – 4 malignant, 6 benign

   Create confusion matrix for the same. And, calculate the accuracy, error rate, sensitivity, specificity, precision, recall and F-measure of the model.

■ **Check your knowledge :**

1. What is the difference between bagging and boosting?
   a) **Bagging reduces the variance of a model, while boosting reduces the bias of a model.**
   b) Bagging reduces the bias of a model, while boosting reduces the variance of a model.
   c) Bagging increases the complexity of a model, while boosting reduces the complexity of a model.

d) Bagging reduces the complexity of a model, while boosting increases the complexity of a model.

2. Which of the following is not a metric that can be calculated using a confusion matrix?
   a) Precision
   b) Recall
   c) F1 score
   d) R-squared

3. Which of the following is a common technique used to prevent overfitting in a machine learning model?
   a) Adding more features to the model
   b) Increasing the number of epochs during training
   c) Reducing the size of the model
   d) Decreasing the learning rate during

4. What is the holdout method used for?
   a) To evaluate the performance of a machine learning model on a dataset that it hasn't seen during training
   b) To train a machine learning model on a subset of a dataset
   c) To generate new data for a machine learning model
   d) To reduce the complexity of a machine learning model

5. Which of the following is a common technique used to prevent underfitting in a machine learning model?
   a) Adding more features to the model
   b) Increasing the number of epochs during training
   c) Reducing the size of the model
   d) Decreasing the learning rate during training

■

4 Supervised Learning : Classification and Regression

4.1 Introduction to Supervised Learning
4.2 Classification Model
4.3 Learning Steps of Classification
4.4 Classification Algorithms
4.4.1 k-Nearest Neighbour Algorithm
4.4.2 Support Vector Machines (SVM)
4.5 Regression
4.5.1 Simple Linear Regression
4.5.2 Multiple Linear Regression
4.5.3 Logistic Regression
■ Questions Bank

4.1 Introduction to Supervised Learning

Supervised learning is a machine learning technique that involves training a model to make predictions based on labeled data. In supervised learning, the model learns from a dataset that includes both input features and corresponding output labels. The goal of supervised learning is to develop a model that can accurately predict the output for new input data that it has not seen before.

The labeled dataset used for supervised learning typically consists of a set of input features and corresponding output labels. The input features can be numerical or categorical, and the output labels can be binary (e.g. true or false) or multi-class (e.g. classification of different objects).

The process of supervised learning involves training a model using the labeled dataset, which involves finding the relationship between the input features and output labels. Once the model is trained, it can be used to predict the output for new input data that it has not seen before.

Supervised learning is commonly used in applications such as image classification, text classification, fraud detection, and recommendation systems. The key advantage of supervised learning is that it can produce highly accurate predictions when trained on a large and diverse dataset. However, it requires a labeled dataset, which can be costly and time-consuming to create.

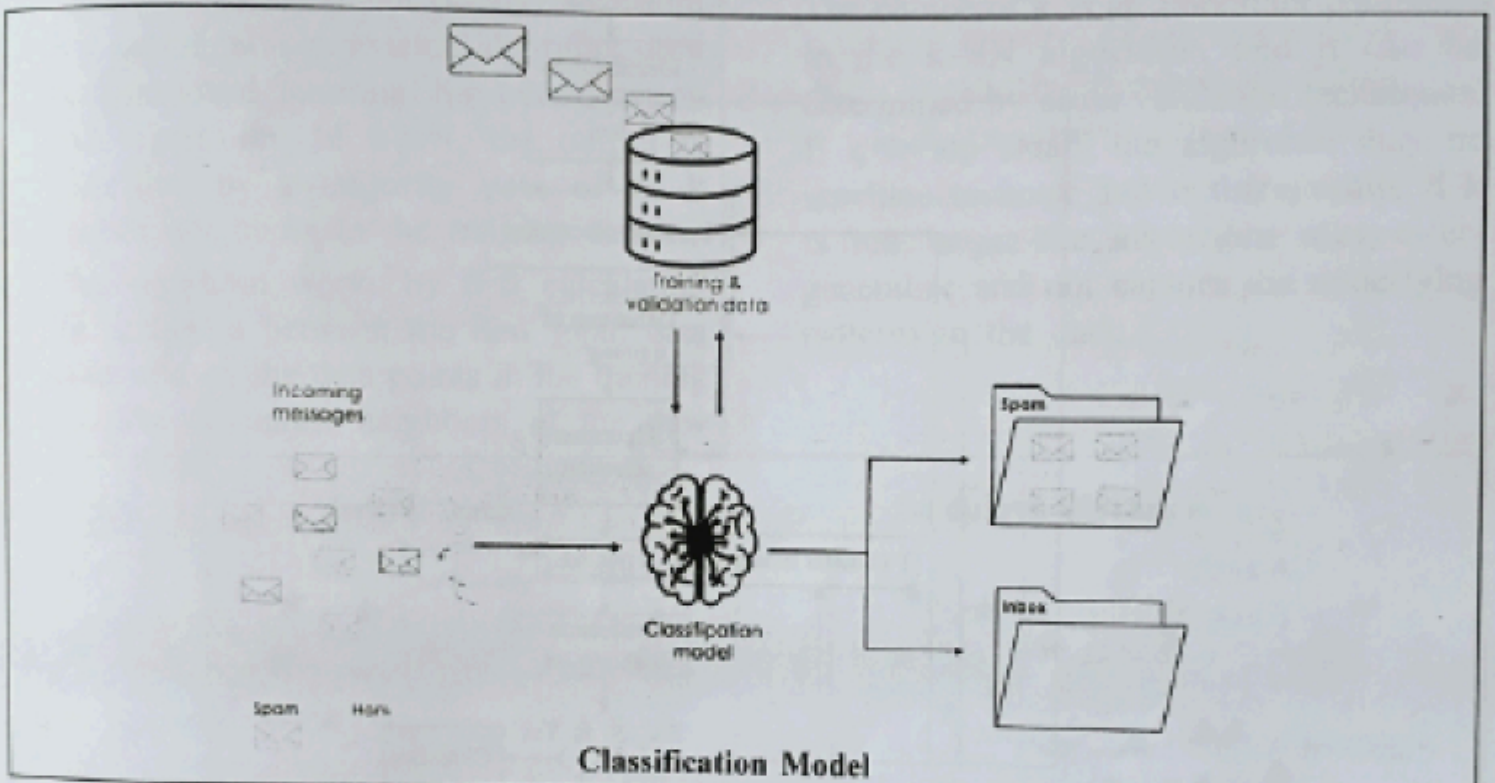
4.2 Classification Model

Classification is a type of supervised learning where the goal is to predict the categorical output variable for a given set of input features. In classification, the model learns to distinguish between different classes based on labeled data.

Classification is a type of supervised learning where the goal is to predict the categorical output variable for a given set of input features. In classification, the model learns to distinguish between different classes based on labeled data.

For example, in a binary classification problem, the goal is to predict one of two possible classes, such as "spam" or "not spam". In multi-class classification, the goal is to predict one of several possible classes, such as identifying different types of animals from an image.

To build a classification model, the first step is to collect labeled data. This involves creating a dataset where each data point includes a set of input features and a corresponding output label. The input features can be anything that might be relevant to the classification task, such as pixel values for image data or demographic data for predicting customer behavior.



Classification Model

Once the dataset is prepared, the next step is to choose an appropriate classification algorithm, such as logistic regression, decision trees, or neural networks. The algorithm is then trained on the labeled dataset to learn the relationship between the input features and the output labels.

During training, the algorithm adjusts its parameters to minimize the difference between the predicted output and the actual output in the labeled dataset. Once the model is trained, it can be used to predict the output label for new input data.

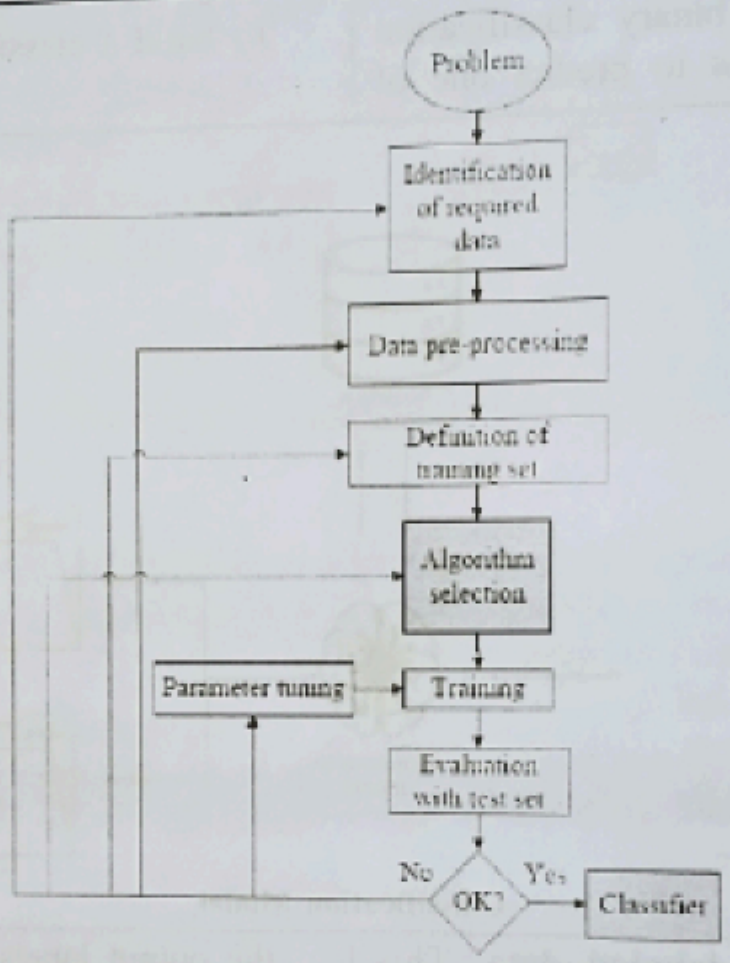
In summary, classification is a type of supervised learning where a target feature, which is of categorical type, is predicted for test data on the basis of the information imparted by the training data. The target categorical feature is known as class.

Evaluation of a classification model

involves measuring its accuracy on a held-out validation dataset or through cross-validation techniques. The accuracy of the model can be improved by using techniques such as feature engineering, regularization, and ensembling.

4.3 Learning Steps of Classification

Step 1: Problem Identification: Identifying the problem is the first step in the supervised learning model. The problem must be well-posed problem.



Step 2: Identification of Required Data: Collecting and preparing a labeled dataset that includes input features and corresponding output labels.

Step 3: Data Preprocessing: Cleaning and preparing dataset, which may involve removing missing data, handling outliers and transferring the data into a suitable format.

Step 4: Definition of Training Data Set: Before starting the analysis, the user should decide type of data set is to be used as a training set. The training set needs to be actively representative of the real-world use of the given problem.

Step 5: Algorithm Selection: Choosing an appropriate classification algorithm based on the nature of the problem and the size and complexity of the dataset.

Step 6: Training: Using the labeled dataset to train the selected algorithm by adjusting its parameters to minimize the difference between the predicted output and the actual output. Also, adjusting the hyperparameters of the algorithm to optimize the model's performance on the validation dataset should be done, if it needs..

Step 7: Evaluation with the test dataset: Training data is run on the algorithm, and its performance is measured here. If a desired output is not obtained, again training of parameters may be required.

4.4 Classification Algorithms

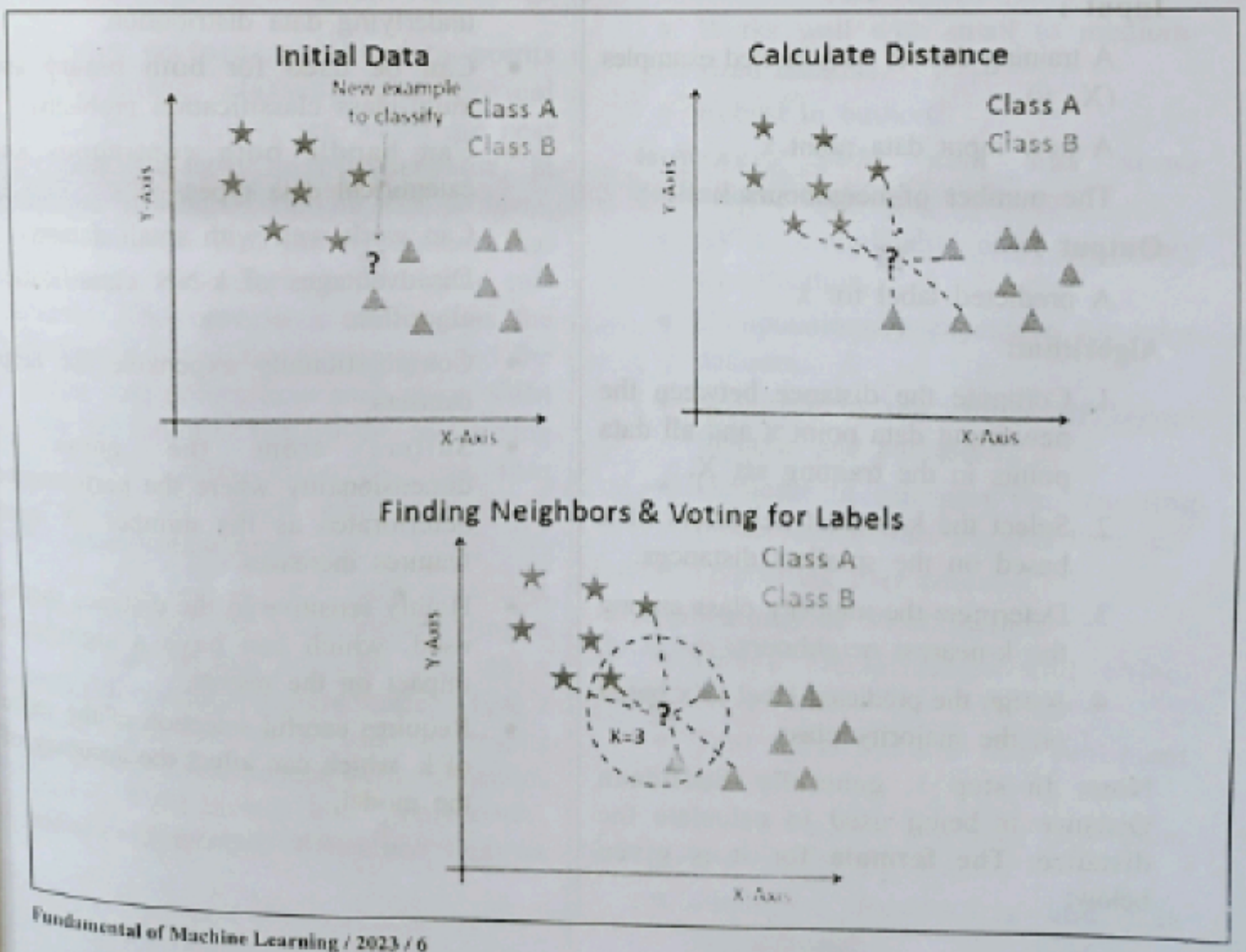
4.4.1 k-Nearest Neighbour Algorithm

The k-Nearest Neighbor (k-NN) algorithm is a simple, non-parametric algorithm used in supervised learning for classification and regression. In k-NN, the output is classified by a majority vote of its k nearest neighbors in the training dataset.

The algorithm works by first calculating the distances between the new input data point and all the data points in the training set. The k-nearest neighbors of the new data point are then determined based on the smallest distances.

In classification, the output label of the new data point is then determined by the majority class among its k-nearest neighbors. For regression, the output value is the average of the output values of its k-nearest neighbors.

The choice of k is an important parameter in the k-NN algorithm, and it can be determined by cross-validation techniques. If k is too small, the algorithm may be sensitive to noise and outliers, while if k is too large; the algorithm may over-generalize and not capture the underlying patterns in the data.



The k-NN algorithm is a simple and easy-to-understand algorithm, but it can be computationally expensive for large datasets. It also suffers from the curse of dimensionality, where the performance of the algorithm deteriorates as the number of input features increases.

Despite its limitations, the k-NN algorithm is still widely used for its simplicity and effectiveness in many applications, including image recognition, recommender systems, and anomaly detection.

For kNN, there is no learning happening in the real sense. Therefore, kNN falls under the category of lazy learner.

Here is the general k-Nearest Neighbor (k-NN) algorithm for classification:

Input :

A training dataset with labeled examples (X, y)

A new input data point x

The number of neighbours k

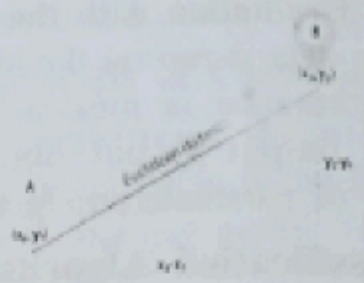
Output :

A predicted label for x

Algorithm:

1. Compute the distance between the new input data point x and all data points in the training set X.
2. Select the k-nearest neighbors of x based on the smallest distances.
3. Determine the majority class among the k-nearest neighbours.
4. Assign the predicted label to x based on the majority class.

Note: In step 1, generally Euclidean Distance is being used to calculate the distance. The formula for it is given below:



$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Advantages of k-NN classification **algorithm :**

- Simple and easy to understand.
- Non-parametric method does not require any assumptions about the underlying data distribution.
- Can be used for both binary and multi-class classification problems.
- Can handle both continuous and categorical data types.
- Can work well with small datasets.

Disadvantages of k-NN classification algorithm:

- Computationally expensive for large datasets.
- Suffers from the curse of dimensionality where the performance deteriorates as the number of input features increases.
- Highly sensitive to the distance metric used, which can have a significant impact on the results.
- Requires careful selection of the value of k, which can affect the accuracy of the model.

Applications of k-NN classification algorithm:

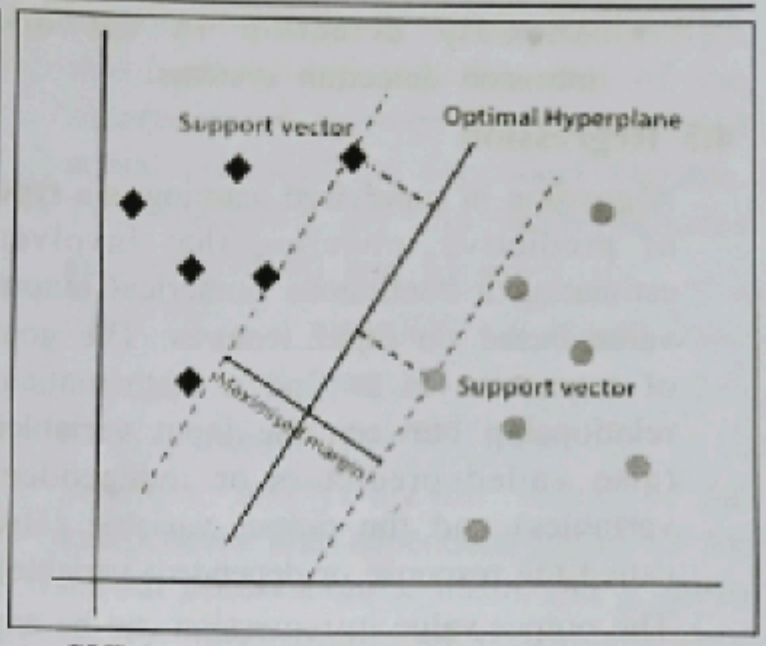
- Image recognition and classification.
- Recommendation systems for online shopping or entertainment.
- Credit risk assessment in finance.
- Medical diagnosis based on patient characteristics.
- Fraud detection in credit card transactions.

4.4.2 Support Vector Machines (SVM)

Support Vector Machine (SVM) is a powerful classification algorithm used in supervised learning. The algorithm works by finding the optimal hyperplane that separates the different classes in the input data.

Support vectors are the data points (representing classes), the critical component in a data set, which are near the identified set of lines (**hyperplane**). In the case of a binary classification problem, SVM aims to find the hyperplane that maximizes the margin between the two classes. The **margin** is defined as the distance between the hyperplane and the closest data points from each class. SVM finds the optimal hyperplane by solving a convex optimization problem that maximizes the margin while minimizing the classification error.

If the input data is not linearly separable, SVM can still be used by applying a kernel function that maps the data into a higher-dimensional feature space, where a linear hyperplane can be used to separate the classes. Some popular kernel functions used in SVM include linear, polynomial, radial basis function (RBF), and sigmoid.



SVM has several advantages, including:

- Effective for high-dimensional data.
- Can handle non-linearly separable data by applying kernel functions.
- Works well with small to medium-sized datasets.
- Robust to outliers.

However, SVM also has some limitations, including :

- SVM is applicable only for binary classification.
- Computationally expensive for large datasets.
- Sensitive to the choice of kernel function and parameters.
- Difficult to interpret the resulting model and extract insights from it.
- Overfitting may cause.

Applications of SVM include :

- Image classification and object recognition.
- Text classification and sentiment analysis.
- Bioinformatics and gene expression analysis.
- Financial forecasting and risk

- Anomaly detection in network intrusion detection systems.

4.5 Regression

Regression in supervised learning is a type of predictive modeling that involves estimating a continuous numerical output value based on input features. The goal of regression is to find a mathematical relationship between the input variables (also called predictors or independent variables) and the output variable (also called the response or dependent variable).

The output value in regression can be any numerical value, including integers, real numbers, or even negative values. Regression models can be used for both linear and nonlinear relationships between the input and output variables.

The main goal of regression is to create a function that can predict the output variable for new inputs with a certain degree of accuracy. In order to achieve this goal, regression algorithms use statistical techniques to find the optimal parameters of the model that minimize the difference between the predicted values and the actual values in the training dataset.

Applications of Regression :

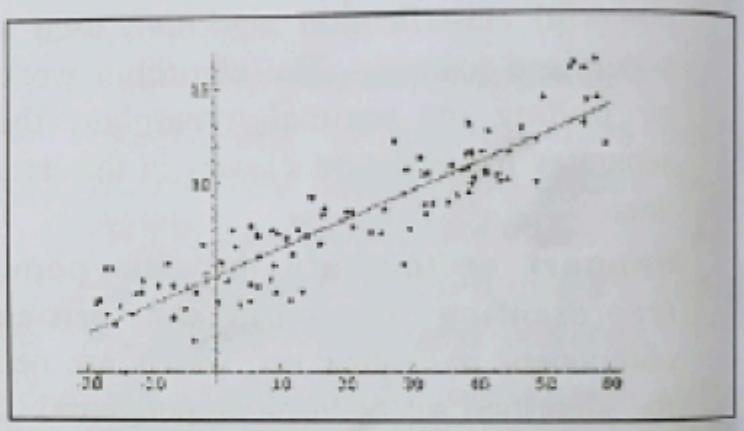
- Predicting sales or revenue based on marketing spend or other factors.
- Estimating the price of a house based on features such as location, size, and number of rooms.
- Forecasting demand for a product based on historical sales data.
- Predicting patient outcomes based on medical data.
- Analyzing trends in financial data such as stock prices or interest rates.

4.5.1 Simple Linear Regression

Linear regression is one of the most basic types of regression in machine learning. The linear regression model consists of a predictor variable and a dependent variable related linearly to each other. In case the data involves more than one independent variable, then linear regression is called multiple linear regression models.

$$y = mx + c + e$$

The below-given equation is used to denote the linear regression model :



Simple linear regression is a type of regression analysis that is used to model the relationship between two variables, where one variable is considered the predictor or independent variable, and the other is the response or dependent variable. The goal of simple linear regression is to find the best linear relationship between the two variables.

Here is an example of simple linear regression :

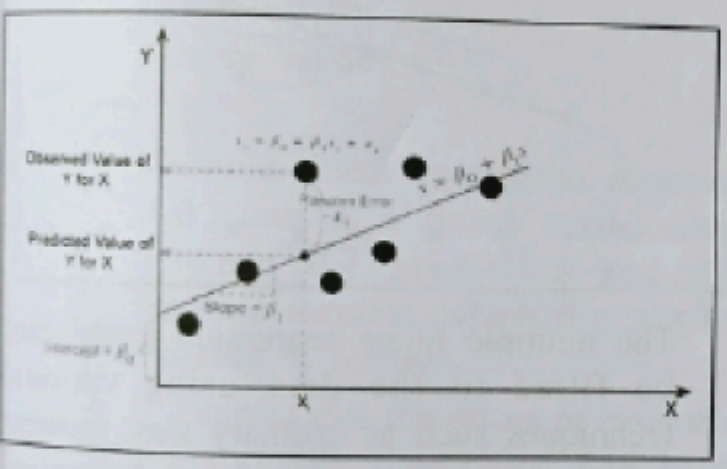
Suppose we have a dataset of 100 homes with two variables: square footage (independent variable) and sale price (dependent variable). We want to determine the relationship between these two variables, and specifically, how much the sale price of a home increases for each additional square foot of living space.

We can use simple linear regression to estimate the line of best fit for the data, which is a straight line that minimizes the sum of the squared differences between the predicted and actual values.

In simple linear regression, the model is expressed as:

$$Y = \beta 0 + \beta 1X + \varepsilon$$

where Y is the dependent variable, X is the independent variable, β0 is the intercept, β1 is the slope associated with the predictor variable X, and å is the random error term. The slope β1 indicates the change in the dependent variable associated with a one-unit change in the independent variable X.



The simple linear regression model can be fitted to the data using various techniques such as ordinary least squares (OLS) or maximum likelihood estimation (MLE). The fitted model can be used to make predictions for new values of the predictor variable or to test hypotheses about the relationship between the variables.

The validity of the simple linear regression model depends on certain assumptions such as linearity, normality, homoscedasticity, and independence of the error term. Violations of these assumptions can lead to biased and inefficient estimates and incorrect inference.

To estimate the values of β0 and β1, we can use the least squares method, which involves minimizing the sum of the squared errors between the predicted and actual values of the dependent variable. The equation for the least squares estimates of β0 and β1 are:

$$\beta 1 = (X_i - x_mean)(Y_i - y_mean) / (X_i - x_mean)^2$$

$$\beta 0 = y_mean - \beta 1 * x_mean$$

Where xi and yi are the values of the independent and dependent variables for the ith observation, x_mean and y_mean are the mean values of the independent and dependent variables, and the sums are taken over all observations in the dataset.

Once we have estimated the values of β0 and β1, we can use the equation of the line to predict the sale price of a home based on its square footage.

For example, if the estimated equation of the line is:

$$y = 100000 + 100x$$

This means that for every additional square foot of living space, the sale price of the home is expected to increase by Rs. 100. So, a home with 1500 square feet of living space would be expected to sell for Rs. 250,000, (100000 + 100*1500).

Pros :

- Simple linear regression is easy to implement and interpret.
- It provides a simple way to quantify the strength and direction of the relationship between two variables.
- It can be used to make predictions and estimate the value of the response variable for a given value of the predictor variable.

Cons:

- Simple linear regression assumes that there is a linear relationship between the predictor and response variables, which may not be the case in reality.
- It assumes that the errors or residuals are normally distributed and have constant variance, which may not hold in practice.
- It can be sensitive to outliers, influential observations, and violations of the underlying assumptions.

Appliactions :

- Simple linear regression is commonly used in finance to model the relationship between two financial variables, such as stock prices and interest rates.
- It is used in marketing to analyze the impact of advertising on sales or the relationship between price and demand for a product.
- It is used in engineering to model the relationship between two physical variables, such as temperature and pressure.
- It is used in social sciences to study the relationship between two psychological variables, such as personality and job satisfaction.

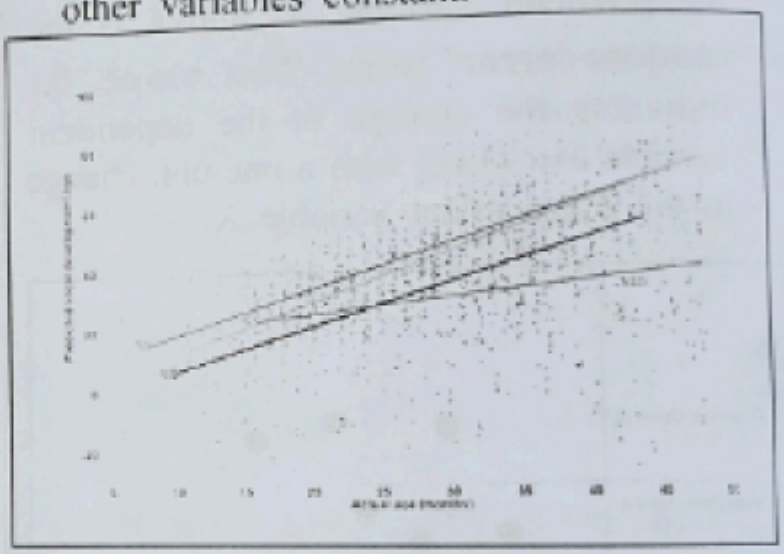
4.5.2 Multiple Linear Regression

Multiple linear regression is a statistical technique used to model the relationship between a response or dependent variable and two or more predictor or independent variables. It extends the simple linear regression model to multiple predictors, allowing for more complex relationships between the variables.

In multiple linear regression, the model is expressed as:

$$Y = \beta 0 + \beta 1X1 + \beta 2X2 + \ldots + \beta nXn + e$$

where Y is the dependent variable, β0 is the intercept, β1 to βn are the coefficients or slopes associated with the predictor variables X1 to Xn, and e is the random error term. The coefficients indicate the change in the dependent variable associated with a one-unit change in the corresponding predictor variable, holding other variables constant.



The multiple linear regression model can be fitted to the data using various techniques such as ordinary least squares (OLS) or maximum likelihood estimation (MLE). The fitted model can be used to make predictions for new values of the predictor variables or to test hypotheses about the relationships between the variables.

The validity of the multiple linear regression model depends on certain assumptions such as linearity, normality, homoscedasticity, and independence of the error terms. Violations of these assumptions can lead to biased and inefficient estimates and incorrect inference.

Pros :

- Multiple linear regression can capture the effects of multiple predictors on the response variable simultaneously, allowing for a more comprehensive analysis of the relationships between the variables.
- It can provide insight into the relative importance of different predictors in explaining the variability in the response variable.
- It can be used to make predictions and estimate the value of the response variable for a given set of predictor variable values.
- It can handle categorical predictor variables by encoding them as dummy variables.

Cons :

- Multiple linear regression assumes that there is a linear relationship between the predictors and the response variable, which may not be the case in reality.
- It assumes that the errors or residuals are normally distributed and have constant variance, which may not hold in practice.
- It can be sensitive to outliers, influential observations, and violations of the underlying assumptions.
- As the number of predictor variables increases, the model can become overfit and less interpretable.

Applications :

- Multiple linear regression is commonly used in economics to model the relationship between a dependent variable such as income, and multiple independent variables such as education, age, and experience.
- It is used in healthcare to study the relationship between health outcomes and multiple risk factors such as smoking, diet, and exercise.
- It is used in marketing to analyze the impact of multiple advertising channels on sales or the relationship between price, promotions, and demand for a product.

Comparison between multiple linear regression and simple linear regression:

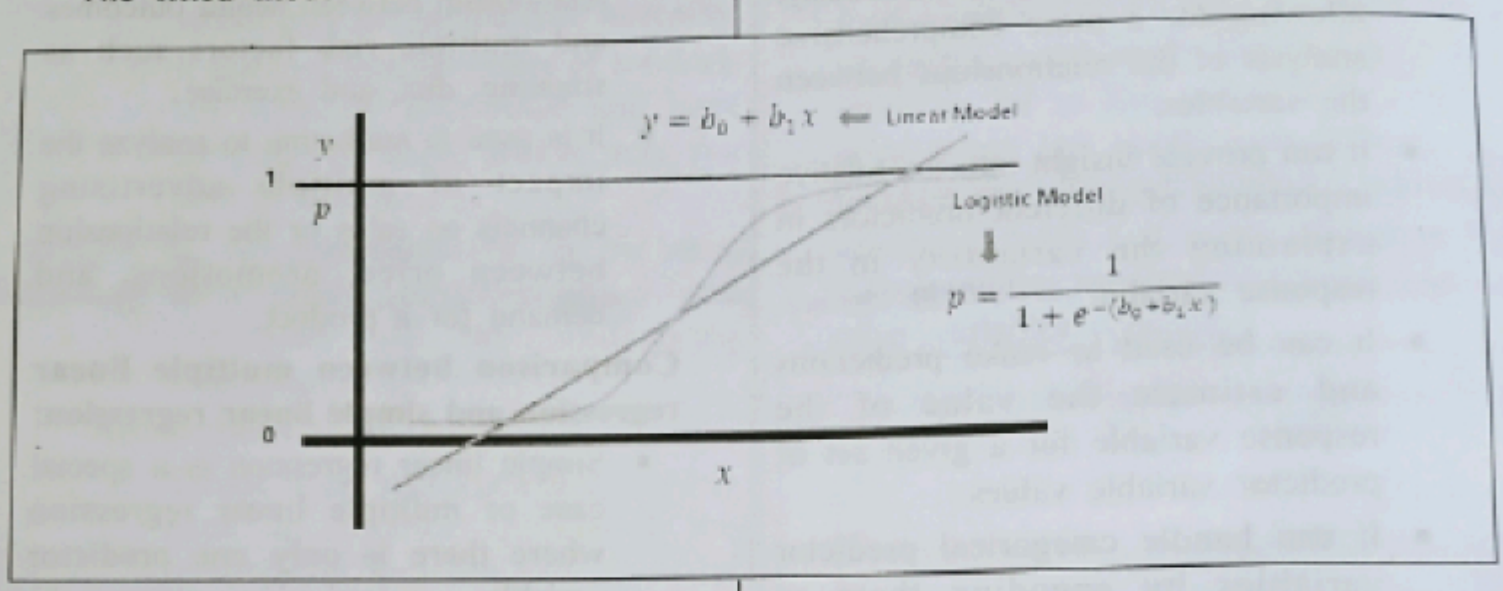
- Simple linear regression is a special case of multiple linear regression where there is only one predictor variable.
- Multiple linear regression can provide more insight into the relationships between the variables and can handle more complex scenarios where there are multiple predictors.
- However, simple linear regression can be easier to interpret and implement, and may be sufficient for simpler problems where there is only one predictor variable.

4.5.3 Logistic Regression

Logistic regression is a statistical technique used to model the relationship between a binary dependent or response variable and one or more independent or predictor variables. The goal of logistic regression is to estimate the probability of the dependent variable taking a particular value (e.g., 1 or 0) given the values of the predictor variables.

The logistic regression model assumes that the relationship between the predictor variables and the log-odds of the dependent variable is linear. The model

can be fitted to the data using various techniques such as maximum likelihood estimation (MLE) or Bayesian methods. The fitted model can be used to make predictions for new values of the predictor variables or to test hypotheses about the relationship between the variables.



In above formula of the logistic regression the constant (b0) moves the curve left and right and the slope (b1) defines the steepness of the curve.

Types of Logistic Regression:

1. Binary Logistic Regression

   The categorical response has only two 2 possible outcomes. Example: Spam or Not.

2. Multinomial Logistic Regression:

   Three or more categories without ordering. Example: Predicting which food is preferred more (Veg, Non-Veg, Vegan).

3. Ordinal Logistic Regression

   Three or more categories with ordering. Example: Movie rating from 1 to 5

Pros of logistic regression :

- Logistic regression is a powerful and widely used statistical technique for binary and ordinal classification problems, and can handle multiple predictor variables.
- The output of logistic regression is easy to interpret as it provides the predicted probabilities of the dependent variable taking a particular value.
- Logistic regression can handle both categorical and continuous predictor variables and can detect interactions between them.
- Logistic regression can be used to test hypotheses about the relationships between the variables and to estimate the impact of the predictor variables on the dependent variable.

Cons of logistic regression :

- Logistic regression assumes that the relationship between the predictor variables and the dependent variable is linear, which may not hold in all cases.
- Logistic regression assumes that the observations are independent, which may not hold in cases where the data are clustered or correlated.
- Logistic regression can be sensitive to outliers and influential observations,

which may affect the estimated coefficients and the goodness-of-fit of the model.

- Logistic regression may suffer from overfitting or underfitting if the model is too complex or too simple, respectively.

Applications of logistic regression:

- Logistic regression is widely used in various fields such as healthcare, marketing, finance, and social sciences for binary and ordinal classification problems such as disease diagnosis, customer churn, credit scoring, and voter prediction.
- Logistic regression is used in biomedical research to model the probability of a disease occurring based on risk factors such as age, sex, and lifestyle factors.
- Logistic regression is used in social sciences to model the likelihood of a particular behavior or attitude given demographic and psychographic characteristics of the individuals.
- Logistic regression is used in marketing to model the likelihood of a customer buying a product given their past behavior and demographic characteristics.

Classification vs. Regression :

Aspect	Classification	Regression
Goal	To predict the categorical outcome	To predict the numerical outcome
Output	Discrete values (classes/labels)	Continuous values (numbers)
Examples	Image classification, spam detection	Stock price prediction, housing prices
Evaluation	Accuracy, precision, recall	Mean squared error, root mean squared error
Algorithms	Logistic regression, decision trees	Linear regression, random forest
Problem types	Binary classification, multi-class	Linear regression, non-linear regression

Question Bank

Short-Answer Questions (3 or 4 marks) :

1. Give any three examples of supervised learning in Industry 4.0
2. Give any three examples of supervised learning in the field of healthcare.
3. Define following terms: Support Vectors, Hyperplane, Margin
4. Explain classification model in brief.
5. Give the difference between classification and regression.
6. Draw the flowchart which shows the classification learning process.
7. Explain different types of logistic regression.

8. Compare and contrast Single linear regression and multiple linear regression.
9. List applications of SVM algorithm.
10. State advantages and disadvantages of k-NN algorithm.

■ **Long-Answer Questions** **(7 marks):**

1. Define Classification. Explain classification learning steps in detail.
2. Write and discuss k-NN Algorithm.
3. Discuss the SVM model in detail with its pros and cons.
4. Explain logistic regression with advantage and disadvantage.
5. Write a short note on Single Linear Regression. Also, state applications of it.
6. Write a short note on Multiple Linear Regression. Also, state applications of it.
7. Explain any three applications of classification in detail.

■ **Check your knowledge :**

1. Which of the following is an example of a regression problem ?
   a) Predicting whether a customer will buy a product or not
   b) Estimating the age of a person based on their photograph
   c) Predicting the stock price of a company
   d) All of above
2. What is the main difference between classification and regression ?
   a) Classification predicts discrete outcomes while regression predicts continuous outcomes
   b) Classification predicts continuous outcomes while regression predicts discrete outcomes
   c) There is no difference between classification and regression
   d) Both are same.
3. Which of the following evaluation metrics is commonly used for regression problems?
   a) Accuracy b) Precision
   c) Mean squared error d) Specificity
   Ans.: c) Mean squared error
4. What is the goal of logistic regression?
   a) To estimate the mean of the dependent variable.
   b) To predict the dependent variable from the independent variables.
   c) To estimate the probability of the dependent variable taking a particular value given the independent variables.
   d) To test the significance of the independent variables.
5. What is the meaning of the parameter k in the kNN classification algorithm?
   a) The number of predictor variables in the model.
   b) The number of training observations used to fit the model.
   c) The number of nearest neighbors to consider when classifying a new observation.
   d) The level of significance for the hypothesis test of the model.

■

5 Unsupervised Learning

5.1 Supervised vs. Unsupervised Learning
5.2 Applications of unsupervised learning
5.3 Clustering
5.3.1 K-means clustering Algorithm
5.4 Finding pattern using Association Rule
5.4.1 Apriori Algorithm
Question Bank

5.1 Supervised vs. Unsupervised Learning

Criteria	Supervised learning	Unsupervised learning
Input Data	Algorithms are trained using labeled data.	Algorithms are used against data that is unlabeled.
Computational Complexity	Simpler method	Computationally complex
Accuracy	Highly accurate	Less accurate
No. of classes	No. of classes is known	No. of classes is not known
Data Analysis	Uses offline analysis	Uses real-time analysis of data
Algorithms used	Linear and Logistics regression, Random forest, Support Vector Machine, Neural Network, etc.	K-Means clustering, Hierarchical clustering, Apriori algorithm, etc.
Output	Desired output is given.	Desired output is not given.
Training data	Use training data to infer model.	No training data is used.
Complex model	It is not possible to learn larger and more complex models than with supervised learning.	It is possible to learn larger and more complex models with unsupervised learning.
Model	We can test our model.	We can not test our model.
Example	Example: Optical character recognition.	Example: Find a face in an image.

Unsupervised Learning:

Unsupervised learning is a type of machine learning algorithm where the input data is not labeled and the algorithm must find patterns or structure within the data on its own.

Unlike supervised learning, there are no target variables to predict, and the algorithm is left to discover patterns and relationships on its own.

5.2 Applications of unsupervised learning

1. Clustering: Unsupervised learning algorithms like k-means and hierarchical clustering are widely used in customer segmentation, image segmentation, document clustering, and social network analysis.

   For example, a clothing retailer may use unsupervised learning to identify three distinct customer segments: young fashion-conscious shoppers, budget-conscious families, and outdoor enthusiasts. They can then tailor their advertising and product offerings to each segment, improving customer engagement and increasing sales.

2. Anomaly detection: Unsupervised learning algorithms like isolation forest and one-class SVM can be used to detect anomalies in data such as fraud detection, intrusion detection, and manufacturing quality control.

   For example, a bank may use unsupervised learning to detect credit card fraud by identifying transactions that are significantly different from a customer's usual spending pattern. They can then flag these transactions for further investigation or automatically block them to prevent fraudulent activity.

3. Dimensionality reduction: Techniques like principal component analysis (PCA), t-SNE, and autoencoders are used for reducing the complexity of high-dimensional data to improve visualization, compression, and feature selection.

4. Association rule learning: Unsupervised learning algorithms like Apriori and FP-Growth can be used for market basket analysis, product recommendation, and customer behavior analysis.

5. Topic modeling: Unsupervised learning algorithms like latent Dirichlet allocation (LDA) and non-negative matrix factorization (NMF) can be used to identify topics in large document collections and social media data.

6. Generative models: Unsupervised learning algorithms like variational autoencoders (VAEs) and generative adversarial networks (GANs) are used for generating synthetic data, image synthesis, and style transfer.

7. Recommendation Systems: Collaborative filtering algorithms like matrix factorization or nearest-neighbor can be used to make personalized recommendations to users based on their past behavior or preferences. This is commonly used in e-commerce, music or video streaming, and social media platforms.

8. Natural Language Processing: Unsupervised learning algorithms like topic modeling or word embeddings can be used to identify patterns and relationships in large text datasets, improving text classification, sentiment analysis, and language translation.

9. Bioinformatics: Unsupervised learning algorithms like clustering or principal component analysis can be used to identify patterns in gene expression data, improve drug discovery, or predict disease risk.

10. Outlier Detection: Clustering or density-based algorithms can be used to identify outliers or anomalies in data that may indicate errors or outliers.

This is used in finance, manufacturing, and healthcare applications.

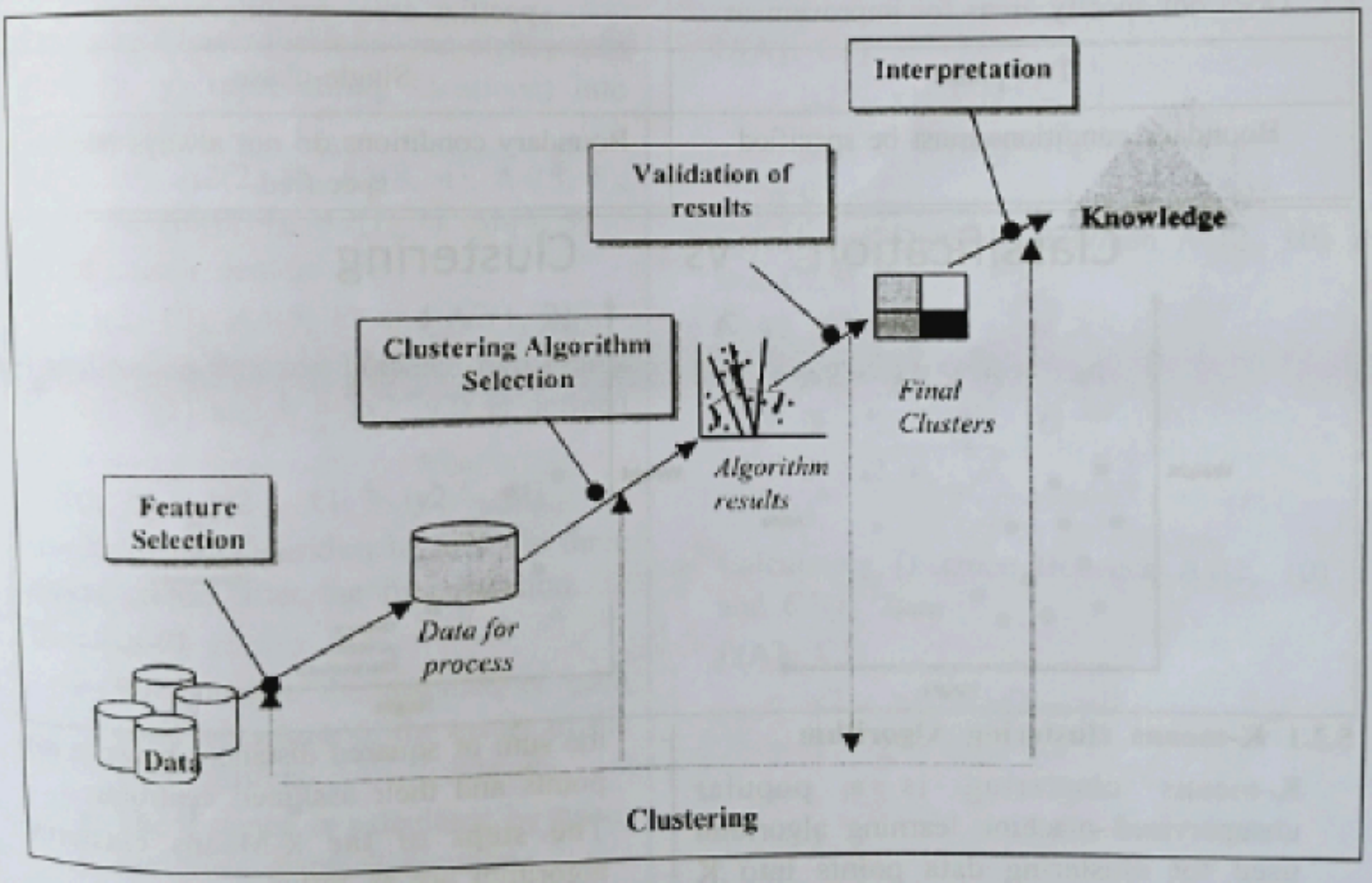
5.3 Clustering

Clustering is a type of unsupervised learning algorithm in machine learning that involves grouping similar data points together into clusters based on their features or attributes. The main objective of clustering is to partition the data in such a way that the points within each cluster are similar to each other and different from points in other clusters.

The choice of clustering algorithm depends on the characteristics of the data and the specific requirements of the application. Clustering algorithms may also require preprocessing steps like normalization or feature scaling to ensure that the features are comparable across different data points.

Overall, clustering is a powerful technique in machine learning that can help uncover hidden patterns in data and facilitate decision-making in various applications.

The following figure shows the steps of clustering process:

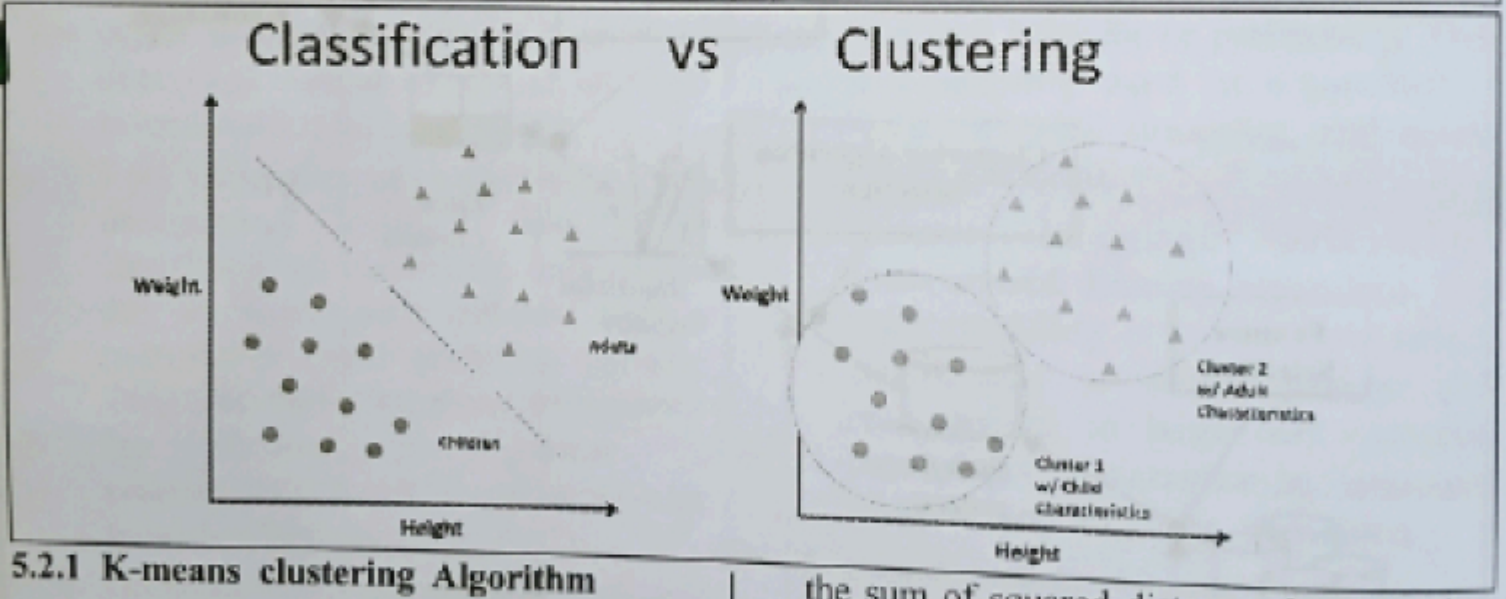


Clustering is very much important as it determines the intrinsic grouping among the unlabelled data present. There are no criteria for good clustering. It depends on the user, what is the criteria they may use which satisfy their need.

Let's see how clustering is differ from classification?

Clustering is the method of unsupervised learning which classification is the method of supervised learning.

	Clustering
Users labelled data as the input	Users unlabelled data as the input
The output is known	The output is unknown
Uses supervised machine learning	Uses unsupervised machine learning
A training data set is provided and used to produce classifications	A training data set is provided and used to produce clusters
Examples of algorithms: Decision-trees, Bayesian Classifiers and Support Vector Machine (SVM)	Examples of algorithms: Partition-based clustering (k-means), Hierarchical clustering (agglomerative & divisive) and DBSCAN
Can be more compex than clustering	Can be less compex than classification
Does not specify areas for improvement	specifies areas for improvement
Two-phase	Single-phase
Boundary conditions must be specified	Boundary conditions do not always need to specified



5.2.1 K-means clustering Algorithm

K-means clustering is a popular unsupervised machine learning algorithm used for clustering data points into K clusters. The algorithm starts with randomly selecting K centroids, and then assigns each data point to the nearest centroid. It then calculates the new centroid of each cluster and repeats the process until convergence. The main objective of the algorithm is to minimize the sum of squared distances between data points and their assigned centroids.

The steps of the k-Means clustering algorithm are as follows:

1. Initialize the algorithm by selecting k random points from the dataset as the initial centroids.
2. Assign each data point to the nearest centroid, based on the Euclidean distance.
3. Calculate the new centroid of each

cluster by taking the mean of all data points assigned to that cluster.

4. Repeat steps 2 and 3 until the centroids no longer change or a specified number of iterations is reached.
5. The resulting clusters are the groups of data points that are closest to their respective centroids.

Pseudo code for the k-means clustering algorithm is:

Algorithm 1 *k*-means algorithm

```
Specify the number k of clusters to assign.
Randomly initialize k centroids.
repeat
    expectation: Assign each point to its closest centroid.
    maximization: Compute the new centroid (mean) of each cluster.
until The centroid positions do not change.
```

Example: Cluster the following eight points (with (x, y) representing locations) into three clusters:

A1(2, 10), A2(2, 5), A3(8, 4), A4(5, 8), A5(7, 5), A6(6, 4), A7(1, 2), A8(4, 9)

Initial cluster centers are :

A1(2, 10), A4(5, 8) and A7(1, 2).

The distance function between two points a = (x1, y1) and b = (x2, y2) is defined as-

$$\tilde{N}(a, b) = |x2 - x1| + |y2 - y1|$$

Use K-Means Algorithm to find the three cluster centers after the first iteration.

Iteration-01 :

- We calculate the distance of each point from each of the center of the three clusters.
- The distance is calculated by using the given distance function.

The following illustration shows the calculation of distance between point A1(2, 10) and each of the center of the three clusters-

Calculating Distance Between A1(2, 10) and C1(2, 10)-

$\tilde{N}$(A1, C1)

$= |x2 - x1| + |y2 - y1|$

$= |2 - 2| + |10 - 10|$

$= 0$

Calculating Distance Between A1(2, 10) and C2(5, 8)-

$\tilde{N}$(A1, C2)

$= |x2 - x1| + |y2 - y1|$

$= |5 - 2| + |8 - 10|$

$= 3 + 2$

$= 5$

Calculating Distance Between A1(2, 10) and C3(1, 2)-

P(A1, C3)

$= |x2 - x1| + |y2 - y1|$

$= |1 - 2| + |2 - 10|$

$= 1 + 8$

$= 9$

In the similar manner, we calculate the distance of other points from each of the center of the three clusters.

Next,

We draw a table showing all the results. Using the table, we decide which point belongs to which cluster.

The given point belongs to that cluster whose center is nearest to it.

Given Points	Distance from center (2, 10) of Cluster-01	Distance from center (5, 8) of Cluster-02	Distance from center (1, 2) of Cluster-03	Point belongs to Cluster
A1(2, 10)	0	5	9	C1
A2(2, 5)	5	6	4	C3
A3(8, 4)	12	7	9	C2
A4(5, 8)	5	0	10	C2
A5(7, 5)	10	5	9	C2
A6(6, 4)	10	5	7	C2
A7(1, 2)	9	10	0	C3
A8(4, 9)	3	2	10	C2

From here, New clusters are-

Cluster-01:

First cluster contains points-

A1(2, 10)

Cluster-02:

Second cluster contains points-

A3(8, 4)

A4(5, 8)

A5(7, 5)

A6(6, 4)

A8(4, 9)

Cluster-03:

Third cluster contains points-

A2(2, 5)

A7(1, 2)

Similarly, we can apply this for iteration 2, 3 and so on.

Pros:

- K-means is fast and scalable, making it ideal for large datasets.
- It is relatively simple to implement and can work well with high-dimensional data.
- K-means is efficient and can handle noisy data.
- It can be used for a wide range of applications, including customer segmentation, image analysis, and text mining.

Cons:

- The performance of K-means depends on the initial placement of centroids, which can lead to suboptimal solutions.
- The algorithm may converge to local optima, which may not be the global optimum.
- K-means assumes that all clusters have the same variance, which may not be the case in some datasets.
- It may not work well with non-linearly separable data.

Applications:

- Customer Segmentation: K-means clustering can be used to group customers based on their behavior, preferences, or needs, allowing businesses to tailor their marketing strategies and offers to specific customer segments.
- Image Segmentation: K-means can be used to segment images by grouping

similar pixels together into clusters, allowing for image compression or object recognition.

- Text Clustering: K-means can be used to group similar text documents together based on their content, improving search results and recommendation systems.
- Anomaly Detection: K-means can be used to identify outliers or anomalies in data that may indicate fraud, intrusion, or equipment malfunction.
- Bioinformatics: K-means can be used to identify patterns in gene expression data or to cluster proteins based on their properties, improving drug discovery or disease diagnosis.

5.4 Finding pattern using Association Rule

The **Association Rule** is a rule-based machine learning method for identifying associations between unrelated elements using pattern recognition.

Support and confidence are two important measures used in association rule mining to evaluate the significance of frequent itemsets and association rules.

Support refers to the frequency of occurrence of an itemset in a dataset, expressed as a percentage or proportion of the total transactions in the dataset that contain the itemset.

Support(A) = (Number of transactions containing A) / (Total number of transactions)

It measures the degree of association between items and is used to identify frequent itemsets that occur frequently enough to be considered interesting or significant.

For example, if a transaction dataset contains 100 transactions and a particular itemset occurs in 20 of them, the support of the itemset would be 20%.

Confidence, on the other hand, refers to the conditional probability that a transaction containing one set of items will also contain another set of items. It measures the strength of the association between items and is used to generate association rules that express the conditional relationships between items.

Confidence(X => Y) = (Number of transactions containing X and Y) / (Number of transactions containing X)

For example, if a transaction dataset contains 100 transactions and a particular association rule has a confidence of 80%, it means that in 80% of the transactions that contain the antecedent (left-hand side) of the rule, the consequent (right-hand side) also occurs.

5.4.1 Apriori Algorithm

Purpose: The Apriori Algorithm is an influential algorithm for mining frequent itemsets for Boolean association rules.

Key Concepts:

Frequent Itemsets: The sets of item which has minimum support (denoted by Li for i^{th}-Itemset).

Apriori Property: Any subset of frequent itemset must be frequent.

Join Operation: To find L_k, a set of candidate k-itemsets is generated by joining Lk-1 itself.

Find the frequent itemsets: the sets of items that have minimum support – A subset of a frequent itemset must also be a frequent itemset (Apriori Property)

i.e., if {AB} is a frequent itemset, both {A} and {B} should be a frequent itemset

– Iteratively find frequent itemsets with cardinality from 1 to k (k-itemset)

Use the frequent itemsets to generate association rules.

The Apriori Algorithm :

The steps of the Apriori algorithm are as follows:

1. Set the minimum support threshold to a desired value.
2. Generate all frequent 1-itemsets by scanning the dataset and counting the support of each item.
3. Repeat the following steps until no more frequent itemsets can be generated:

a. Generate candidate itemsets by joining pairs of frequent (k-1)-itemsets.

b. Prune the candidate itemsets that contain infrequent (k-1)-itemsets.

c. Count the support of each candidate itemset by scanning the dataset.

d. Keep only the frequent itemsets that meet the minimum support threshold.

4. Generate association rules from the frequent itemsets by applying a minimum confidence threshold.

Pseudo code

- Join Step: C k is generated by joining L_k-1with itself
- Prune Step: Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset

Algorithm: *Apriori algorithm*

Input: *D*: Input Dataset
minSup: minimum support threshold

Output: All 2 to *k*-frequent itemsets

```
L1 = {1-frequent itemset} // found separately
for (k = 2; Lk-1 ≠ φ; k++)
  Ck = apriori_gen(Lk-1) // finds k-candidate itemsets by
    joining and pruning Lk-1 with itself
    for each transaction t in D
        Ct = subset (Ck, t) // finds candidate itemsets in t
        for each c in Ct
            c.count++
        end for each
    end for each
    Lk = {c ∈ Ck | c.count ≥ minSup}
end for
Return ∪k Lk
```

Let's see an example of the Apriori Algorithm.

Transaction ID	Items
T1	Hot Dogs, Buns, Ketchup
T2	Hot Dogs, Buns
T3	Hot Dogs, Coke, Chips
T4	Chips, Coke
T5	Chips, Ketchup
T6	Hot Dogs, Coke, Chips

Find the **frequent itemsets** and generate **association rules** on this. Assume that minimum support threshold (s = 33.33%) and minimum confident threshold (c = 60%)

Let's start,

$$\text{minimum support count} = \frac{33.33}{100} \times 6$$
$$= 2$$

Item set	Sup-count
Hot Dogs	4
Buns	2
Ketchup	2
Coke	3
Chips	4

Item set	Sup-count
Hot Dogs	4
Buns	2
Ketchup	2
Coke	3
Chips	4

Item set	Sup-count
Hot Dogs, Buns	2
Hot Dogs, Ketchup	1
Hot Dogs, Coke	2
Hot Dogs, Chips	2
Buns, Ketchup	1
Buns, Coke	0
Buns, Chips	0
Ketchup, Coke	0
Ketchup, Chips	1
Coke, Chips	3

Item set	Sup-count
Hot Dogs, Buns	2
Hot Dogs, Coke	2
Hot Dogs, Chips	2
Coke, Chips	3

Item set	Sup-count
Hot Dogs, Buns, Coke	0
Hot Dogs, Buns, Chips	0
Hot Dogs, Coke, Chips	2

Item set	Sup-count
Hot Dogs, Coke, Chips	2

There is only one itemset with minimum support 2.

So only one itemset is frequent.

Frequent Itemset (I) = {Hot Dogs, Coke, Chips}

Association rules,

- [Hot Dogs^Coke]=>[Chips] // confidence = sup(Hot Dogs^Coke^Chips)/sup(Hot Dogs^Coke) = 2/2*100=100% // **Selected**
- [Hot Dogs^Chips]=>[Coke] // confidence = sup(Hot Dogs^Coke^Chips)/sup(Hot Dogs^Chips) = 2/2*100=100% // **Selected**
- [Coke^Chips]=>[Hot Dogs] // confidence = sup(Hot Dogs^Coke^Chips)/sup(Coke^Chips) = 2/3*100=66.67% //**Selected**
- [Hot Dogs]=>[Coke^Chips] // confidence = sup(Hot Dogs^Coke^Chips)/sup(Hot Dogs) = 2/4*100=50% //**Rejected**
- [Coke]=>[Hot Dogs^Chips] // confidence = sup(Hot Dogs^Coke^Chips)/sup(Coke) = 2/3*100=66.67% //**Selected**
- [Chips]=>[Hot Dogs^Coke] // confidence = sup(Hot Dogs^Coke^Chips)/sup(Chips) = 2/4*100=50% //**Rejected**

There are four strong results (minimum confidence greater than 60%)

Advantages of Apriori Algorithm:

- Scalability : The Apriori algorithm is a scalable algorithm and can handle large datasets efficiently.
- Interpretability : The Apriori algorithm generates frequent itemsets and association rules that are easy to understand and interpret.
- Flexibility : The algorithm can be customized by adjusting the minimum support and confidence thresholds to suit the specific requirements of the user.
- Applicability : The Apriori algorithm can be used to mine association rules from various types of data, including market basket transactions, web logs, and biological sequences.

Disadvantages of Apriori Algorithm:

- Computational Complexity : The Apriori algorithm has a high computational complexity and may take a long time to generate frequent itemsets and association rules from large datasets.
- Memory Requirements : The algorithm requires a significant amount of memory to store the candidate itemsets and their support counts.
- Curse of Dimensionality : The performance of the algorithm may degrade rapidly as the number of items or attributes in the data increases.

Applications of Apriori Algorithm:

- Market Basket Analysis : The Apriori algorithm is commonly used in market basket analysis to identify the co-occurrence of items in customer transactions and generate recommendations for cross-selling and up-selling.
- Web Usage Mining : The algorithm can be used to mine patterns and trends in web logs, such as frequently visited pages or clickstreams, to improve website design and user experience.
- Bioinformatics : The Apriori algorithm can be applied to biological sequences, such as DNA or protein sequences, to discover frequent patterns or motifs that may be related to gene expression or function.
- Fraud Detection : The algorithm can be used to detect fraudulent behavior in financial transactions by identifying patterns of unusual or suspicious activities.
- Social Network Analysis : The algorithm can be applied to social network data to identify groups or communities of individuals with similar interests or behaviors.

Question Bank

■ Short-Answer Questions

(3 or 4 marks) :

1. Give the difference between supervised learning and unsupervised learning.
2. State any four applications of unsupervised learning.
3. Differentiate clustering with classification.
4. Write pseudo code of k-means clustering algorithm.
5. Write strength and weakness of k-means clustering algorithm.
6. Define: Support, Confidence
7. State apriori property.
8. Explain any two applications of apriori algorithm.
9. Write strength and weakness of apriori clustering algorithm.
10. How unsupervised learning is useful in fraud detection?

■ Long-Answer Questions

(7 marks):

1. Write and explain applications of unsupervised learning.
2. Write and explain apriori algorithm in detail.
3. Write and explain k-means clustering approach in detail.
4. You are given a set of one-dimensional data points: {5, 10, 15, 20, 25, 30, 35}. Assume that k = 2 and first set of random centroid is selected as {15, 32} and then it is refined with {12, 30}. Create two clusters with each set of centroid mentioned above following the k-means approach.
5. Generate frequent itemsets and generate association rules based on it using apriori algorithm. Minimum support is 50% and minimum confidence is 70%.

TID	Items
100	1, 3, 4
200	2, 3, 5
300	1, 2, 3, 5
400	2, 5

■ Check your knowledge:

1. Which of the following is a clustering algorithm ?

   a) Linear regression
   b) K-means
   c) Naive Bayes
   d) SVM

2. Which of the following is an example of an unsupervised learning problem?
   a) Classifying images of animals into different categories
   b) Predicting the price of a house
   c) Clustering customers based on their purchasing behavior
   d) All of above

   hi

3. Apriori algorithm is used for:
   a) Clustering
   b) Classification
   c) Association rule mining
   d) Regression

4. What is the minimum support threshold in Apriori algorithm ?
   a) The minimum number of transactions that an itemset must appear in.
   b) The maximum number of transactions that an itemset must appear in.
   c) The minimum frequency with which an item must appear in the transaction database.
   d) The maximum frequency with which an item must appear in the transaction database.

5. What is the main drawback of the k-means algorithm?
   a) It is only effective for datasets with a small number of dimensions.
   b) It can get stuck in local optima.
   c) It requires the number of clusters to be specified in advance.
   d) It is not effective for datasets with categorical variables.

■

6 Python libraries for Machine Learning

6.1 Pandas
6.2 Matplotlib
6.3 Numpy
6.4 Scikit-learn
■ Question Bank

When it comes to machine learning and deep learning projects written in Python, there are thousands of libraries to pick and choose from. However, they're not all on the same level of code quality, diversity, or size. To help you choose, here are the best Python libraries for machine learning. Here we are going to discuss about Pandas, Numpy, Matplotlib and Scikit-learn.

6.1 Pandas

Features of Pandas:

1. Data structures: Pandas provides two main data structures - Series (1D labeled array) and DataFrame (2D labeled data structure) - that allow for easy handling and manipulation of data. These data structures provide powerful and flexible ways to represent and analyze data.
2. Data cleaning: Pandas offers various functionalities for data cleaning, such as handling missing values, removing duplicates, and data type conversion. These operations are crucial for preparing data for machine learning models, as clean and well-structured data is essential for accurate model training.
3. Data transformation: Pandas provides functions for transforming data, such as filtering, sorting, and grouping. These operations are commonly used for data preprocessing and feature engineering in machine learning workflows.
4. Data visualization: Pandas integrates well with other popular data visualization libraries in Python, such as Matplotlib and Seaborn, allowing for easy visualization of data to gain insights and perform exploratory data analysis (EDA).
5. Data I/O: Pandas offers functions to read and write data in various formats, such as CSV, Excel, and SQL, which are commonly used for loading and storing data in machine learning workflows.
6. Flexible indexing and slicing: Pandas allows for flexible indexing and slicing of data, making it easy to extract, filter, and manipulate subsets of data based on various conditions and criteria.
7. Data alignment: Pandas automatically aligns data based on labeled indexes, making it easy to perform operations on data with different shapes and

sizes, and handle missing data gracefully.

8. Handling time-series data: Pandas provides built-in support for handling time-series data, making it easy to work with data that has a time component, such as financial or sensor data.
9. Memory-efficient operations: Pandas provides efficient operations for data manipulation, such as vectorized operations and optimized algorithms, making it suitable for handling large datasets.
10. Interoperability: Pandas seamlessly integrates with other popular Python libraries such as NumPy, Scikit-learn, and more, making it a versatile tool for data analysis and machine learning workflows.Bottom of Form

Application of Pandas:

1. Data preprocessing: Pandas provides functions for handling missing values, converting data types, removing duplicates, and handling outliers, which are crucial steps in preparing data for machine learning models.

For example:

```
import pandas as pd
# Load data into a DataFrame
data = pd.read_csv('data.csv')
# Handling missing values
data.dropna() # Drop rows with missing values
data.fillna(value) # Fill missing values with a specified value
# Data type conversion
data['column_name'] = data['column_name'].astype('data_type')
# Removing duplicates
data.drop_duplicates() # Drop duplicate rows
```

2. Feature engineering: Pandas provides powerful functions for data transformation, such as filtering, sorting, grouping, and merging, which are commonly used for feature engineering.

For example:

```
# Filtering data
filtered_data = data[data['column_name'] > threshold]
# Grouping data
grouped_data = data.groupby('column_name').mean()
# Merging data
merged_data = pd.merge(data1, data2, on='common_column', how='inner')
```

3. Data analysis: Pandas makes it easy to perform exploratory data analysis (EDA) and gain insights from data before building machine learning models. For example:

```
# Descriptive statistics
data.describe() # Get summary statistics of data
# Data visualization
importmatplotlib.pyplot as plt
data.plot(x='column_name', y='column_name', kind='scatter')
# Create scatter plot
plt.show()
# Pivot tables
pivot_table = data.pivot_table(values='value_column', index='index_column', columns='column_name')
```

4. Data preparation for model training: Pandas allows for easy data slicing,

indexing, and splitting, which are essential for preparing data for model training. For example:

```
# Data slicing and indexing
X = data[['feature1', 'feature2']] # Select specific features as input (X)
y = data['target'] # Select target variable (y)
# Data splitting
fromsklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state
```

6.2 Matplotlib

Matplotlib is a popular data visualization library in Python that provides a wide range of functionalities for creating static, animated, and interactive plots. Some of the key features of Matplotlib include:

Features of Matplotlib:

1. Comprehensive Plotting: Matplotlib provides a vast collection of plot types, including line plots, scatter plots, bar plots, histogram, pie charts, 3D plots, and more. It offers extensive customization options for colors, markers, linestyles, labels, titles, legends, and axes.
2. Multi-platform Support: Matplotlib supports various output formats, including PNG, JPEG, PDF, SVG, and more. It also provides an interactive mode for embedded applications and supports various GUI backends, such as Tkinter, Qt, GTK, and more, making it suitable for different platforms and environments.
3. Data Visualization: Matplotlib allows for visualizing complex data through different plotting techniques, such as subplots, grids, multiple axes, shared axes, and dual axes. It also provides support for adding text, annotations, arrows, images, and other visual elements to enhance the plot's visual appeal and readability.
4. Customization: Matplotlib provides extensive customization options, allowing users to customize the appearance of plots based on their specific needs. Users can customize plot properties, including figure size, axes limits, ticks, labels, fonts, colors, and more, to create visually appealing and informative plots.
5. Plotting Styles: Matplotlib provides support for various predefined plotting styles, including classic, ggplot, seaborn, and more, allowing users to easily switch between different styles and customize the visual appearance of their plots.
6. Object-oriented and Pyplot Interfaces: Matplotlib offers two interfaces for creating plots: the object-oriented interface and the pyplot interface. The object-oriented interface provides more flexibility and control over plot creation and customization, while the pyplot interface provides a simple and intuitive way to create plots using a MATLAB-like syntax.
7. Animation and Interactivity: Matplotlib provides support for creating animated plots, allowing users to visualize data changes over time. It also provides support for adding interactivity to plots, allowing users to zoom, pan, and interact with plots to explore the data in a more dynamic way.

8. Integration with Other Libraries: Matplotlib integrates well with other popular Python libraries such as NumPy, Pandas, and Scikit-learn, making it a powerful tool for visualizing data in machine learning workflows and data analysis tasks.

Applications of Matplotlib:

1. Exploratory Data Analysis (EDA): Matplotlib can be used to visualize and explore the data before building machine learning models. This can include creating scatter plots, histograms, box plots, bar plots, and other types of plots to gain insights into the distribution, trends, and relationships of the data. EDA with Matplotlib helps in identifying patterns, outliers, and potential issues in the data, which can inform feature engineering and model selection decisions.
2. Model Evaluation: Matplotlib can be used to visualize the performance of machine learning models by plotting various evaluation metrics, such as accuracy, precision, recall, F1-score, and ROC curves. These visualizations can help in comparing the performance of different models, selecting the best-performing model, and identifying potential issues, such as overfitting or underfitting.
3. Feature Visualization: Matplotlib can be used to visualize the importance or contribution of features in a machine learning model. This can include creating bar plots or heatmaps to display feature importances, coefficients, or weights from models such as linear regression or logistic regression. These visualizations can help in understanding the importance of different features in the model's prediction and interpreting the model's behavior.
4. Model Interpretability: Matplotlib can be used to create visualizations that help interpret complex machine learning models, such as decision trees, random forests, or support vector machines. These visualizations can help in understanding the decision-making process of the model, visualizing decision boundaries, and identifying critical features or instances that influence the model's prediction.
5. Model Explainability: Matplotlib can be used to create visualizations that explain the prediction of machine learning models, such as partial dependence plots, individual conditional expectation plots, or LIME plots. These visualizations can help in explaining how the model's prediction changes with respect to specific features or instances and provide insights into the model's behavior and predictions for interpretability and transparency.
6. Model Deployment: Matplotlib can be used to create visualizations for model deployment in production systems, such as creating interactive plots, dashboards, or reports to monitor model performance, visualize real-time predictions, or provide insights to stakeholders.

6.3 Numpy

Numpy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of

high-level mathematical functions to operate on these arrays. Moreover Numpy forms the foundation of the Machine Learning stack.

NumPy (Numerical Python) is a powerful numerical computing library in Python that provides essential functionalities for performing numerical operations efficiently.

Features of Numpy:

Some of the key features of NumPy include:

1. N-dimensional array: NumPy provides a powerful N-dimensional array object called **ndarray**, which is a homogeneous collection of elements with fixed-size and enables efficient operations on large arrays of data. It allows for easy manipulation and computation of multi-dimensional data, such as matrices, images, and time series.
2. Broadcasting: NumPy allows for element-wise operations on arrays of different shapes and sizes through broadcasting, which automatically handles shape and size compatibility. This enables efficient and concise operations on arrays with different dimensions, eliminating the need for explicit loops.
3. Mathematical Functions: NumPy provides a vast collection of mathematical functions for performing operations such as arithmetic, trigonometry, logarithm, exponential, linear algebra, statistical analysis, and more. These functions are optimized for performance and provide efficient computations for numerical operations.
4. Array Manipulation: NumPy provides a wide range of functions for manipulating arrays, including reshaping, resizing, stacking, splitting, transposing, and more. These functions allow for efficient manipulation of array data, making it easy to preprocess and transform data for machine learning and other numerical computations.
5. Broadcasting: NumPy allows for element-wise operations on arrays of different shapes and sizes through broadcasting, which automatically handles shape and size compatibility. This enables efficient and concise operations on arrays with different dimensions, eliminating the need for explicit loops.
6. Vectorized Operations: NumPy allows for vectorized operations, which enable efficient computation on entire arrays without the need for explicit loops. This makes NumPy code more concise, readable, and optimized for performance.
7. Interoperability: NumPy provides support for interoperability with other libraries and data formats. It can seamlessly integrate with other Python libraries, such as pandas, Matplotlib, and scikit-learn, making it a fundamental building block for scientific computing and data analysis in Python.
8. Performance Optimization: NumPy is designed for efficient numerical computing and provides optimized implementations of mathematical functions that are written in C or Fortran. It also allows for vectorized operations and provides support for parallel computing, making it highly

efficient for processing large datasets and performing computationally-intensive tasks.

9. Random Number Generation: NumPy provides functions for generating random numbers, including various probability distributions, random sampling, and random data generation. This is useful for simulations, statistical analysis, and machine learning applications that require randomization.

Applications of Numpy:

1. Scientific Computing: NumPy provides a powerful array object, **ndarray**, which allows for efficient handling of large numerical data, such as matrices, vectors, and tensors. It provides essential functions for performing mathematical operations on arrays, such as linear algebra, Fourier analysis, numerical integration, interpolation, and more, making it widely used in scientific computing for tasks such as simulations, data analysis, and modeling.
2. Data Analysis: NumPy is often used in combination with other data analysis libraries, such as pandas, for data manipulation, preprocessing, and analysis tasks. NumPy provides efficient array operations for handling large datasets, numerical computations, and mathematical operations, making it a foundational library for data analysis tasks, including data cleaning, feature engineering, statistical analysis, and more.
3. Machine Learning: NumPy is a fundamental library used in machine learning tasks for data preprocessing, feature engineering, and model evaluation. It provides efficient operations for manipulating and transforming data in arrays, such as reshaping, slicing, and stacking, which are commonly used in machine learning workflows. NumPy is also used in implementing numerical algorithms for machine learning models, such as linear regression, logistic regression, support vector machines, and deep learning models.
4. Image and Signal Processing: NumPy provides functions for image and signal processing tasks, such as image manipulation, filtering, Fourier analysis, and convolution operations. It is widely used in computer vision, image processing, and signal processing applications for tasks such as image filtering, feature extraction, and image reconstruction.
5. Computational Mathematics: NumPy is used in various computational mathematics applications, including numerical optimization, numerical integration, numerical solving of equations, and more. It provides efficient array operations, mathematical functions, and linear algebra operations, which are essential in solving numerical problems in various fields, such as physics, engineering, and finance.
6. Data Visualization: NumPy is often used in conjunction with data visualization libraries, such as Matplotlib and Seaborn, for creating visualizations of numerical data. NumPy provides efficient operations for data manipulation, aggregation, and computation, which can be used to prepare data for visualization and

perform calculations for generating plots, charts, and graphs.

6.4 Scikit-learn

Scikit-learn is an open-source Python library that implements a range of machine learning, pre-processing, cross-validation, and visualization algorithms using a unified interface.

Scikit-learn, a popular machine learning library in Python, offers a wide range of features that make it a powerful tool for various machine learning tasks.

Features of Scikit-learn :

Some of the key features of scikit-learn are:

1. Broad Range of Machine Learning Algorithms: Scikit-learn provides a diverse collection of machine learning algorithms for classification, regression, clustering, dimensionality reduction, ensemble methods, and more. It includes popular algorithms such as linear regression, logistic regression, decision trees, support vector machines, k-nearest neighbors, random forests, gradient boosting, and many more.
2. Consistent API: Scikit-learn follows a consistent API for all its algorithms, making it easy to learn and use. The "fit-transform-predict" paradigm provides a consistent approach for training models, transforming data, and making predictions. This consistent API allows for easy switching between different algorithms and models.
3. Data Preprocessing: Scikit-learn provides tools for data preprocessing, including handling missing values, feature scaling, feature encoding, and feature selection. These tools help in preparing data for machine learning algorithms by handling common data-related issues.
4. Model Selection and Evaluation: Scikit-learn offers tools for model selection and evaluation, including techniques such as cross-validation, grid search, and randomized search for hyperparameter tuning. It also includes tools for evaluating model performance using various metrics, such as accuracy, precision, recall, F1-score, and more.
5. Feature Extraction and Dimensionality Reduction: Scikit-learn provides methods for feature extraction, such as Principal Component Analysis (PCA) and feature selection techniques like SelectKBest, SelectPercentile, and Recursive Feature Elimination (RFE). These techniques help in reducing the dimensionality of the data and selecting relevant features for machine learning models.
6. Model Persistence: Scikit-learn allows for saving trained models to disk and reloading them later for making predictions on new data. This makes it easy to reuse models in production environments without the need to retrain them every time.
7. Integrated Data Visualization: Scikit-learn integrates well with other popular Python libraries for data visualization, such as Matplotlib and Seaborn, allowing for visual exploration and analysis of data.
8. Robust Documentation and Community Support: Scikit-learn has extensive documentation and a large community of users and developers, providing resources and support for beginners

and advanced users alike. The library is actively maintained and regularly updated with new features and bug fixes.

9. Scalability: Scikit-learn is designed to be scalable and efficient, making it suitable for handling large datasets and running on various computing platforms, including desktops, servers, and distributed clusters.

Applications of Scikit-learn:

1. Classification: Scikit-learn provides a wide range of algorithms for classification tasks, such as logistic regression, support vector machines, decision trees, random forests, and naive Bayes, among others. These algorithms can be used for tasks like spam detection, fraud detection, sentiment analysis, and image classification.
2. Regression : Scikit-learn offers algorithms for regression tasks, such as linear regression, ridge regression, and support vector regression. These algorithms can be used for tasks like predicting housing prices, stock prices, and other continuous variables.
3. Clustering : Scikit-learn provides algorithms for unsupervised learning tasks, such as clustering, including k-means, hierarchical clustering, and DBSCAN. These algorithms can be used for tasks like customer segmentation, anomaly detection, and image segmentation.
4. Dimensionality Reduction: Scikit-learn includes methods for dimensionality reduction, such as Principal Component Analysis (PCA), which can be used for tasks like reducing the dimensionality of high-dimensional data for visualization, feature extraction, and model building.
5. Model Selection and Evaluation: Scikit-learn offers tools for model selection and evaluation, such as cross-validation, grid search, and randomized search for hyperparameter tuning. These tools help in selecting the best model and optimizing its performance.
6. Ensemble Methods: Scikit-learn includes ensemble methods, such as Random Forests, Gradient Boosting, and Bagging, which can be used for tasks like improving the accuracy and robustness of models.
7. Preprocessing: Scikit-learn provides tools for data preprocessing, such as handling missing values, feature scaling, and feature encoding. These tools help in preparing data for machine learning algorithms.
8. Text Mining: Scikit-learn includes text mining tools, such as text feature extraction techniques, vectorizers, and natural language processing (NLP) tools, which can be used for tasks like text classification, sentiment analysis, and topic modeling.
9. Model Deployment: Scikit-learn models can be easily deployed in production environments for making real-time predictions on new data, making it suitable for building machine learning applications.

Difference between numpy, pandas, matplotlib and scikit-learn

Library	Purpose	Main Features
Numpy	Numerical computing	- Provides support for handling large, multi-dimensional arrays and matrices. - Offers mathematical functions for operations on arrays, such as linear algebra and statistics.
Pandas	Data manipulation and analysis	- Provides data structures like Series and DataFrame for efficient data handling. - Offers powerful tools for data cleaning, aggregation, filtering, and transformation.
Matplotlib	Data visualization	- Provides a wide range of plotting and visualization options for creating static visualizations. - Supports various types of plots, including line plots, scatter plots, histograms, and more.
Scikit-learn	Machine learning and data modeling	- Offers a wide range of machine learning algorithms for classification, regression, clustering, and dimensionality reduction tasks. - Provides tools for model selection, evaluation, and preprocessing of data. - Includes utilities for model evaluation, metrics, and model deployment.

Example of classification using numpy, pandas, Matplotlib, Scikit-learn.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
fromsklearn.model_selection import train_test_split
fromsklearn.linear_model import LogisticRegression
fromsklearn.metrics import accuracy_score, confusion_matrix
# Load data into a Pandas DataFrame
data = pd.read_csv('data.csv')

# Data preprocessing
X = data[['feature1', 'feature2']] # Select input features (X)
y = data['target'] # Select target variable (y)
```

```
# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Model training
clf = LogisticRegression() # Instantiate a logistic regression model
clf.fit(X_train, y_train) # Fit the model to the training data

# Model prediction
y_pred = clf.predict(X_test) # Predict the target variable for the test data

# Model evaluation
accuracy = accuracy_score(y_test, y_pred) # Calculate accuracy
confusion = confusion_matrix(y_test, y_pred) # Calculate confusion matrix
# Data visualization
plt.scatter(X_test['feature1'], X_test['feature2'], c=y_pred) # Scatter plot of predicted data
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.title('Classification Results')
plt.show( )
```

Question Bank

- **Short-Answer Questions** **(3 or 4 marks) :**

1. Give the purpose of Numpy in Machine Learning.
2. Compare and contrast Pandas and Numpy.
3. What is the main purpose of matplotlib in Machine Learning ?
4. What are the advantages of using Scikit-learn?
5. How to load dataset using Numpy ? Explain.
6. How to load dataset using Panda ? Explain.
7. How to plot a vertical line and a horizontal line using Matplotlib ?

- **Long-Answer Questions** **(7 marks):**

1. Explain features and applications of Pandas.
2. Explain features and applications of Numpy.
3. Explain features and applications of Matplotlib.
4. Explain features and applications of Scikit-Learn.
5. Write a Python program to load the iris data from a given csv file into a dataframe and print the shape of the data, type of the data and first 3 rows using Scikit-Learn.

Check your knowledge :

1. What is a DataFrame in pandas ?
   a) A type of function
   b) A two-dimensional table-like data structure with rows and columns
   c) A way to iterate through a pandas object
   d) None of above
2. How can you create a NumPy array ?
   a) Using the array() function b) Using the list() function
   c) Using the dict() function d) All of above
3. How can you add a label to the x-axis of a plot in Matplotlib ?
   a) Using the xlabel() function b) Using the title() function
   c) Using the ylabel() function
   d) Using the xplot() function
4. How can you split a dataset into training and testing sets in Scikit-learn ?
   a) Using the split() function
   b) Using the train_test_split() function
   c) Using the shuffle() function
   d) Using the train_test_shuffle() function
5. What is pandas?
   a) A type of bear
   b) A Python library used for data manipulation and analysis
   c) A programming language
   d) All of above

■

7 Lab Manual for Machine Learning

Practical : 1
AIM : Numerical Computing with Python (NumPy, Matplotlib)

Practical : 2
AIM : Introduction to Pandas for data import and export (Excel, CSV etc.)

Practical : 3
AIM : Basic Introduction to Scikit learn

Practical : 4
AIM : Implement the Find-S concept learning algorithm that finds the most specific hypothesis that is consistent with the given training data.

Practical : 5
AIM : Import Pima Indian diabetes data. Apply select K best and chi2 for feature selection. Identify the best features

Practical : 6
AIM : Write a program to learn a decision tree and use it to predict class labels of test data
Training and test data will be explicitly provided by instructor. Tree pruning should not be performed.

Practical : 7
AIM : ML Project. Use the following dataset as music.csv

Practical : 8
AIM : Write a program to use a K-nearest neighbor it to predict class labels of test data. Training and test data must be provided explicitly.

Practical : 9
AIM : Import vgsales.csv from kaggle platform.
a. Find rows and columns in dataset
b. Find basic information regarding dataset using describe command.
c. Find values using values command.

Practical : 10
AIM : Project on regression
a. Import home_data.csv on kaggle using pandas
b. Understand data by running head ,info and describe command
c. Plot the price of house with respect to area using matplotlib library
d. Apply linear regression model to predict the price of house

Practical : 11
AIM : Write a program to cluster a set of points using K-means. Training and test data must be provided explicitly.

Practical : 12
AIM : Import Iris dataset
a. Find rows and columns using shape command
b. Print first 30 instances using head command
c. Find out the data instances in each class.(use groupby and size)
e. Plot the univariategraphs(box plot and histograms)
f. Plot the multivariate plot(scatter matrix)
g. Split data to train model by 80% data values
h. Apply K-NN and k means clustering to check accuracy and decide which is better.

PRACTICAL : 1

AIM: Numerical Computing with Python (NumPy, Matplotlib)

Introduction:

Numerical computing with Python refers to the use of Python programming language for performing mathematical and scientific computations. Python is a high-level programming language that is easy to learn, has a clear and concise syntax, and is widely used in scientific computing.

Python provides several libraries that are useful for numerical computing, including:

1. NumPy: NumPy is a library for numerical computing in Python. It provides an array object, which is faster and more efficient than traditional Python lists for performing mathematical operations. NumPy also provides functions for linear algebra, Fourier transforms, and random number generation.
2. SciPy: SciPy is a library for scientific computing in Python. It provides functions for optimization, signal processing, and statistics, among other things.
3. Matplotlib: Matplotlib is a library for creating data visualizations in Python. It provides a variety of plots, including line plots, scatter plots, and histograms.
4. Pandas: Pandas is a library for data analysis in Python. It provides data structures for working with structured data, such as tables and time series.
5. SymPy: SymPy is a library for symbolic mathematics in Python. It provides tools for working with mathematical expressions and equations symbolically, rather than numerically.

Numerical computing with Python is used in a wide range of scientific and engineering applications, including data analysis, image processing, and simulation. It is also used in machine learning and deep learning applications, where numerical computations are performed on large datasets.

Numpy

→ **Installing Numpy**

pip install numpy

→ **Importing Numpy Library**

importnumpy as np

→ **Creating arrays : NumPy provides several functions for creating arrays, including np.array, np.zeros, np.ones, and np.random.rand. For example, to create an array of zeros with shape (3, 3), you can use:**

Code:

```
importnumpy as np
a = np.zeros((3, 3))
```

Output:

```
In [2]: import numpy as np
        a = np.zeros((3, 3))
        a

Out[2]: array([[0., 0., 0.],
               [0., 0., 0.],
               [0., 0., 0.]])
```

→ **Array indexing and slicing: NumPy arrays can be indexed and sliced like regular Python lists, but also support more advanced indexing and slicing operations. For example, to extract the first row of a 2-dimensional array a, you can use:**

Code:

row = a[0, :]

Output:

```
In [4]: row = a[0, :]
        row

Out[4]: array([0., 0., 0.])
```

→ **Mathematical functions: NumPy provides a range of mathematical functions for performing operations on arrays, such as np.add, np.subtract, np.multiply, np.divide, np.power, and np.sqrt. For example, to compute the element-wise square root of an array a, you can use:**

Code:

x=np.array([4,9,16,25])

b = np.sqrt(x)

Output:

```
In [6]: x=np.array([4,9,16,25])
        b = np.sqrt(x)
        b

Out[6]: array([2., 3., 4., 5.])
```

→ **Aggregation functions: NumPy provides functions for aggregating arrays, such as np.sum, np.mean, np.median, np.std, and np.max. For example, to compute the sum of all elements in an array a, you can use:**

x=np.array([4,9,16,25])

total = np.sum(x)

Output:

```
In [9]: x=np.array([4,9,16,25])
        total = np.sum(x)
        total

Out[9]: 54
```

→ **Linear algebra functions: NumPy provides functions for performing linear algebra operations, such as matrix multiplication (np.dot), matrix inversion (np.linalg.inv), and eigendecomposition (np.linalg.eig). For example, to compute the dot product of two matrices A and B, you can use:**

Code:

```
x=np.array([4,9,16,25])
y=np.array([1,2,3,4])
C = np.dot(x,y)
```

Output

```
In [11]: x=np.array([4,9,16,25])
         y=np.array([1,2,3,4])
         C = np.dot(x,y)
         C

Out[11]: 170
```

Matplotlib

→ **Installing Library**

pip install matplotlib

→ **Importing Library**

importmatplotlib.pyplot as plt

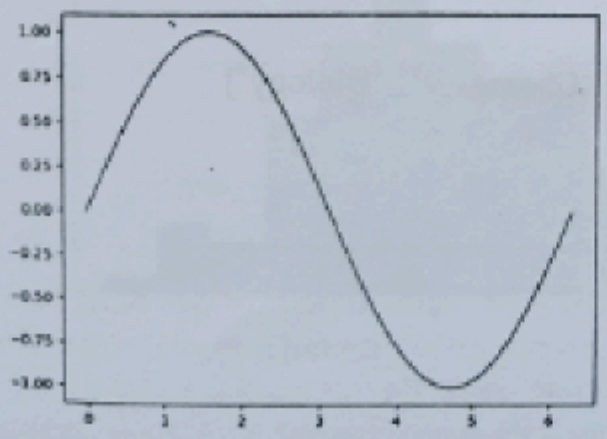
→ **Creating a line plot: Line plots are a simple way to visualize a sequence of data points. To create a line plot in Matplotlib, you can use the plot function. For example, to plot a sine wave from 0 to 2δ, you can use:**

Code:

```
importnumpy as np
importmatplotlib.pyplot as plt
x = np.linspace(0, 2*np.pi, 100)
y = np.sin(x)
plt.plot(x, y)
plt.show()
```

Output:

```
In [14]: import numpy as np
         import matplotlib.pyplot as plt
         x = np.linspace(0, 2*np.pi, 100)
         y = np.sin(x)
         plt.plot(x, y)
         plt.show()
```

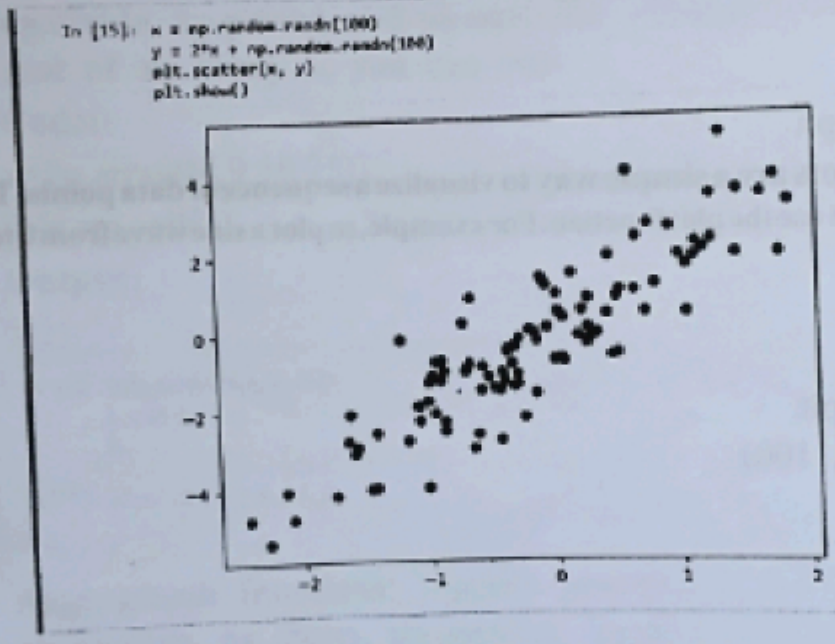


→ **Creating a scatter plot: Scatter plots are a useful way to visualize the relationship between two variables. To create a scatter plot in Matplotlib, you can use the scatter function. For example, to plot the relationship between two variables x and y, you can use:**

Code:

```
x = np.random.randn(100)
y = 2*x + np.random.randn(100)
plt.scatter(x, y)
plt.show()
```

Output:



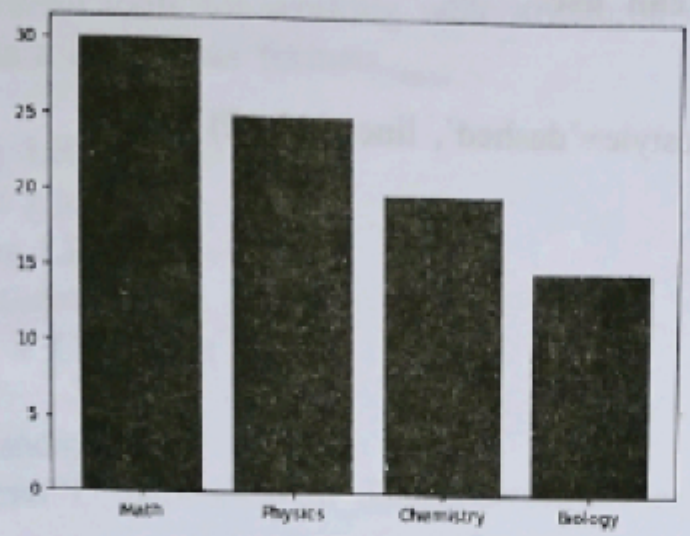
→ **Creating a bar plot: Bar plots are a useful way to compare the values of different categories. To create a bar plot in Matplotlib, you can use the bar or barh function. For example, to plot the number of students who prefer different subjects, you can use:**

Code:

```
subjects = ['Math', 'Physics', 'Chemistry', 'Biology']
students = [30, 25, 20, 15]
plt.bar(subjects, students)
plt.show()
```

Output:

```
In [16]: subjects = ['Math', 'Physics', 'Chemistry', 'Biology']
         students = [30, 25, 20, 15]
         plt.bar(subjects, students)
         plt.show()
```



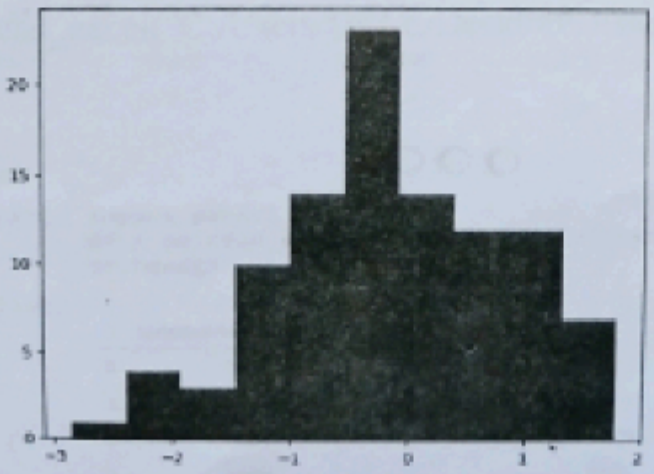
→ **Creating a histogram: Histograms are a useful way to visualize the distribution of a dataset. To create a histogram in Matplotlib, you can use the hist function. For example, to plot the distribution of exam scores, you can use:**

Code:

```
scores = np.random.randn(100)
plt.hist(scores, bins=10)
plt.show()
```

Output:

```
In [17]: scores = np.random.randn(100)
         plt.hist(scores, bins=10)
         plt.show()
```

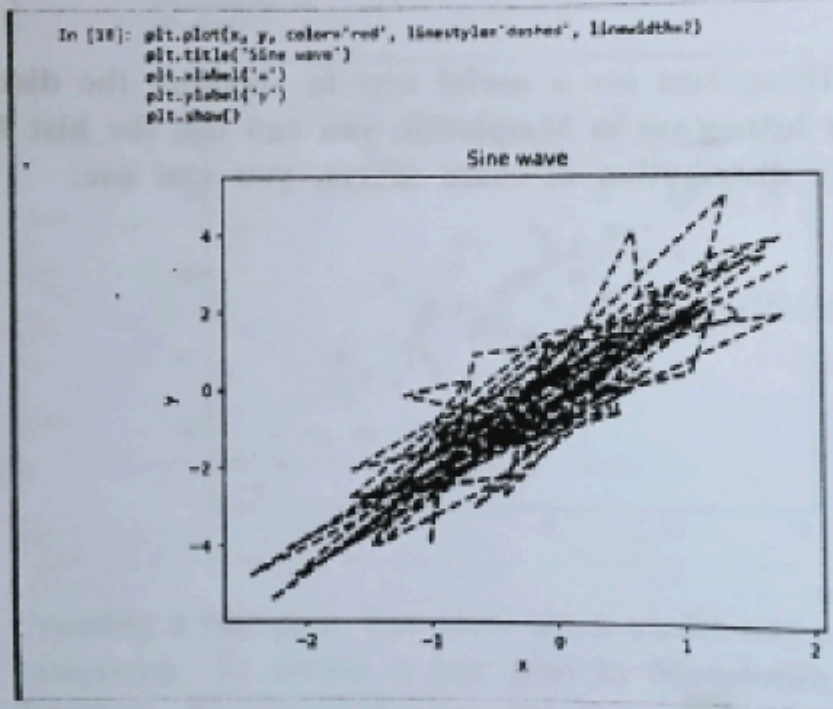


→ **Customizing plots: Matplotlib provides a range of options for customizing plots, such as changing the colors and styles of lines, adding labels and titles, and changing the size and aspect ratio of the plot. For example, to add a title and axis labels to a plot, you can use:**

Code:

```
plt.plot(x, y, color='red', linestyle='dashed', linewidth=2)
plt.title('Sine wave')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

Output:



○○○

PRACTICAL : 2

AIM: Introduction to Pandas for data import and export (Excel, CSV etc.)

Pandas is a popular library in Python for data analysis and manipulation. It provides a range of powerful tools for working with structured data, including the ability to import and export data in various formats.

Pandas

→ **Installing Library**

pip install pandas

→ **Importing Library**

import pandas as pd

→ **Reading a CSV file:**

Code:

```
import pandas as pd
df=pd.read_csv("D:\photos\Covid_dataset\country_wise_latest.csv")
df.head()
```

Output:

```
In [1]: import pandas as pd
        df = pd.read_csv("D:\photos\Covid_dataset\country_wise_latest.csv")
        df.head()
```

Out[1]:

	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths / 100 Cases	Recovered / 100 Cases	Deaths / 100 Recovered	Confirmed last week	1 week change	1 week % increase	WHO Regi
0	Afghanistan	36263	1269	25198	9796	106	10	18	3.50	69.49	5.04	35526	737	2.07	East Mediterrane
1	Albania	4880	144	2745	1991	117	6	63	2.95	56.25	5.25	4171	709	17.00	Euro
2	Algeria	27973	1163	18837	7973	616	8	749	4.16	67.34	6.17	23691	4282	18.07	Afri
3	Andorra	907	52	803	52	10	0	0	5.73	88.53	6.48	884	23	2.60	Euro
4	Angola	950	41	242	667	18	1	0	4.32	25.47	16.94	749	201	26.84	Afri

→ **Reading an Excel file:**

Code:

```
import pandas as pd
df = pd.read_excel("C:/Users/DELL/Desktop/Historicalinvesttemp.xlsx");
df.head()
```

Output:

```
In [17]: import pandas as pd
         df = pd.read_excel("C:/Users/DELL/Desktop/Historicalinvesttemp.xlsx");
         df.head()
```

Out[17]:

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3
0	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN
4	NaN	Annual Returns on Investments in	NaN	NaN

Required Dependency

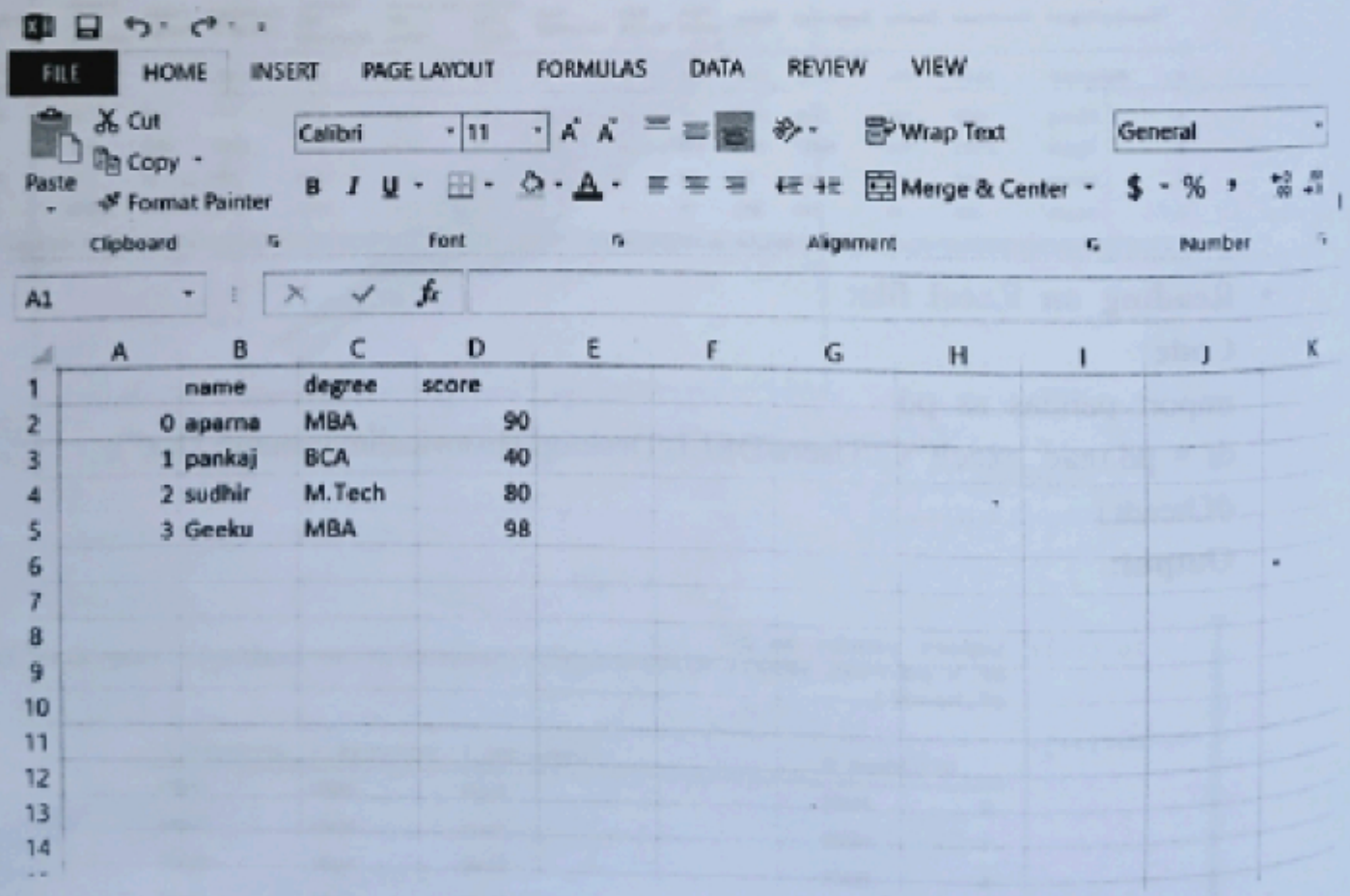
pip install openpyxl

→ **Writing a CSV file:**

Code:

```
import pandas as pd
# list of name, degree, score
nme = ["aparna", "pankaj", "sudhir", "Geeku"]
deg = ["MBA", "BCA", "M.Tech", "MBA"]
scr = [90, 40, 80, 98]
# dictionary of lists
dict = {'name': nme, 'degree': deg, 'score': scr}
df = pd.DataFrame(dict)
df.to_csv('C:/Users/DELL/Desktop/file3.csv')
```

Output:

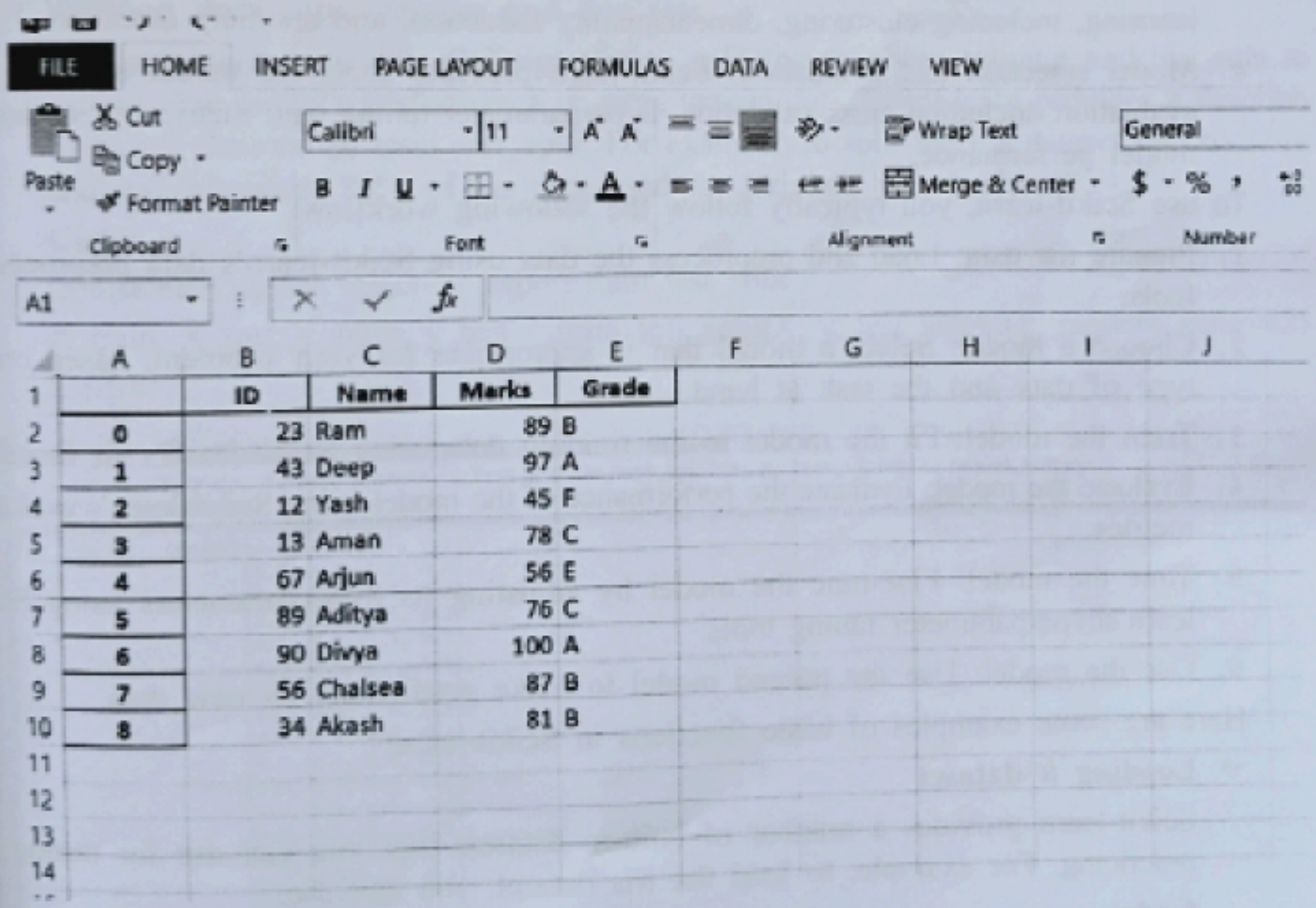


→ **Writing an Excel file:**

Code:

```
# importing packages
import pandas as pd
# dictionary of data
dct = {'ID': {0: 23, 1: 43, 2: 12,
        3: 13, 4: 67, 5: 89,
        6: 90, 7: 56, 8: 34},
  'Name': {0: 'Ram', 1: 'Deep',
        2: 'Yash', 3: 'Aman',
        4: 'Arjun', 5: 'Aditya',
        6: 'Divya', 7: 'Chalsea',
        8: 'Akash' },
  'Marks': {0: 89, 1: 97, 2: 45, 3: 78,
        4: 56, 5: 76, 6: 100, 7: 87,
  8: 81},
  'Grade': {0: 'B', 1: 'A', 2: 'F', 3: 'C',
        4: 'E', 5: 'C', 6: 'A', 7: 'B',
        8: 'B'}
  }
# forming dataframe
data = pd.DataFrame(dct)
# storing into the excel file
data.to_excel("C:/Users/DELL/Desktop/
output.xlsx")
```

Output :



	ID	Name	Marks	Grade
0	23	Ram	89	B
1	43	Deep	97	A
2	12	Yash	45	F
3	13	Aman	78	C
4	67	Arjun	56	E
5	89	Aditya	76	C
6	90	Divya	100	A
7	56	Chalsea	87	B
8	34	Akash	81	B

○○○

PRACTICAL : 3

AIM: Basic Introduction to Scikit learn

Scikit-learn, also known as sklearn, is a popular Python library used for machine learning and statistical modeling. It provides a range of tools for data preprocessing, feature extraction, supervised and unsupervised learning, and model evaluation.

Scikit-learn is built on top of other popular Python libraries such as NumPy, SciPy, and matplotlib, and provides a consistent interface for various machine learning algorithms.

Some of the key features of Scikit-learn include:

1. Data preprocessing: Scikit-learn provides a range of tools for data preprocessing, such as scaling, normalization, and feature selection.
2. Supervised learning: Scikit-learn provides a range of algorithms for supervised learning, including linear regression, logistic regression, decision trees, and support vector machines.
3. Unsupervised learning: Scikit-learn provides a range of algorithms for unsupervised learning, including clustering, dimensionality reduction, and anomaly detection.
4. Model selection and evaluation: Scikit-learn provides tools for model selection and evaluation, including cross-validation, hyperparameter tuning, and metrics for evaluating model performance.

To use Scikit-learn, you typically follow the following workflow:

1. Prepare the data: Load and preprocess the data using Scikit-learn's data preprocessing tools.
2. Choose a model: Select a model that is appropriate for your problem, based on the type of data and the task at hand.
3. Train the model: Fit the model to the training data using Scikit-learn's fit function.
4. Evaluate the model: Evaluate the performance of the model using Scikit-learn's evaluation metrics.
5. Tune the model: Fine-tune the model by adjusting its hyperparameters using Scikit-learn'shyperparameter tuning tools.
6. Use the model: Use the trained model to make predictions on new data.

Here are some examples of basic functions in Scikit-learn:

→ **Loading a dataset**

Scikit-learn provides a number of built-in datasets that you can use for testing and practicing. For example, to load the iris dataset, you can use:

Code:

```
fromsklearn.datasets import load_iris
iris = load_iris()
X, y = iris.data, iris.target
```

Output :

```
In [31]: from sklearn.datasets import load_iris
         iris = load_iris()
         X, y = iris.data, iris.target
         X

Out[31]: array([[5.1, 3.5, 1.4, 0.2],
                [4.9, 3. , 1.4, 0.2],
                [4.7, 3.2, 1.3, 0.2],
                [4.6, 3.1, 1.5, 0.2],
                [5. , 3.6, 1.4, 0.2],
                [5.4, 3.9, 1.7, 0.4],
                [4.6, 3.4, 1.4, 0.3],
                [5. , 3.4, 1.5, 0.2],
                [4.4, 2.9, 1.4, 0.2],
                [4.9, 3.1, 1.5, 0.1],
                [5.4, 3.7, 1.5, 0.2],
                [4.8, 3.4, 1.6, 0.2],
                [4.8, 3. , 1.4, 0.1],
                [4.3, 3. , 1.1, 0.1],
                [5.8, 4. , 1.2, 0.2],
                [5.7, 4.4, 1.5, 0.4],
                [5.4, 3.9, 1.3, 0.4],
                [5.1, 3.5, 1.4, 0.3],
                [5.7, 3.8, 1.7, 0.3],
```

→ **Splitting data into training and test sets**

Before training a model, it's important to split the data into training and test sets to evaluate the model's performance on new, unseen data. Scikit-learn provides a function for doing this, called train_test_split. For example, to split the iris dataset into training and test sets with 80% of the data used for training, you can use

Code:

fromsklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

Output :

```
In [34]: from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
         X_train

Out[34]: array([[4.6, 3.6, 1. , 0.2],
                [5.7, 4.4, 1.5, 0.4],
                [6.7, 3.1, 4.4, 1.4],
                [4.8, 3.4, 1.6, 0.2],
                [4.4, 3.2, 1.3, 0.2],
                [6.3, 2.5, 5. , 1.9],
                [6.4, 3.2, 4.5, 1.5],
                [5.2, 3.5, 1.5, 0.2],
                [5. , 3.6, 1.4, 0.2],
                [5.2, 4.1, 1.5, 0.1],
                [5.8, 2.7, 5.1, 1.9],
                [6. , 3.4, 4.5, 1.6],
                [6.7, 3.1, 4.7, 1.5],
                [5.4, 3.9, 1.3, 0.4],
                [5.4, 3.7, 1.5, 0.2],
                [5.5, 2.4, 3.7, 1. ],
                [6.3, 2.8, 5.1, 1.5],
                [6.4, 3.1, 5.5, 1.8],
                [6.6, 3. , 4.4, 1.4],
```

→ **Training a model**

Scikit-learn provides a wide range of machine learning models that you can use for different tasks. For example, to train a logistic regression model on the iris dataset, you can use

Code:

```
fromsklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(X_train, y_train)
```

Output:

```
In [35]: from sklearn.linear_model import LogisticRegression
         model = LogisticRegression()
         model.fit(X_train, y_train)

         C:\Users\DELL\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\linear_model\_logistic
         g: lbfgs failed to converge (status=1):
         STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

         Increase the number of iterations (max_iter) or scale the data as shown in:
             https://scikit-learn.org/stable/modules/preprocessing.html
         Please also refer to the documentation for alternative solver options:
             https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
           n_iter_i = _check_optimize_result(

Out[35]: ▾ LogisticRegression
         LogisticRegression()
```

→ **Evaluating a model**

Once you've trained a model, you can evaluate its performance on the test set using a variety of metrics. Scikit-learn provides functions for computing many of these metrics, such as accuracy, precision, recall, and F1 score. For example, to compute the accuracy of the logistic regression model on the test set, you can use:

Code:

```
fromsklearn.metrics import accuracy_score
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
```

Output:

```
In [37]: from sklearn.metrics import accuracy_score
         y_pred = model.predict(X_test)
         accuracy = accuracy_score(y_test, y_pred)
         accuracy

Out[37]: 1.0
```

→ **Cross-validation**

Cross-validation is a technique for evaluating a model's performance by splitting the data into multiple subsets and training and testing the model on different combinations of the subsets. Scikit-learn provides a function for performing k-fold cross-validation, called KFold. For example, to perform 5-fold cross-validation on the logistic regression model, you can use:

Code:

```
fromsklearn.model_selection import KFold
kf = KFold(n_splits=5)
scores = []
fortrain_index, test_index in kf.split(X):
X_train,X_test = X[train_index], X[test_index]
y_train, y_test = y[train_index], y[test_index]
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
score = accuracy_score(y_test, y_pred)
scores.append(score)
mean_score = np.mean(scores)
```

Output :

```
In [39]: from sklearn.model_selection import KFold
         kf = KFold(n_splits=5)
         scores = []
         for train_index, test_index in kf.split(X):
             X_train, X_test = X[train_index], X[test_index]
             y_train, y_test = y[train_index], y[test_index]
             model.fit(X_train, y_train)
             y_pred = model.predict(X_test)
             score = accuracy_score(y_test, y_pred)
             scores.append(score)
         mean_score = np.mean(scores)
         mean_score

         C:\Users\DELL\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\linear_model\_logi
         g: lbfgs failed to converge (status=1):
         STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

         Increase the number of iterations (max_iter) or scale the data as shown in:
             https://scikit-learn.org/stable/modules/preprocessing.html
         Please also refer to the documentation for alternative solver options:
             https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
           n_iter_i = _check_optimize_result(
         C:\Users\DELL\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\linear_model\_logi
         g: lbfgs failed to converge (status=1):
         STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

         Increase the number of iterations (max_iter) or scale the data as shown in:
             https://scikit-learn.org/stable/modules/preprocessing.html
         Please also refer to the documentation for alternative solver options:
             https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
           n_iter_i = _check_optimize_result(

Out[39]: 0.9266666666666665
```

○○○

PRACTICAL : 4

AIM: Implement the Find-S concept learning algorithm that finds the most specific hypothesis that is consistent with the given training data.

Conditions:Hypothesis can only be conjunction (AND) of literals. Literals are either

attributes or their negations.

Steps Involved In Find-S :

1. start with the most specific hypothesis.
   h = {Ø, Ø, Ø, Ø, Ø, Ø}
2. Take the next example and if it is negative, then no changes occur to the hypothesis.
3. If the example is positive and we find that our initial hypothesis is too specific then we update our current hypothesis to a general condition.
4. Keep repeating the above steps till all the training examples are complete.
5. After we have completed all the training examples we will have the final hypothesis when can use to classify the new examples.

Algorithm :

1. Initialize h to the most specific hypothesis in H
2. For each positive training instance x
   For each attribute constraint a, in h
   If the constraint a, is satisfied by x
   Then do nothing
   Else replace a, in h by the next more general constraint that is satisfied by x
3. Output hypothesis h

Code:

```
import pandas as pd
importnumpy as np
data = pd.read_csv('C:/Users/DELL/Desktop/data.csv')
    concepts = np.array(data)[:,:-1]
    target = np.array(data)[:,-1]
    def train(con, tar):
    fori, val in enumerate(tar):
    ifval == 'yes':
    specific_h = con[i].copy()
    break

    fori, val in enumerate(con):
    if tar[i] == 'yes':
    for x in range(len(specific_h)):
```

ifval[x] != specific_h[x]:
specific_h[x] = '?'
else:
pass
returnspecific_h
print(train(concepts, target))

Output

```
In [55]: import pandas as pd
         import numpy as np
         data = pd.read_csv('C:/Users/DELL/Desktop/data.csv')
         concepts = np.array(data)[:,:-1]
         target = np.array(data)[:,-1]
         def train(con, tar):
             for i, val in enumerate(tar):
                 if val == 'yes':
                     specific_h = con[i].copy()
                     break

             for i, val in enumerate(con):
                 if tar[i] == 'yes':
                     for x in range(len(specific_h)):
                         if val[x] != specific_h[x]:
                             specific_h[x] = '?'
                         else:
                             pass
             return specific_h
         print(train(concepts, target))

         ['sunny' 'warm' '?' 'strong' '?' '?']
```

The find_s function takes in a dataset data as input, where each row represents an instance and the last column represents the class label (0 or 1). The function initializes the most specific hypothesis to a string of '0's with the same length as the number of attributes in the dataset.

The function then loops through each instance in the dataset. If the instance is positive (i.e., the class label is 1), the function loops through each attribute in the instance. If the attribute is not already part of the hypothesis and the current value of the attribute in the hypothesis is '0', the function adds the attribute to the hypothesis.

Finally, the function returns the final hypothesis.

○○○

PRACTICAL : 5

AIM: Import Pima Indian diabetes data. Apply select K best and chi2 for feature selection. Identify the best features

Code:

```
import pandas as pd
importnumpy as np
fromsklearn.feature_selection import SelectKBest, chi2

# Load the dataset
url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
dataset = pd.read_csv(url, names=names)

# Split dataset into features (X) and target variable (y)
X = dataset.iloc[:, :-1]
y = dataset.iloc[:, -1]

# Apply feature selection using chi2 and SelectKBest
kbest = SelectKBest(score_func=chi2, k=4)
X_new = kbest.fit_transform(X, y)

# Print the names of the best features
mask = kbest.get_support()
best_features = X.columns[mask]
print(best_features)
```

Output:

```
In [20]: import pandas as pd
         import numpy as np
         from sklearn.feature_selection import SelectKBest, chi2

         # Load the dataset
         url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"
         names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
         dataset = pd.read_csv(url, names=names)

         # Split dataset into features (X) and target variable (y)
         X = dataset.iloc[:, :-1]
         y = dataset.iloc[:, -1]

         # Apply feature selection using chi2 and SelectKBest
         kbest = SelectKBest(score_func=chi2, k=4)
         X_new = kbest.fit_transform(X, y)

         # Print the names of the best features
         mask = kbest.get_support()
         best_features = X.columns[mask]
         print(best_features)

         Index(['plas', 'test', 'mass', 'age'], dtype='object')
```

○○○

PRACTICAL : 6

AIM: Write a program to learn a decision tree and use it to predict class labels of test data

Training and test data will be explicitly provided by instructor. Tree pruning should not be performed.

Code:

```
fromsklearn.tree import DecisionTreeClassifier
fromsklearn.metrics import accuracy_score

# Load the training and test data
X_train = [[1, 2], [2, 3], [3, 1], [4, 2], [3, 3], [2, 1], [1, 3], [4, 1]]
y_train = [0, 0, 1, 1, 0, 1, 1, 0]
X_test = [[1, 1], [2, 2], [3, 4], [4, 3]]
y_test = [1, 0, 1, 0]

# Create a decision tree classifier and fit it to the training data
clf = DecisionTreeClassifier(random_state=42)
clf.fit(X_train, y_train)

# Make predictions on the test data
y_pred = clf.predict(X_test)

# Calculate the accuracy of the classifier
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

Output:

```
In [31]: from sklearn.tree import DecisionTreeClassifier
         from sklearn.metrics import accuracy_score

         # Load the training and test data
         X_train = [[1, 2], [2, 3], [3, 1], [4, 2], [3, 3], [2, 1], [1, 3], [4, 1]]
         y_train = [0, 0, 1, 1, 0, 1, 1, 0]
         X_test = [[1, 1], [2, 2], [3, 4], [4, 3]]
         y_test = [1, 0, 1, 0]

         # Create a decision tree classifier and fit it to the training data
         clf = DecisionTreeClassifier(random_state=42)
         clf.fit(X_train, y_train)

         # Make predictions on the test data
         y_pred = clf.predict(X_test)

         # Calculate the accuracy of the classifier
         accuracy = accuracy_score(y_test, y_pred)
         print("Accuracy:", accuracy)

         Accuracy: 0.5
```

In this example, we are using a small synthetic dataset for demonstration purposes. You can replace X_train, y_train, X_test, and y_test with your own training and test data.

The DecisionTreeClassifier class is used to create a decision tree classifier. We set random_state to 42 for reproducibility. The fit() method is used to train the classifier on the training data. We then use the predict() method to make predictions on the test data, and calculate the accuracy of the classifier using the accuracy_score() function from scikit-learn's metrics module.

Note that we are not performing any tree pruning in this example. If you want to perform tree pruning, you can set the max_depth parameter or other pruning parameters in the DecisionTreeClassifier class.

❍❍❍

PRACTICAL : 7

AIM: ML Project. Use the following dataset as music.csv

a. **Store file as music.csv and import it to python using pandas**

b. **Prepare the data by splitting data in input(age ,gender) and output(genre) data set**

c. **Use decision tree model from sklearn to predict the genre of various age group people.(Ex A male of age 21 likes hiphop whereas female of age 22 likes dance)**

d. **Calculate the accuracy of the model. vary training and test size to check different accuracy values model achieves.**

Code:

```
#Storing data into CSV
fromsklearn.model_selection import train_test_split
import pandas as pd
fromsklearn.tree import DecisionTreeClassifier
fromsklearn.metrics import accuracy_score
# list of name, degree, score
age = [20,23,25,26,29,30,31,33,37,20,21,25,26,27,30,31,34,34]
gender = [1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0]
genre=["HipHop","HipHop","HipHop","Jazz","Jazz","Jazz","Classical","Classical","Classical","
Dance","Dance","Dance","Acoustic","Acoustic","Acoustic","Classical","Classical","Classical"]
# dictionary of lists
dict = {'Age': age, 'gender': gender, 'genre': genre}
df = pd.DataFrame(dict)
df.to_csv('C:/Users/DELL/Desktop/music.csv')

# Load the music data
data = pd.read_csv('C:/Users/DELL/Desktop/music.csv')

# Split the data into input and output sets
X = data.iloc[:, :-1].values
y = data.iloc[:, -1].values

# Convert gender to binary values
X[:, 1] = pd.factorize(X[:, 1])[0]

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

```
# Train a decision tree classifier
clf = DecisionTreeClassifier(random_state=42)
clf.fit(X_train, y_train)

# Make predictions on the test data
y_pred = clf.predict(X_test)

# Calculate the accuracy of the classifier
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

Output:

```
In [4]: #Storing data into CSV
        from sklearn.model_selection import train_test_split
        import pandas as pd
        from sklearn.tree import DecisionTreeClassifier
        from sklearn.metrics import accuracy_score
        # list of name, degree, score
        age = [20,23,25,26,29,30,31,33,37,20,21,25,26,27,30,31,34,34]
        gender = [1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0]
        genres=["HipHop","HipHop","HipHop","Jazz","Jazz","Jazz","Classical","Classical","Classical","Dance","Dance","Dance","Acoustic","A
        # dictionary of lists
        dict = {'Age': age, 'gender': gender, 'genre': genre}
        df = pd.DataFrame(dict)
        df.to_csv('C:/Users/DELL/Desktop/music.csv')

        # Load the music data
        data = pd.read_csv('C:/Users/DELL/Desktop/music.csv')

        # Split the data into input and output sets
        X = data.iloc[:, :-1].values
        y = data.iloc[:, -1].values

        # Convert gender to binary values
        X[:, 1] = pd.factorize(X[:, 1])[0]

        # Split the data into training and test sets
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

        # Train a decision tree classifier
        clf = DecisionTreeClassifier(random_state=42)
        clf.fit(X_train, y_train)

        # Make predictions on the test data
        y_pred = clf.predict(X_test)

        # Calculate the accuracy of the classifier
        accuracy = accuracy_score(y_test, y_pred)
        print("Accuracy:", accuracy)

        Accuracy: 0.6666666666666666
```

In this example, we first load the music data from the 'music.csv' file using pandas and split it into input (age and gender) and output (genre) sets. We convert the gender values to binary using the factorize() function from pandas. We then split the data into training and test sets using the train_test_split() function from scikit-learn.

Next, we train a decision tree classifier on the training data and make predictions on the test data using the predict() method. We calculate the accuracy of the classifier using the accuracy_score() function from scikit-learn's metrics module.

Finally, we can vary the test_size parameter in train_test_split() to check the different accuracy values that the model achieves. For example, we can set test_size=0.2 to use 20% of the data for testing, or test_size=0.5 to use 50% of the data for testing.

○○○

PRACTICAL : 8

AIM: Write a program to use a K-nearest neighbor it to predict class labels of test data. Training and test data must be provided explicitly.

Code :

```
fromsklearn.neighbors import KNeighborsClassifier
fromsklearn.metrics import accuracy_score

# Load the training and test data
X_train = [[1, 2], [2, 3], [3, 1], [4, 2], [3, 3], [2, 1], [1, 3], [4, 1]]
y_train = [0, 0, 1, 1, 0, 1, 1, 0]
X_test = [[1, 1], [2, 2], [3, 4], [4, 3]]
y_test = [1, 0, 1, 0]

# Create a K-nearest neighbor classifier and fit it to the training data
clf = KNeighborsClassifier(n_neighbors=3)
clf.fit(X_train, y_train)

# Make predictions on the test data
y_pred = clf.predict(X_test)

# Calculate the accuracy of the classifier
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

Output:

```
In [5]: from sklearn.neighbors import KNeighborsClassifier
        from sklearn.metrics import accuracy_score

        # Load the training and test data
        X_train = [[1, 2], [2, 3], [3, 1], [4, 2], [3, 3], [2, 1], [1, 3], [4, 1]]
        y_train = [0, 0, 1, 1, 0, 1, 1, 0]
        X_test = [[1, 1], [2, 2], [3, 4], [4, 3]]
        y_test = [1, 0, 1, 0]

        # Create a K-nearest neighbor classifier and fit it to the training data
        clf = KNeighborsClassifier(n_neighbors=3)
        clf.fit(X_train, y_train)

        # Make predictions on the test data
        y_pred = clf.predict(X_test)

        # Calculate the accuracy of the classifier
        accuracy = accuracy_score(y_test, y_pred)
        print("Accuracy:", accuracy)

        Accuracy: 0.75
```

In this example, we are using a small synthetic dataset for demonstration purposes. You can replace X_train, y_train, X_test, and y_test with your own training and test data. The KNeighborsClassifier class is used to create a K-nearest neighbor classifier. We set the n_neighbors parameter to 3, which specifies the number of neighbors to use in the classification. The fit() method is used to train the classifier on the training data. We then use the predict() method to make predictions on the test data, and calculate the accuracy of the classifier using the accuracy_score() function from scikit-learn's metrics module.

Note that the K-nearest neighbor algorithm is a lazy learner, which means it does not actually learn a model from the training data, but instead memorizes the training data and uses it to make predictions on new data. This can make the algorithm slow when dealing with large datasets, but it can also be more flexible and adaptive than other algorithms. ○○○

PRACTICAL : 9

AIM: Import vgsales.csv from kaggle platform.

a. Find rows and columns in dataset

b. Find basic information regarding dataset using describe command.

c. Find values using values command.

Code:

```
import pandas as pd
df = pd.read_csv("C:/Users/HP/Downloads/vgsales.csv")
df
```

Output:

```
import pandas as pd
df = pd.read_csv("C:/Users/HP/Downloads/vgsales.csv")
df
```

✓ 0.1s

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	259	Asteroids	2600	1980	Shooter	Atari	4.00	0.26	0.00	0.05	4.31
1	545	Missile Command	2600	1980	Shooter	Atari	2.56	0.17	0.00	0.03	2.76
2	1768	Kaboom!	2600	1980	Misc	Activision	1.07	0.07	0.00	0.01	1.15
3	1971	Defender	2600	1980	Misc	Atari	0.99	0.05	0.00	0.01	1.05
4	2575	Boxing	2600	1980	Fighting	Activision	0.72	0.04	0.00	0.01	0.77
...	...	...	...	...	...	...	...	...	...	...	...
16319	16565	Mighty No. 9	3DS	2016	Platform	Deep Silver	0.01	0.00	0.00	0.00	0.01
16320	16572	Resident Evil 4 HD	3DS	2016	Shooter	Capcom	0.01	0.00	0.00	0.00	0.01
16321	16573	Farming 2017 - The Simulation	PS4	2016	Simulation	UIG Entertainment	0.00	0.01	0.00	0.00	0.01
16322	16579	Rugby Challenge 3	3DS	2016	Sports	Alternative Software	0.00	0.01	0.00	0.00	0.01
16323	16592	Chou Sizeu na Akai Hana: Kiji wa Tsuki ni Shim...	PSV	2016	Action	dramatic create	0.00	0.00	0.01	0.00	0.01

16324 rows × 11 columns

a. Find rows and columns in dataset

Code:

```
print("Number of rows and columns in the dataset:", df.shape)
```

Output:

```
# Find the number of rows and columns in the dataset
print("Number of rows and columns in the dataset:", df.shape)
```

✓ 0.0s

```
Number of rows and columns in the dataset: (16324, 11)
```

b. Find basic information regarding dataset using describe command.

Code: print("Basic information about the dataset:\n", df.describe())

Output:

```
print("Basic information about the dataset:\n", df.describe())
✓ 0.0s

Basic information about the dataset:
                Rank          Year     NA_Sales     EU_Sales     JP_Sales  \
count  16324.000000  16324.000000  16324.000000  16324.000000  16324.000000
mean    8291.508270   2006.404251      0.265464      0.147581      0.078673
std     4792.043734      5.826744      0.821659      0.508889      0.311584
min        1.000000   1980.000000      0.000000      0.000000      0.000000
25%     4135.750000   2003.000000      0.000000      0.000000      0.000000
50%     8293.500000   2007.000000      0.080000      0.020000      0.000000
75%    12439.250000   2010.000000      0.240000      0.110000      0.040000
max    16600.000000   2016.000000     41.490000     29.020000     10.220000

       Other_Sales  Global_Sales
count  16324.000000  16324.000000
mean       0.048334      0.540328
std        0.189902      1.565860
min        0.000000      0.010000
25%        0.000000      0.060000
50%        0.010000      0.170000
75%        0.040000      0.480000
max       10.570000     82.740000
```

c. Find values using values command.

Code:

print("Values in the dataset:\n", df.values)

Output:

```
print("Values in the dataset:\n", df.values)
✓ 0.0s

Values in the dataset:
 [[259 'Asteroids' '2600' ... 0.0 0.06 4.31]
 [545 'Missile Command' '2600' ... 0.0 0.03 2.76]
 [1768 'Kaboom!' '2600' ... 0.0 0.01 1.15]
 ...
 [16573 'Farming 2017 - The Simulation' 'PS4' ... 0.0 0.0 0.01]
 [16579 'Rugby Challenge 3' 'XOne' ... 0.0 0.0 0.01]
 [16592 'Chou Ezaru wa Akai Hana: Koi wa Tsuki ni Shiryobe Kareru' 'PSV'
  ... 0.01 0.0 0.01]]
```

○○○

PRACTICAL : 10

AIM: Project on regression

a. Import home_data.csv on kaggle using pandas

b. Understand data by running head ,info and describe command

c. Plot the price of house with respect to area using matplotlib library

d. Apply linear regression model to predict the price of house

Solution:

a. Import home_data.csv on Kaggle using pandas:

To import the "home_data.csv" file, you first need to download it from Kaggle to your local machine. Then, you can use the following code to import it into a pandasdataframe:

Python

Code:

```
import pandas as pd
df = pd.read_csv("C:/Users/DELL/Desktop/home_data.csv")
df
```

Output:

```
In [3]: import pandas as pd
        df = pd.read_csv("C:/Users/DELL/Desktop/home_data.csv")
        df

Out[3]:
```

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	...	grade	sqft_above	sqft_basement	yr_bu
0	7129300520	20141013T000000	221900	3	1.00	1180	5650	1.0	0	0	...	7	1180	0	19
1	6414100192	20141209T000000	538000	3	2.25	2570	7242	2.0	0	0	...	7	2170	400	19
2	5631500400	20150225T000000	180000	2	1.00	770	10000	1.0	0	0	...	6	770	0	19
3	2487200875	20141209T000000	604000	4	3.00	1960	5000	1.0	0	0	...	7	1050	910	19
4	1954400510	20150218T000000	510000	3	2.00	1680	8080	1.0	0	0	...	8	1680	0	19
...															
21608	263000018	20140521T000000	360000	3	2.50	1530	1131	3.0	0	0	...	8	1530	0	20
21609	6600060120	20150223T000000	400000	4	2.50	2310	5813	2.0	0	0	...	8	2310	0	20
21610	1523300141	20140623T000000	402101	2	0.75	1020	1350	2.0	0	0	...	7	1020	0	20
21611	291310100	20150116T000000	400000	3	2.50	1600	2388	2.0	0	0	...	8	1600	0	20
21612	1523300157	20141015T000000	325000	2	0.75	1020	1076	2.0	0	0	...	7	1020	0	20

21613 rows × 21 columns

b. Understand data by running head, info, and describe command:

You can use the following commands to get a better understanding of the data:

Code:

```
# View the first 5 rows of the dataframe
df.head()
 # Get information about the dataframe
df.info()
```

Get descriptive statistics of the dataframe

df.describe()

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21613 entries, 0 to 21612
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   id             21613 non-null  int64
 1   date           21613 non-null  object
 2   price          21613 non-null  int64
 3   bedrooms       21613 non-null  int64
 4   bathrooms      21613 non-null  float64
 5   sqft_living    21613 non-null  int64
 6   sqft_lot       21613 non-null  int64
 7   floors         21613 non-null  float64
 8   waterfront     21613 non-null  int64
 9   view           21613 non-null  int64
 10  condition      21613 non-null  int64
 11  grade          21613 non-null  int64
 12  sqft_above     21613 non-null  int64
 13  sqft_basement  21613 non-null  int64
 14  yr_built       21613 non-null  int64
 15  yr_renovated   21613 non-null  int64
 16  zipcode        21613 non-null  int64
 17  lat            21613 non-null  float64
 18  long           21613 non-null  float64
 19  sqft_living15  21613 non-null  int64
 20  sqft_lot15     21613 non-null  int64
dtypes: float64(4), int64(16), object(1)
memory usage: 3.5+ MB
```

Out[2]:

	id	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	g
count	2.161300e+04	2.161300e+04	21613.000000	21613.000000	21613.000000	2.161300e+04	21613.000000	21613.000000	21613.000000	21613.000000	21613.00
mean	4.580302e+09	5.400881e+05	3.370842	2.114757	2079.899736	1.510697e+04	1.494309	0.007542	0.234303	3.409430	7.66
std	2.876566e+09	3.671272e+05	0.930062	0.770163	918.440897	4.142051e+04	0.539989	0.086517	0.766318	0.650743	1.17
min	1.000102e+06	7.500000e+04	0.000000	0.000000	290.000000	5.200000e+02	1.000000	0.000000	0.000000	1.000000	1.00
25%	2.123049e+09	3.219500e+05	3.000000	1.750000	1427.000000	5.040000e+03	1.000000	0.000000	0.000000	3.000000	7.00
50%	3.904930e+09	4.500000e+05	3.000000	2.250000	1910.000000	7.618000e+03	1.500000	0.000000	0.000000	3.000000	7.00
75%	7.308900e+09	6.450000e+05	4.000000	2.500000	2550.000000	1.068800e+04	2.000000	0.000000	0.000000	4.000000	8.00
max	9.900000e+09	7.700000e+06	33.000000	8.000000	13540.000000	1.651359e+06	3.500000	1.000000	4.000000	5.000000	13.00

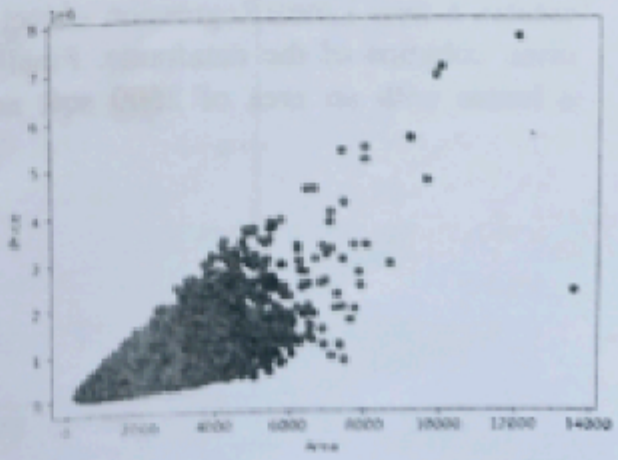
The head() method displays the first 5 rows of the dataframe, while info() provides information about the dataframe's columns and data types. The describe() method gives you descriptive statistics of the numeric columns in the dataframe.

c. Plot the price of house with respect to area using matplotlib library

Code:

```
importmatplotlib.pyplot as plt
plt.scatter(df['sqft_living'], df['price'])
plt.xlabel('Area')
plt.ylabel('Price')
plt.show()
```

Output:



This code creates a scatter plot of the sqft_living column (representing the area of the house) on the x-axis and the price column on the y-axis. The xlabel() and ylabel() functions are used to label the x and y-axes, respectively.

d. Apply linear regression model to predict the price of house

Code:

```
fromsklearn.linear_model import LinearRegression
# Create a linear regression object
model = LinearRegression()
# Train the model using the sqft_living and price columns
model.fit(df[['sqft_living']], df['price'])
 # Predict the price of a house with an area of 2500 sqft
predicted_price = model.predict([[2500]])
print(predicted_price)
```

Output:

```
In [13]: import pandas as pd
         df = pd.read_csv("C:/Users/DELL/Desktop/home_data.csv")
         # View the first 5 rows of the dataframe
         from sklearn.linear_model import LinearRegression
         # Create a linear regression object
         model = LinearRegression()
         # Train the model using the sqft_living and price columns
         model.fit(df[['sqft_living']], df['price'])
          # Predict the price of a house with an area of 2500 sqft
         predicted_price = model.predict([[2500]])
         print(predicted_price)

         [657978.17635389]
```

This code imports the LinearRegression class from the sklearn.linear_modelmodule and creates a new LinearRegression object. It then trains the model using the sqft_living and price columns of the dataframe. Finally, it uses the trained model to predict the price of a house with an area of 2500 sqft and prints the predicted price to the console.

○○○

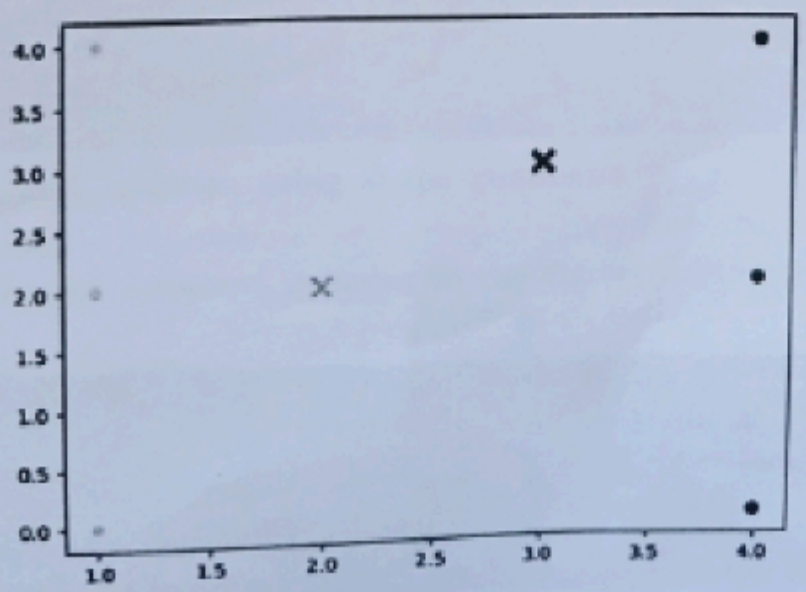
PRACTICAL : 11

AIM: Write a program to cluster a set of points using K-means. Training and test data must be provided explicitly.

Code:

```
importnumpy as np
importmatplotlib.pyplot as plt
fromsklearn.cluster import KMeans

# Generate some sample data
X_train = np.array([[1, 2], [1, 4], [1, 0], [4, 2], [4, 4], [4, 0]])
X_test = np.array([[2, 2], [3, 3]])

# Create a KMeans object with 2 clusters
kmeans = KMeans(n_clusters=2)

# Fit the KMeans model to the training data
kmeans.fit(X_train)

# Predict the cluster labels for the test data
y_pred = kmeans.predict(X_test)

# Visualize the clusters and the test data
plt.scatter(X_train[:, 0], X_train[:, 1], c=kmeans.labels_)
plt.scatter(X_test[:, 0], X_test[:, 1], marker='x', s=100, linewidths=3, c=y_pred)
plt.show()
```

Output:



In this program, we first generate some sample training and test data. The X_train array contains 6 points with 2 features each, and the X_test array contains 2 points with the same 2 features.

We then create a KMeans object with 2 clusters and fit it to the training data using the fit() method. This trains the KMeans model on the training data and determines the centroids of the clusters.

Next, we predict the cluster labels for the test data using the predict() method. This assigns each test point to one of the two clusters.

Finally, we visualize the clusters and the test data using a scatter plot. The training data points are colored according to their assigned cluster label, while the test data points are marked with an 'x' and colored according to their predicted cluster label. The resulting plot shows the two clusters and how the test data points are assigned to them.

○○○

PRACTICAL : 12

AIM: Import Iris dataset

a. Find rows and columns using shape command
b. Print first 30 instances using head command
c. Find out the data instances in each class.(use groupby and size)
d. Plot the univariategraphs(box plot and histograms)
e. Plot the multivariate plot(scatter matrix)
f. Split data to train model by 80% data values
g. Apply K-NN and k means clustering to check accuracy and decide which is better.

Dataset link: https://www.kaggle.com/datasets/jillanisofttech/iris-dataset-uci

Code:

```
import pandas as pd
df = pd.read_csv("C:/Users/HP/Downloads/iris.csv")
df
```

Output:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

a. **Find rows and columns using shape command**

Code:

```
print("Number of rows and columns in the dataset:", df.shape)
```

Output:

```
# Find the number of rows and columns in the dataset
print("Number of rows and columns in the dataset:", df.shape)
✓ 0.7s

Number of rows and columns in the dataset: (150, 6)
```

b. Print first 30 instances using head command

Code:

```
print("First 30 instances in the dataset:\n", df.head(30))
```

Output:

```
First 30 instances in the dataset:
    Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
0    1            5.1           3.5            1.4           0.2  Iris-setosa
1    2            4.9           3.0            1.4           0.2  Iris-setosa
2    3            4.7           3.2            1.3           0.2  Iris-setosa
3    4            4.6           3.1            1.5           0.2  Iris-setosa
4    5            5.0           3.6            1.4           0.2  Iris-setosa
5    6            5.4           3.9            1.7           0.4  Iris-setosa
6    7            4.6           3.4            1.4           0.3  Iris-setosa
7    8            5.0           3.4            1.5           0.2  Iris-setosa
8    9            4.4           2.9            1.4           0.2  Iris-setosa
9   10            4.9           3.1            1.5           0.1  Iris-setosa
10  11            5.4           3.7            1.5           0.2  Iris-setosa
11  12            4.8           3.4            1.6           0.2  Iris-setosa
12  13            4.8           3.0            1.4           0.1  Iris-setosa
13  14            4.3           3.0            1.1           0.1  Iris-setosa
14  15            5.8           4.0            1.2           0.2  Iris-setosa
15  16            5.7           4.4            1.5           0.4  Iris-setosa
16  17            5.4           3.9            1.3           0.4  Iris-setosa
17  18            5.1           3.5            1.4           0.3  Iris-setosa
18  19            5.7           3.8            1.7           0.3  Iris-setosa
19  20            5.1           3.8            1.5           0.3  Iris-setosa
20  21            5.4           3.4            1.7           0.2  Iris-setosa
21  22            5.1           3.7            1.5           0.4  Iris-setosa
22  23            4.6           3.6            1.0           0.2  Iris-setosa
23  24            5.1           3.3            1.7           0.5  Iris-setosa
24  25            4.8           3.4            1.9           0.2  Iris-setosa
25  26            5.0           3.0            1.6           0.2  Iris-setosa
26  27            5.0           3.4            1.6           0.4  Iris-setosa
27  28            5.2           3.5            1.5           0.2  Iris-setosa
28  29            5.2           3.4            1.4           0.2  Iris-setosa
29  30            4.7           3.2            1.6           0.2  Iris-setosa
```

c. Find out the data instances in each class.(use groupby and size)

Code:

```
class_counts = df.groupby('Species').size()
print("Data instances in each class:\n", class_counts)
```

Output:

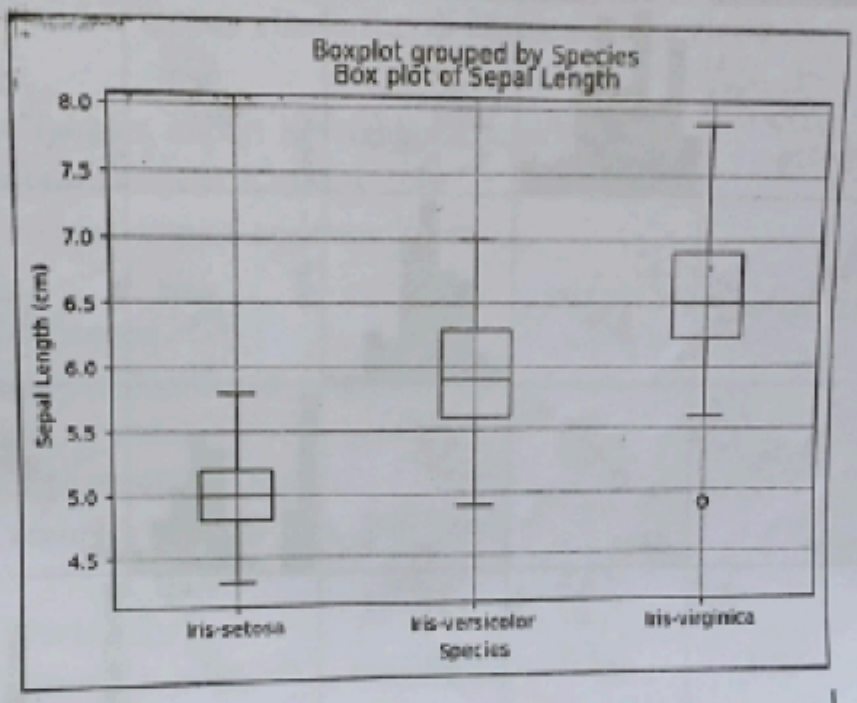
```
Data instances in each class:
 Species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
dtype: int64
```

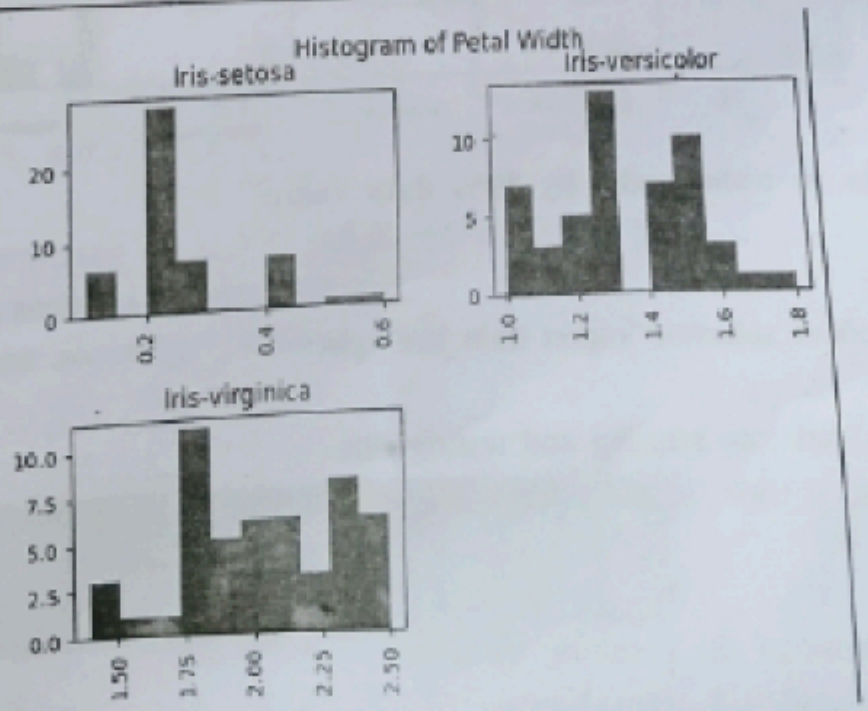
d. Plot the univariategraphs(box plot and histograms)

Code:

```
importmatplotlib.pyplot as plt

# Box plot of sepal length
df.boxplot(column='SepalLengthCm',
by='Species')
plt.title("Box plot of Sepal Length")
plt.xlabel("Species")
plt.ylabel("Sepal Length (cm)")
plt.show()

# Histogram of petal width
df.hist(column='PetalWidthCm',
by='Species')
plt.suptitle("Histogram of Petal Width")
plt.xlabel("Petal Width (cm)")
plt.ylabel("Count")
plt.show()
```

Output:



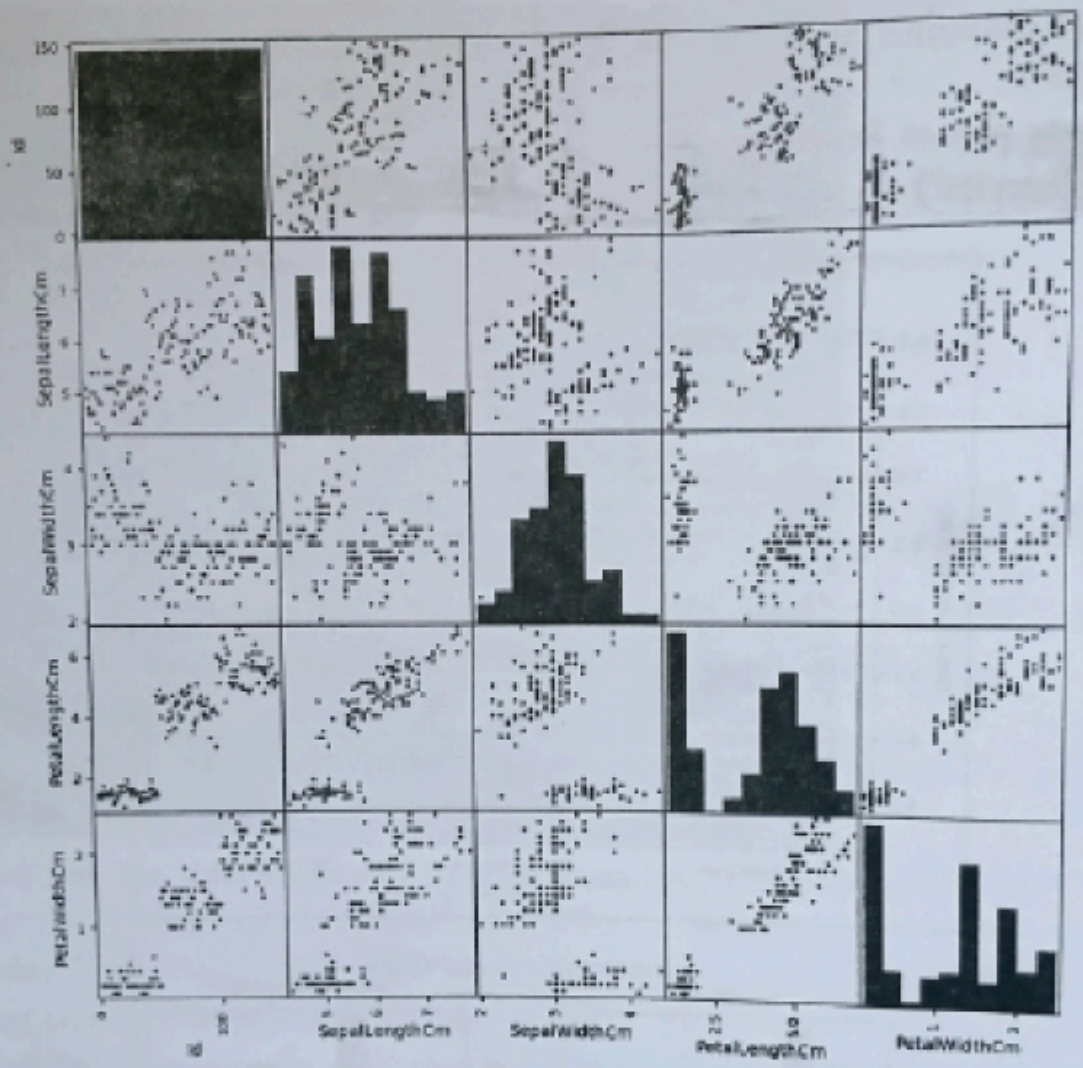


e. Plot the multivariate plot(scatter matrix)

Code:

```
frompandas.plotting import scatter_matrix
scatter_matrix(df, alpha=0.8, figsize=(10,10), diagonal='hist')
plt.show()
```

Output:



f. Split data to train model by 80% data values

Code:

```
fromsklearn.model_selection import train_test_split

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(df.iloc[:, :-1], df.iloc[:, -1], test_size=0.2,
random_state=42)

# Print the shapes of the resulting datasets
print("X_train shape:", X_train.shape)
```

```
print("X_test shape:", X_test.shape)
print("y_train shape:", y_train.shape)
print("y_test shape:", y_test.shape)
```

Output:

```
X_train shape: (120, 5)
X_test shape: (30, 5)
y_train shape: (120,)
y_test shape: (30,)
```

h. Apply K-NN and k means clustering to check accuracy and decide which is better.

Code:

```
fromsklearn.neighbors import KNeighborsClassifier
fromsklearn.cluster import KMeans
fromsklearn.metrics import accuracy_score

# K-NN classification
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)
knn_pred = knn.predict(X_test)
knn_acc = accuracy_score(y_test, knn_pred)

# K-means clustering
kmeans = KMeans(n_clusters=3, random_state=42)
kmeans.fit(X_train)
kmeans_pred = kmeans.predict(X_test)
kmeans_acc = accuracy_score(y_test, kmeans_pred)

# Compare accuracy of K-NN and K-means clustering
print("K-NN accuracy:", knn_acc)
print("K-means accuracy:", kmeans_acc)
```

Output:

```
K-NN accuracy: 1.0
K-means accuracy: 0.0
```

○○○

MODEL QUESTION PAPER-1 FOR MID SEMESTER EXAM

(UNIT 1, 2, 3, 6)

Marks : 30

Q.1 a. Give the difference between supervised learning and unsupervised learning. 3

b. Explain well-posed learning problem with suitable example. 7

Q.2 a. Explain types of data with example. 3

b. Write a short note on data quality and remediation. 7

OR

Q.2 a. Explain structure of box-plot. 3

b. Write a short note on feature subset selection method. 7

Q.3 a. Describe k-fold cross validation method with example. 7

b. Compare and contrast NumPy with Pandas. 3

OR

Q.3 a. Consider the following confusion matrix of the win/loss prediction of cricket match. Calculate the accuracy, error rate, sensitivity, specificity, precision, recall and F-measure of the model. 7

	Actual Win	Actual Loss
Predicted Win	80	6
Predicted Loss	5	9

b. State features of Matplotlib. 3

■

MODEL QUESTION PAPER-2 FOR MID SEMESTER EXAM

(UNIT 1, 2, 3, 6)

Marks : 30

Q.1 a. Give the difference between qualitative data and quantitative data. 3

b. Write a short note on dimensionality reduction. 7

Q.2 a. Describe application of Machine Learning in the field of healthcare. 3

b. Explain different tools and technology used in Machine Learning. 7

OR

Q.2 a. Identify main three features for following well-posed problem 3

Spam Detection Sytem

b. Explain types of machine learning in detail. 7

Q.3 a. Explain Holdout method in detail. 7

b. How to load dataset using Numpy? Explain. 3

OR

Q.3 a. Explain overfitting and underfitting with suitable example. 7

b. How to plot a vertical line and a horizontal line using Matplotlib? 3

■

MODEL QUESTION PAPER-3 FOR MID SEMESTER EXAM

(UNIT 1, 2, 3, 6)

Marks : 30

Q.1 a. Give the applications of Scikit-Learn. 3

b. Describe basic concept of Machine Learning and its application. 7

Q.2 a. State various strategies to handle missing values. 3

b. Describe measures of central tendency with appropriate examples. 7

OR

Q.2 a. What is IQR? How it is meseaured? 3

b. Describe Machine Learning Activities in detail. 7

Q.3 a. Describe model parameter tuning in detail. 7

b. Give the difference between predictive model and descriptive model. 3

OR

Q.3 a. Consider the following confusion matrix of the win/loss prediction of cricket match. Calculate the accuracy, error rate, sensitivity, specificity, precision, recall and F-measure of the model. 7

	Actual Win	Actual Loss
Predicted Win	70	10
Predicted Loss	15	5

b. Give the difference between Bagging and Boosting. 3

■

MODEL QUESTION PAPER 1

DE – SEMESTER IV (NEW SYLLABUS) EXAMINATION

Subject: Fundamentals of Machine Learning (4341603)

Time: 2 hours 30 min **Total Marks: 70**

Instructions:

1. Attempt all questions.
2. Make suitable assumptions wherever necessary.
3. Figures to the right indicate full marks.

			Marks	CO (Course Outcome)	Cognitive Level (As per Revised Bloom's Taxonomy)
Q.1	(a)	State the features of NumPy.	03	CO 5	R
	(b)	Define Human Learning. Give the types of human learning.	04	CO 1	R
	(c)	Describe different types of Machine Learning activities.	07	CO 2	U
		OR			
	(c)	Describe types of data in Machine Learning with suitable examples.	07	CO 2	U
Q.2	(a)	Give the difference between predictive model and descriptive model.	03	CO 2	U
	(b)	Explain the concept of penalty and reward in reinforcement learning.	04	CO 1	U
	(c)	While predicting malignancy of tumour of a set of patients using a classification model, following are the data recorded: 1. Correct predictions – 15 malignant, 75 benign 2. Incorrect predictions – 4 malignant, 6 benign Create confusion matrix. Calculate the sensitivity, specificity, precision, Recall and F-measure of the model.	07	CO 3	A
		OR			
Q.2	(a)	Explain model underfitting in brief.	03	CO 2	U
	(b)	Explain application of ML in healthcare sector.	04	CO 1	U
	(c)	Can the performance of a learning model be improved? If yes, explain how.	07	CO 2	A
Q. 3	(a)	Out of 200 emails, a classification model correctly predicted 150 spam emails and 30 ham emails. What is the error rate of the model?	03	CO 2	A
	(b)	Write a short note on PCA.	04	CO 2	U
	(c)	Discuss the kNN model in detail.	07	CO 3	U
		OR			

Abbreviation

AI	:	Artificial Intelligence
ANN	:	Artificial Neural Networks
F1	:	F1 Score
GPU	:	Graphics Processing Unit
kNN	:	k-Nearest Neighbour
LDA	:	Linear Discriminant Analysis
LOOCV	:	Leave-One-Out Cross Validation
ML	:	Machine Learning
NLP	:	Natural Language Processing
NN	:	Neural Network
PCA	:	Principal Component Analysis
RF	:	Random Forest
RL	:	Reinforcement Learning
SAS	:	Statistical Analysis System
SVD	:	Single Value Decomposition
SVM	:	Support Vector Machine

ooo

Q. 3	(a)	Out of 200 emails, a classification model correctly predicted 150 spam emails and 30 ham emails. What is the accuracy of the model?	03	CO 2	A
	(b)	Write a short note on Histogram.	04	CO 2	U
	(c)	Explain classification steps in detail.	07	CO 3	U
Q. 4	(a)	Give any three examples of supervised learning.	03	CO 3	R
	(b)	Explain the use of unsupervised learning in expression identification.	04	CO 4	U
	(c)	Explain simple linear regression using a graph explaining slope and intercept.	07	CO 3	U
		OR			
Q. 4	(a)	Give any three applications of multiple linear regression.	03	CO 3	R
	(b)	Give the difference between supervised learning and unsupervised learning.	04	CO 4	U
	(c)	Explain logistic regression in detail.	07	CO 3	U
Q.5	(a)	Explain data types used in Scikit-Learn.	03	CO 5	U
	(b)	Explain use of Pandas in Machine Learning.	04	CO 5	U
	(c)	Explain how the Market Basket Analysis uses the concepts of association analysis.	07	CO 4	U
		OR			
Q.5	(a)	Give the application of Matplotlib.	03	CO 5	U
	(b)	Give the difference between Pandas and NumPy.	04	CO 5	U
	(c)	Explain the k-means clustering method with a step-by-step algorithm.	07	CO 4	U

❖ ❖ ❖

MODEL QUESTION PAPER 2

DE – SEMESTER IV (NEW SYLLABUS) EXAMINATION

Subject: Fundamentals of Machine Learning (4341603)

Time: 2 hours 30 min **Total Marks: 70**

Instructions:

1. **Attempt all questions.**
2. **Make suitable assumptions wherever necessary.**
3. **Figures to the right indicate full marks.**

			Marks	CO (Course Outcome)	Cognitive Level (As per Revised Bloom's Taxonomy)
Q.1	(a)	State the appropriate data type of following examples. a) Height of students b) Blood Group of students c) Grade of students	03	CO 2	U
	(b)	Explain well-posed problem with suitable example.	04	CO 1	U
	(c)	Define box plot. Explain detailed interpretation of a box plot	07	CO 2	R
		OR			
	(c)	Explain, in details, the different strategies of addressing missing data values.	07	CO 2	R
Q.2	(a)	Give the difference between bagging and boosting.	03	CO 2	U
	(b)	Explain the use of Machine Learning approach in Robotics.	04	CO 1	U
	(c)	Explain, in details, the process of evaluating the performance of a classification model.	07	CO 3	U
		OR			
Q.2	(a)	Explain the main purpose of a descriptive model.	03	CO 2	U
	(b)	Explain the use of Python in the field of Machine Learning.	04	CO 1	U
	(c)	Explain, in details, Holdout method.	07	CO 2	U
Q. 3	(a)	Define outliers. Give one example.	03	CO 2	R
	(b)	Find mean and median for the following data: 4, 5, 7, 8, 8, 9, 11, 12, 14	04	CO 2	A
	(c)	Write a short note on Support Vector Machine.	07	CO 3	U
		OR			
Q. 3	(a)	Define predictive models. Give one example.	03	CO 2	R

	(b)	Find standard deviation for the following data: 5, 10, 15, 20, 25, 29	04	CO 2	A
	(c)	Draw and describe classification model in detail.	07	CO 3	U
Q. 4	(a)	Give the weakness of logistic regression.	03	CO 3	R
	(b)	Explain test data and training data in brief.	04	CO 3	U
	(c)	Generate large itemsets and association rules using Apriori algorithm on the following data set with minimum support value and minimum confidence value set as 50% and 75% respectively.	07	CO 4	A

TID	Item Purchased
T1	Cheese, Milk, Cookies
T2	Butter, Milk, Bread
T3	Cheese, Milk, Butter, Brread
T4	Butter, Bread

		OR			
Q. 4	(a)	Explain the applications of simple linear regression.	03	CO 3	R
	(b)	State the strengths and weakness of kNN algorithm.	04	CO 3	U
	(c)	During a research work, you found 7 observations as described with the data points below. You want to create 3 clusters from these observations using K-means algorithm. After first iteration, the clusters C1, C2, C3 has following observations: C1: {(2,2), (4,4), (6,6)} C2: {(0,4), (4,0)} C3: {(5,5), (9,9)} If you want to run a second iteration then what will be the cluster centroids?	07	CO 4	A
Q.5	(a)	How can we create 2D arrays in NumPy?	03	CO 5	A
	(b)	Explain features of Scikit-Learn.	04	CO 5	U
	(c)	Describe the concept of clustering using appropriate real-world example.	07	CO 4	U
		OR			
Q.5	(a)	How can we plot a horizontal line in matplotlib?	03	CO 5	A
	(b)	Explain the steps to import csv file in Pandas.	04	CO 5	U
	(c)	Compare and contrast supervised learning and unsupervised learning.	07	CO 4	U

❖ ❖ ❖

MODEL QUESTION PAPER 3

DE – SEMESTER IV (NEW SYLLABUS) EXAMINATION

Subject: Fundamentals of Machine Learning (4341603)

Time: 2 hours 30 min **Total Marks: 70**

Instructions:

1. **Attempt all questions.**
2. **Make suitable assumptions wherever necessary.**
3. **Figures to the right indicate full marks.**

			Marks	CO (Course Outcome)	Cognitive Level (As per Revised Bloom's Taxonomy)
Q.1	(a)	Give the difference between qualitative data and quantitative data.	03	CO 2	R
	(b)	Describe use of R in Machine Learning.	04	CO 1	R
	(c)	Give the difference between Supervised Learning, Unsupervised Learning and Reinforcement Learning.	07	CO 1	U
		OR			
	(c)	Define Machine Learning. Explain any four applications of Machine Learning in brief.	07	CO 1	U
Q.2	(a)	Write basic steps for feature subset selection.	03	CO 2	R
	(b)	Explain data pre-processing in brief.	04	CO 2	U
	(c)	Describe K-fold cross validation method in detail.	07	CO 2	U
		OR			
Q.2	(a)	Explain the factors which lead to the data quality issues.	03	CO 2	R
	(b)	Write a brief note on dimensionality reduction.	04	CO 2	U
	(c)	Describe ensemble learning approach in detail.	07	CO 2	U
Q. 3	(a)	Give the difference between classification and regression.	03	CO 3	U
	(b)	Define following terms: Support Vectors, Hyperplane, Margin	04	CO 3	R
	(c)	Define Classification. Explain classification learning steps in detail using flowchart.	07	CO 3	U
		OR			
Q. 3	(a)	Give any three examples of supervised learning in Industry 4.0.	03	CO 3	R
	(b)	Compare and contrast between Single linear regression and multiple linear regression.	04	CO 3	U
	(c)	Write and discuss kNN algorithm.	07	CO 3	U
Q. 4	(a)	Define: Support, Confidence	03	CO 4	R

	(b)	Draw the diagram which shows the clustering process.	04	CO 4	U
	(c)	Describe applications of unsupervised learning in detail.	07	CO 4	A
		OR			
Q. 4	(a)	State pros and cons of k-means clustering algorithm.	03	CO 4	R
	(b)	Give the difference between classification and clustering.	04	CO 4	U
	(c)	Write and explain Apriori algorithm with example.	07	CO 4	A
Q.5	(a)	Explain any three applications of Pandas.	03	CO 5	U
	(b)	How to plot a vertical line and a horizontal line using Matplotlib? Give the code.	04	CO 5	A
	(c)	Write a Python program to load the iris data from a given csv file into a dataframe and print the shape of the data, type of the data and first 3 rows using Scikit-Learn.	07	CO 5	A
		OR			
Q.5	(a)	Explain any three applications of Scikit-Learn.	03	CO 5	U
	(b)	How can you split data into training and test data using Pandas? Give the code.	04	CO 5	A
	(c)	Write a program to cluster a set of points using K-means. Training and test data must be provided explicitly.	07	CO 5	A

❖ ❖ ❖