SIX WEEKS SUMMER TRAINING

on

DATA SCIENCE USING PYTHON

by

CIPHER SCHOOLS

A training report

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology

Computer Science & Engineering

Submitted to

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB

From 10/06/2023 to 20/07/2023

SUBMITTED BY

Name of student: ASHUTOSH KUMAR

Registration Number: 12105223

Signature of the student: 

List of Contents

Student Declaration

To whom so ever it may concern

I, **ASHUTOSH KUMAR, REG. NO.- 12105223**, hereby declare that the six weeks summer training on "**DATA SCIENCE USING PYTHON**" at **CIPHER SCHOOLS** from **JUNE,2023** to **JULY, 2023,** under the guidance of **MR. ARUPARNA MAITY**(Data Science tutor at Cipher schools)is a record of original work for the partial fulfilment of the requirements for the award of the degree, **B.Tech CSE, LOVELY PROFESSIONAL UNIVERSITY, PUNJAB**.

Name of student: ASHUTOSH KUMAR

Registration Number: 12105223

Signature of the student: 

Dated: 18/08/2023

TRAINING CERTIFICATE FROM CIPHER SCHOOLS

ABOUT CIPHER SCHOOLS

CipherSchools founded by Mr. Anurag Mishra, is a leading online training platform that offers a wide range of training programs in various fields of tech like programming, web development, mobile app development, Data Science, AI & ML,etc. It also provides Live Summer Training Programs in collaboration with different universities all over the country for different domains.

List of Abbreviations

S.NO.	Abbreviations	Full form/meaning
1.	AI	Artificial Intelligence
2.	ML	Machine Learning
3.	PDF	Probability Density Function
4.	w.r.t	With respect to
5.	EDA	Exploratory Data Analysis
6.	corr	Correlation
7.	var	Variance
8.	TSS	Total Sum of Squares
9.	ESS	Explained Sum of Squares
10.	RSS	Residual Sum of Squares
11.	KNN	K-Nearest Neighbours
12.	TP, TN	True Positive, True Negative
13.	FP, FN	False Positive, False Negative
14.	SVM	Support Vector Machine
15.	MSE	Mean squared Error
16.	I.G.	Information Gain
17.	G.I.	Gini Index

Chapter -1

1. Introduction of the course undertaken

Technology/Language/Modules/Libraries learnt:

1.1 What is Data Science?

Data science is an interdisciplinary field that involves extracting knowledge and insights from data using various techniques from statistics, computer science, machine learning, artificial intelligence and domain expertise. It encompasses a wide range of activities, including data collection, data cleaning and preprocessing, data analysis, exploratory data analysis, predictive analysis, feature engineering, model building, and deploying solutions. Data scientists use their skills to uncover patterns, make predictions, and generate valuable insights from data to support decision-making processes.

Data Analysis: Data analysis is the process of inspecting, cleaning, transforming, and modelling data in order to discover useful information, draw conclusions, and support decision-making. It involves applying statistical and computational methods to identify patterns, trends, and relationships within the data. Data analysis can be exploratory, aiming to uncover insights from data that weren't initially known, or confirmatory, where existing hypotheses are tested against the data. The goal of data analysis is to extract actionable insights and draw meaningful conclusions from the data.

Predictive Analysis: Predictive analysis is a subset of data analysis that focuses on predicting future outcomes or trends based on historical data. It involves the use of statistical and machine learning techniques to build predictive models that can make informed predictions about future events or behaviours. Predictive analysis uses historical data as input and generates predictions as output, helping businesses and organizations anticipate future scenarios and make more informed decisions. Common applications include sales forecasting, customer churn prediction, stock market forecasting, and medical diagnosis.

In summary, data science involves a broader set of activities that encompass data analysis and predictive analysis. Data analysis is the process of exploring and drawing conclusions from data, while predictive analysis specifically focuses on building models to make predictions about future outcomes based on historical data.

1.2 Why Python is widely used for Data Science?

Python is widely used for data science for several reasons:

1. **Ease of Use and Readability:** Python is known for its simple and readable syntax, which makes it accessible to both beginners and experienced programmers. This readability allows data scientists to focus on the logic of their analysis rather than getting bogged down in complex syntax.
2. **Rich Ecosystem of Libraries:** Python has a vast ecosystem of open-source libraries and frameworks specifically designed for data science, such as NumPy, pandas, Matplotlib, Seaborn, scikit-learn, TensorFlow, and PyTorch. These libraries provide pre-built functions and tools for data manipulation, analysis, visualization, and machine learning, saving a lot of development time.
3. **Community and Documentation:** Python has a large and active community of data scientists, researchers, and developers. This means that there are plenty of resources, tutorials, and forums available for troubleshooting, learning, and sharing ideas.
4. **Interoperability:** Python can easily integrate with other programming languages like C, C++, and Java, making it flexible for various use cases. This interoperability is essential for incorporating existing software components and systems into data science workflows.
5. **Data Analysis and Visualization:** Libraries like pandas and Matplotlib provide powerful tools for data manipulation and visualization. Data scientists can easily load, clean, and transform data, and then create informative visualizations for exploratory data analysis.
6. **Machine Learning and Deep Learning:** Python's machine learning and deep learning libraries (scikit-learn, TensorFlow, PyTorch, etc.) are widely adopted for building and training complex models. These libraries provide user-friendly interfaces and extensive functionalities for a wide range of machine learning and AI tasks.
7. **Open Source Philosophy:** Python is an open-source programming language, which means it's freely available and can be customized according to specific needs. This fosters innovation and collaboration in the data science community.
8. **Support for Jupyter Notebooks:** Python's integration with Jupyter Notebooks offers an interactive and visual way to perform data analysis. It allows data scientists to combine code, visualizations, and explanatory text in a single document.

Overall, Python's combination of simplicity, a rich library ecosystem, and a supportive community make it an ideal choice for data scientists to efficiently tackle a wide variety of data-related tasks.

1.3 Jupyter Notebook

Jupyter Notebook for Python is a popular interactive environment that combines code execution, data analysis, and documentation. It allows us to create documents called notebooks that contain live Python code, visualizations, explanatory text, and more. Key points about Jupyter Notebook for Python:

1. **Interactive Coding:** Write and execute Python code in cells, seeing results immediately below each cell.
2. **Data Analysis:** Perform data manipulation, exploration, and visualization using libraries like pandas and Matplotlib.
3. **Visualizations:** Generate charts, graphs, and plots directly within the notebook for better data understanding.
4. **Markdown Support:** Use Markdown cells to provide explanations, documentation, and formatted text.
5. **Rich Media:** Embed images, videos, LaTeX equations, and interactive widgets.
6. Step-by-Step: Break down complex analyses into smaller code cells, allowing for easy debugging and understanding.
7. **Exploratory Data Analysis (EDA):** Ideal for exploring datasets, understanding patterns, and identifying insights.
8. **Reproducibility:** Share notebooks to ensure others can reproduce your analysis and results.

Jupyter Notebook provides a flexible and interactive environment for Python programming, making it a valuable tool for data scientists, researchers, educators, and developers.

1.4 Software requirements analysis

Software requirements analysis for data science using Python and Jupyter Notebook involves identifying and specifying the software components, tools, and resources needed to effectively conduct data analysis tasks using Jupyter Notebook. Here's an overview of the key aspects to consider:

Jupyter Notebook Installation:

Python must be installed on our system.

Installing Jupyter Notebook using package managers like pip or conda.

Verifying the installation and launching Jupyter Notebook.

Required Libraries and Packages:

Identifying the libraries needed for data manipulation (pandas), visualization (**plotly express**, **Seaborn**), machine learning (**scikit-learn**),and other specific tasks.

Installing the required libraries using pip or conda. Eg.- pip install pandas.

Data Access and Storage:

Specifying how we will access and store data. This could include reading from CSV files, databases, APIs, or web scraping.

Visualization and Plotting:

Identifying the types of visualizations we'll create (line charts, bar plots, etc.).

Installing visualization libraries like **plotly express** and **Seaborn** for creating plots and graphs.

Machine Learning and Analytics:

Installing machine learning libraries such as **scikit-learn** for various algorithms like regression, classification, clustering, etc.

Chapter-2

2.Mini- Project: Exploratory Data Analysis(EDA)

2.1. What is EDA and why it is performed on a dataset?

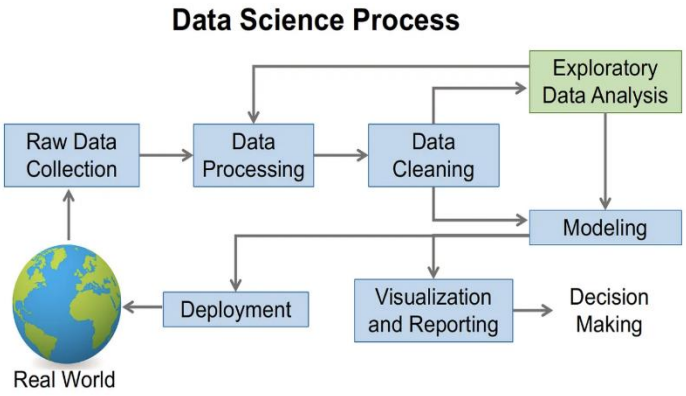
Exploratory Data Analysis (EDA) is a crucial initial step in the data analysis process. It involves summarizing, visualizing, and understanding the main characteristics of a dataset before applying more complex statistical or machine learning techniques. EDA aims to uncover patterns, relationships, anomalies, and insights within the data, which can guide further analysis and decision-making. Here's why EDA is done:

1. **Data Understanding:** EDA helps you become familiar with the data you're working with. This includes understanding the structure, types of variables, and their meanings. This understanding is essential for making accurate interpretations during analysis.
2. **Data Cleaning:** During EDA, you identify and address issues like missing values, duplicates, and outliers. Cleaning the data ensures that subsequent analysis is based on accurate and reliable information.
3. **Detecting Anomalies:** EDA can help identify unusual or anomalous data points that might be errors or require further investigation. These anomalies could impact the quality of your analysis and conclusions.
4. **Pattern Recognition:** EDA helps in identifying trends, patterns, and relationships between variables in the dataset. This information can be used to formulate hypotheses or guide further statistical analyses.
5. **Variable Selection:** EDA can guide you in selecting the most relevant variables for analysis. Understanding the relationships between variables can help you focus on those that have the most impact.
6. **Feature Engineering:** For machine learning projects, EDA can provide insights into feature engineering. You can identify which features are most predictive or relevant for the target variable.
7. **Data Visualization**: EDA involves creating visualizations such as histograms, scatter plots, box plots, and more. These visualizations make it easier to understand the distribution of data, identify outliers, and visualize patterns.
8. **Hypothesis Generation:** EDA can help you formulate hypotheses about potential relationships or trends in the data. These hypotheses can be tested using more formal statistical methods.
9. **Decision-Making:** Insights gained from EDA can guide decision-making processes. For example, businesses can use EDA to understand customer behavior, optimize marketing strategies, or identify potential areas for improvement.
10. **Communication:** EDA visualizations and findings can be communicated to stakeholders or non-technical audiences. Visualizations make it easier to convey complex insights in a more understandable manner.
11. **Quality Assurance:** EDA is a part of data quality assurance. It allows you to identify inconsistencies or discrepancies in the data that might impact the quality of the final analysis.

In summary, EDA is conducted to explore, understand, and prepare the data for more advanced analysis techniques. It provides a foundation for making informed decisions, generating hypotheses, and extracting meaningful insights from the data.

Where is EDA in the Data Science Process

Before we delve into EDA, it is important to first get a sense of where EDA fits in the whole data science process.



With reference to the process chart from Wikipedia, after the data has been collected, it undergoes some processing before being cleaned and EDA is then performed. Notice that after EDA, we may go back to processing and cleaning of data, i.e., this can be an iterative process. Subsequently, we can then use the cleaned dataset and knowledge from EDA to perform modelling and reporting.

2.2 Prerequisite knowledge before performing EDA

1. Modules and Packages

- **Module:**

In Python, a module is a single file containing Python code.

Modules can define functions, classes, and variables that can be used in other Python programs.

Modules provide a way to organize code and break it into smaller, manageable pieces.

To use a module in your code, you import it using the import keyword followed by the module name.

Example: import math imports the built-in math module, which contains various mathematical functions.

- **Package:**

A package is a collection of related modules grouped together in a directory hierarchy.

Packages provide a higher level of organization for large projects.

Packages have a special file called __init__.py inside their directories to indicate that they are packages.

Submodules within a package are accessed using dot notation, like package.module.

Example: The numpy package contains various modules for numerical computations, and you can import its array module with import numpy.array.

Installing a package: pip install numpy

2. Statistics

Statistics refers to the branch of mathematics that involves the collection, analysis, interpretation, presentation, and organization of data. It provides methods and techniques for dealing with data in order to gain insights, make informed decisions, and draw meaningful conclusions.

There are two main aspects of statistics:

a) **Descriptive Statistics**: Descriptive statistics involves summarizing and describing data using various measures such as:
   - **Measures of central tendency:** Mean, median, and mode.
   - **Measures of dispersion:** Standard deviation, variance.
   - **Visualizations: Graphs, charts, and plots to represent data visually.**

b) **Inferential Statistics:** Inferential statistics involves drawing conclusions and making predictions about a larger population based on a sample of data. Key concepts include:
   - **Hypothesis testing:** Evaluating whether observed effects are statistically significant.
   - **Confidence intervals:** Estimating the range within which a population parameter is likely to fall.
   - **Regression analysis:** Modeling relationships between variables.
   - **Probability distributions:** Describing the likelihood of different outcomes.
   - **Sampling techniques:** Choosing representative samples from a larger population.

Statistics is used across various fields such as science, social science, business, engineering, and more, to analyze data and gain insights into patterns, trends, relationships, and

uncertainties. In data-driven fields like data science and analytics, statistics forms the foundation for making data-driven decisions, building predictive models, and extracting valuable information from data.

3. Data Analysis with Numpy and Pandas

a. Intro to Numpy

NumPy (Numerical Python) is a fundamental package for scientific computing in Python. It provides support for working with large, multi-dimensional arrays and matrices, as well as a wide range of mathematical functions to operate on these arrays. NumPy is an essential library for data manipulation, numerical analysis, and machine learning tasks.

b. Creating an array

import numpy as np

Create two arrays

a = np.array([1, 2, 3])

b = np.array([4, 5, 6])

```
In [1]: import numpy as cipher_np #Importing numpy
        import pandas as cipher_pd

In [3]: dummy_list = [1, 2, 3, 4, 5]  #List creation
        dummy_list
Out[3]: [1, 2, 3, 4, 5]

In [4]: type(dummy_list)
Out[4]: list

In [5]: dummy_array = cipher_np.array(dummy_list) #Passing List as numpy array
        dummy_array
Out[5]: array([1, 2, 3, 4, 5])

In [6]: type(dummy_array)
Out[6]: numpy.ndarray
```

Figure 1Creating np array using list

c. Indexing and Slicing

Indexing

```
In [7]: print(dummy_list[0])
        print(dummy_array[0])
1
1
```

Slicing

```
In [8]: len(dummy_array)
Out[8]: 5

In [9]: # create a random nd-array
        cipher_np.random.randint(-10, 100, (4, 5))
Out[9]: array([[ 13,  97,  12,  84,  54],
               [ 27, -10,  24,  47,  15],
               [ 32,  33,  21,  17,  57],
               [ 22,  57,  19,  51,  79]])

In [10]: # Create an nd-array of 1's
         cipher_np.ones((3, 4))
Out[10]: array([[1., 1., 1., 1.],
                [1., 1., 1., 1.],
                [1., 1., 1., 1.]])
```

```
In [11]: dummy_md_list = [[1, 2, 3, 4, 5], [11, 22, 33, 44, 55], [111, 222, 333, 444, 555]]
         dummy_md_list

Out[11]: [[1, 2, 3, 4, 5], [11, 22, 33, 44, 55], [111, 222, 333, 444, 555]]

In [12]: dummy_md_array = cipher_np.array(dummy_md_list)
         dummy_md_array

Out[12]: array([[  1,   2,   3,   4,   5],
                [ 11,  22,  33,  44,  55],
                [111, 222, 333, 444, 555]])

In [ ]: # dummy_md_array = dummy_md_array[:, :-3] ## XXXXXXX

In [13]: # Checking the order of the multi-dimensional array
         dummy_md_array.shape

Out[13]: (3, 5)

In [14]: # Get ALL rows and ALL FIRST 3 COLUMNS of the multi-dimensional array
         # Expected output:
         # [[1, 2, 3],
         # [11, 22, 33],
         # [111, 222, 333]]

         # How to do slicing in nd-array?
             # "," - Before the comma, we mention row indices (as list or as single integers), and after the comma we mention column indi
             # ":" - Collon is used to access all the rows or all the columns in the nd-array
             # ":4" - This slices the nd-array till 4th index

         dummy_md_array[:, :]

Out[14]: array([[  1,   2,   3,   4,   5],
                [ 11,  22,  33,  44,  55],
                [111, 222, 333, 444, 555]])
```

d. Statistical Operations using Numpy

array([[1, 2, 3, 4, 5],

[11, 22, 33, 44, 55],

[111, 222, 333, 444, 555]])

Sum

Sum of ALL elements in the md-array

dummy_md_array.sum()

1845

Row-wise sum of elements in md-array

dummy_md_array.sum(axis = 1

array([15, 165, 1665])

Column-wise sum of elements in md-array

dummy_md_array.sum(axis = 0)

array([123, 246, 369, 492, 615])

Mean

Mean of ALL elements in the md-array

dummy_md_array.mean()
123.0

Variance

# Variance of ALL elements in the md-array
dummy_md_array.var()
30495.333333333332

Standard deviation

# Standard-deviation of ALL elements in the md-array
dummy_md_array.std()
174.62913082682778

e. Introduction to Pandas

Pandas is a popular Python library for data manipulation and analysis. It provides powerful data structures and tools designed to make working with structured data, such as tabular data and time series, more intuitive and efficient. Pandas is widely used in data science, machine learning, and analytics tasks.

Key Features:

- **DataFrame:** The core of Pandas is the DataFrame, a two-dimensional tabular data structure similar to a spreadsheet or a SQL table. It allows you to store and manipulate data with rows and columns.
- **Creating a dataframe**

dummy_df=cipher_pd.DataFrame(dummy_md_array) #passing a mdarray as DataFrame

dummy_df

	0	1	2	3	4
0	1	2	3	4	5

	1	2	3	4	
0					
1	11	22	33	44	55
2	111	222	333	444	555

I. Read CSV files

csv_df = cipher_pd.read_csv("./dummy_csv.csv",index_col=0) #./dummy_csv.csv is the path of csv file in system

csv_df

	Col1	col2	col3	col4	col5	col6	col7
Row 1	1.0	2.0	3.0	4.0	5.0	6.0	7.0
Row 2	11.0	22.0	33.0	44.0	55.0	66.0	77.0
Row 3	111.0	222.0	333.0	444.0	555.0	666.0	777.0
Row 4	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Check the column names of the dataframe

csv_df.columns

Index(['Col1', 'col2', 'col3', 'col4', 'col5', 'col6', 'col7'], dtype='object')

Get the row names of the dataframe

csv_df.index

Index(['Row 1', 'Row 2', 'Row 3', 'Row 4'], dtype='object')

II. Reading XLSX data

```
mangoes_xlsx_df = cipher_pd.read_excel("./mangoes_basket.xlsx", engine = "openpyxl",
index_col = 0)
```

```
mangoes_xlsx_df
```

	weight	length	diameter	age
Mango 1	1.0	10.0	10	1
Mango 2	1.4	12.0	21	2
Mango 3	1.1	1.0	11	5
Mango 4	332.0	3.0	13	7
Mango 5	1.3	NaN	9	1
Mango 6	1.6	NaN	10	3
Mango 7	1.7	12.0	11	5
Mango 8	1.9	34.0	22	4
Mango 9	1.8	23.0	33	0
Mango 10	1.2	21.0	13	10

 # ALL Statistical properties of the dataframe

```
mangoes_xlsx_df.describe()
```

	weight	length	diameter	age
count	10.000000	8.000000	10.000000	10.000000
mean	34.500000	14.500000	15.300000	3.800000
std	104.531282	10.967484	7.703535	3.084009

	weight	length	diameter	age
min	1.000000	1.000000	9.000000	0.000000
25%	1.225000	8.250000	10.250000	1.250000
50%	1.500000	12.000000	12.000000	3.500000
75%	1.775000	21.500000	19.000000	5.000000
max	332.000000	34.000000	33.000000	10.000000

- **Series:** A Series is a one-dimensional labeled array that can hold various data types. It serves as the building block for columns within a DataFrame.

```
# Series

mangoes_xlsx_df["length"]

Mango 1     10.0
Mango 2     12.0
Mango 3      1.0
Mango 4      3.0
Mango 5      NaN
Mango 6      NaN
Mango 7     12.0
Mango 8     34.0
Mango 9     23.0
Mango 10    21.0
Name: length, dtype: float64
```

III. Handling missing Values-

```
# Handle missing values

mangoes_xlsx_df.fillna(mangoes_xlsx_df["length"].mean(), inplace = True)

#Filling missing length with the average length
```

Link for jupyter notebook implementation:- Data Analysis Using Numpy and Pandas

2.3 Basic steps to start an EDA

- Reading / importing / download the dataset into Python environment
- Size of data - w.r.t. number of rows and columns, or disk space
- Columns / fields - important to know what the dataset is about
- Datatypes - what are different datatypes in the columns
- Unique values - how many different values are there in each columns
- Value counts - what are the counts of each element in a particular column
- Missing values - check and handle
- Check the statistical properties
- Correlation - within columns
- Visualization - scatter plots, bar charts, histograms, pie charts
- Deriving basic conclusions

1. **EDA on Iris Dataset:-**

   Link:- **EDA Iris**

2. **Final project on EDA**

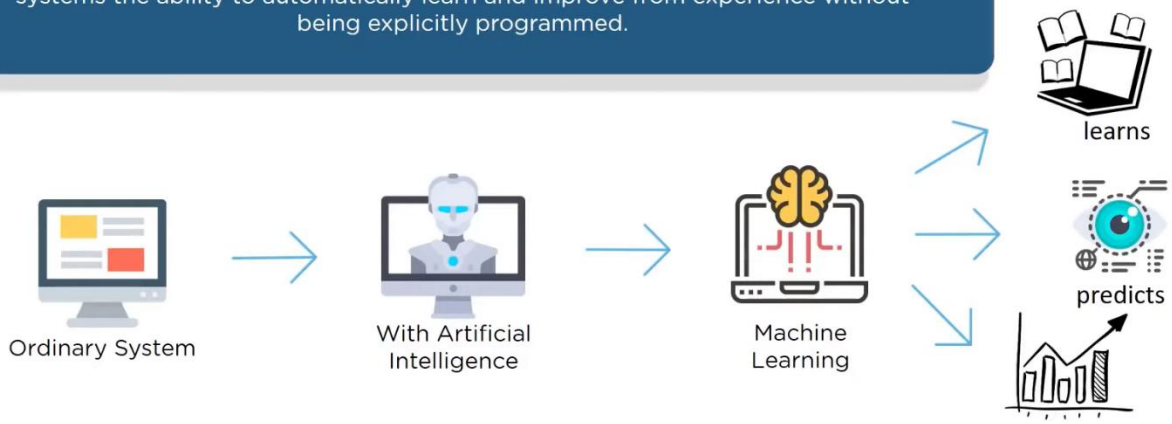
   Link for jupyter notebook:- **EDA project**

CHAPTER-3

SUPERVISED MACHINE LEARNING

What is Machine Learning?

Machine Learning is an application of Artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed.



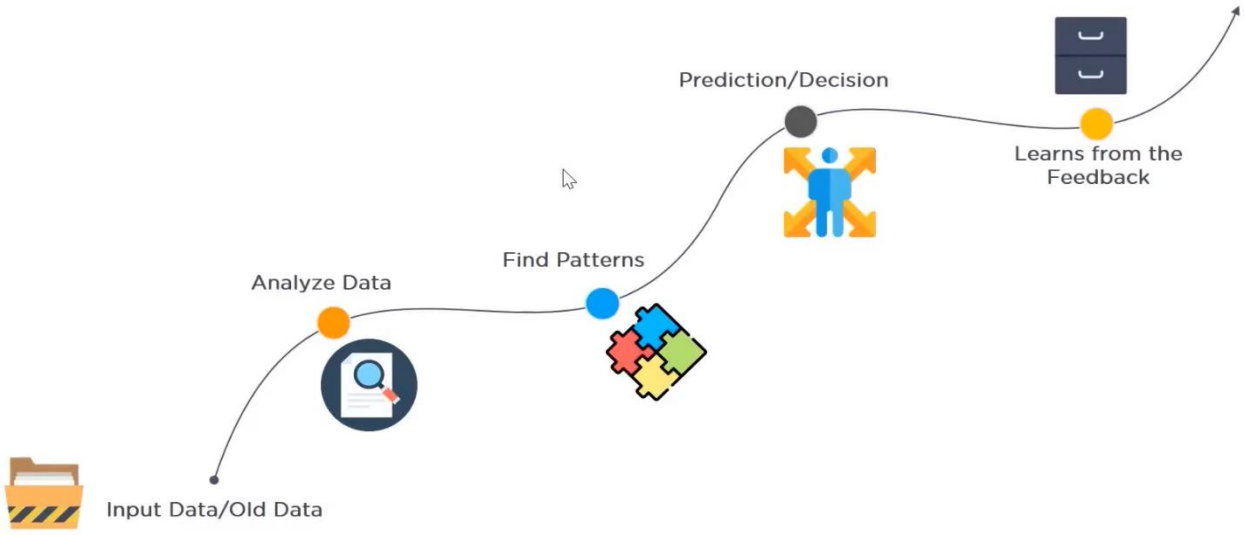
Machine Learning is said as a subset of **artificial intelligence** that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own. The term machine learning was first introduced by **Arthur Samuel** in **1959**. We can define it in a summarized way as:

> "Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed."

With the help of sample historical data, which is known as **training data**, machine learning algorithms build a **mathematical model** that helps in making predictions or decisions without being explicitly programmed. Machine learning brings computer science and statistics together for creating predictive models. Machine learning constructs or uses the algorithms that learn from historical data. The more we will provide the information, the higher will be the performance.

A machine has the ability to learn if it can improve its performance by gaining more data.

Machine Learning process



MACHINE LEARNING

SUPERVISED LEARNING

UNSUPERVISED LEARNING

REGRESSION

CLASSIFICATION

1. Linear Regression

1. Logistic Regression
2. KNN(K Nearest Neighbours)
3. SVM(Support Vector Machine)
4. Decision Trees
5. Random Forest

3.1 Supervised Machine 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.

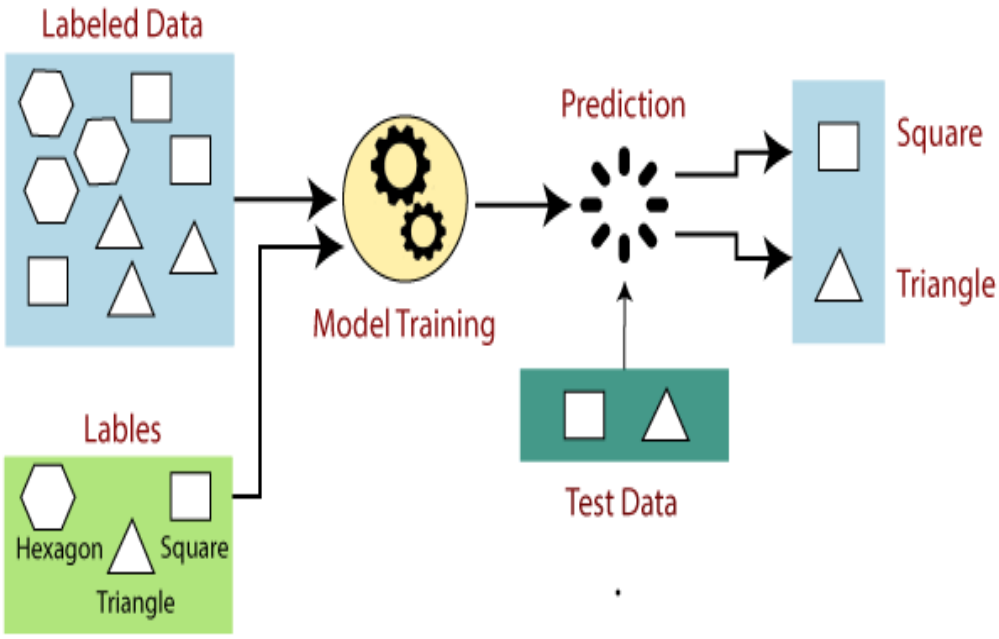
In supervised learning, the training data provided to the machines work as the supervisor that teaches the machines to predict the output correctly. It applies the same concept as a student learns in the supervision of the teacher.

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)**.

How Supervised Learning Works?

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 (a subset of the training set), and then it predicts the output.

The working of Supervised learning can be easily understood by the below example and diagram:

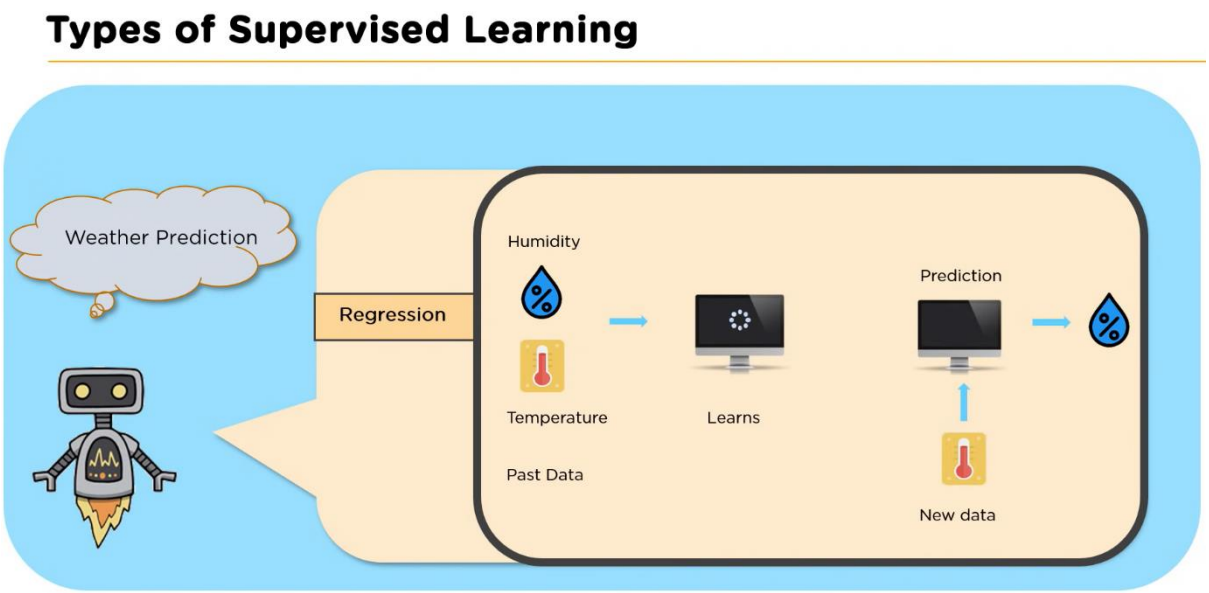


3.1.1 Regression

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.

In this journey, we will grasp the significance of choosing appropriate evaluation metrics, such as mean squared error and R-squared, to assess the accuracy of regression models. We will also touch upon the potential challenges and pitfalls, such as overfitting and underfitting, that researchers and practitioners encounter when constructing and deploying regression models in real-world scenarios.

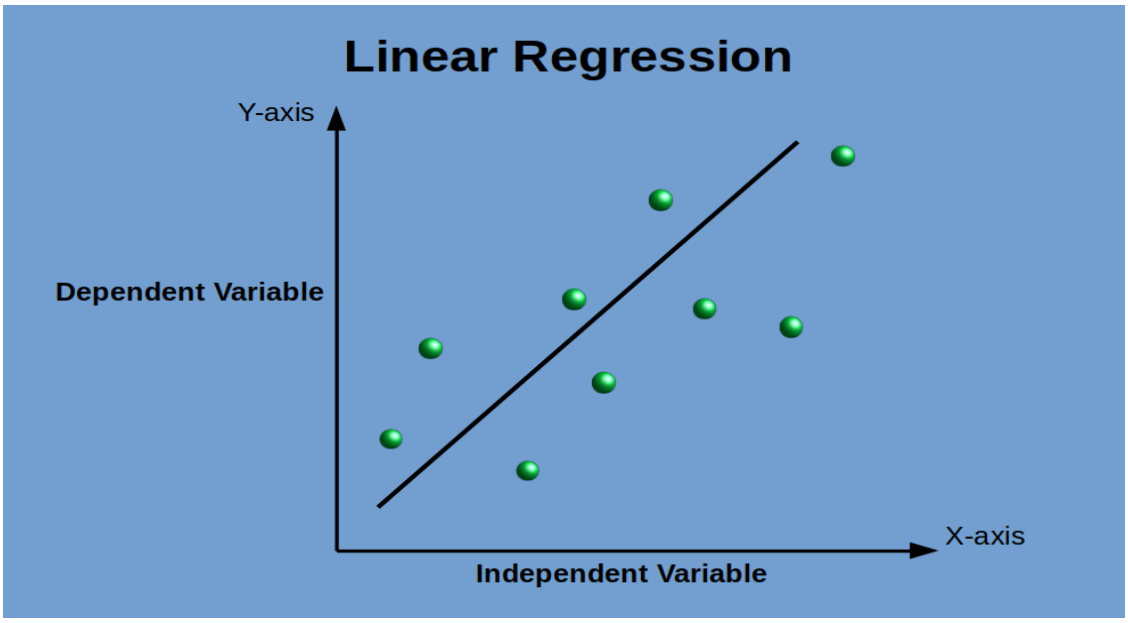
Ultimately, regression equips us with the tools to make informed predictions about numerical outcomes, enabling businesses to make data-driven decisions, researchers to extract insights from empirical observations, and individuals to gain a deeper understanding of the relationships embedded within their data. So, let's embark on this exploration of regression in supervised learning, where mathematical concepts converge with data to pave the way for accurate and valuable predictions.



a. **<u>Linear Regression:</u>** Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as **sales, salary, age, product price,** etc.

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

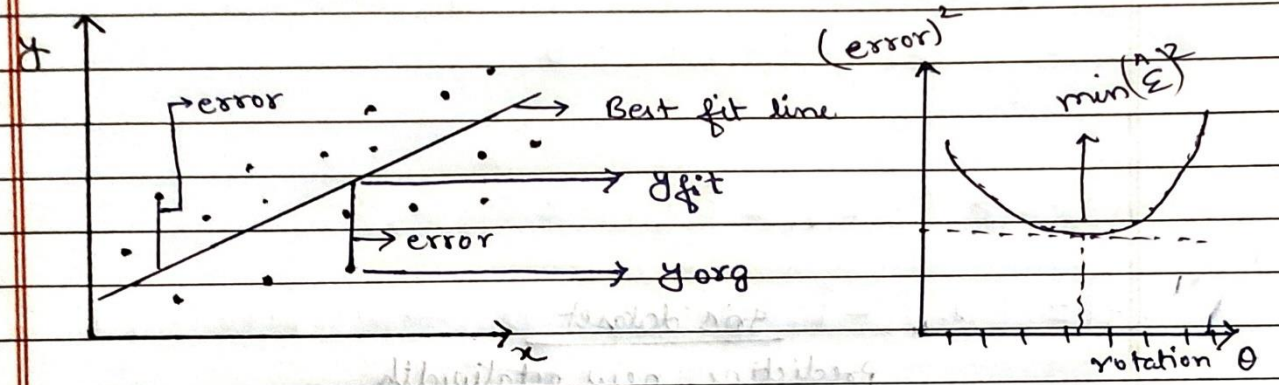
The linear regression model provides a sloped straight line representing the relationship between the variables. Consider the below image:



- **Finding the best fit line:**

When working with linear regression, our main goal is to find the best fit line that means the error between predicted values and actual values should be minimized. The best fit line will have the least error.

LINEAR REGRESSION



$$error = |y_{org} - y_{fit}|$$

→ y_{org} = dependent variable = y

→ y_{fit} = fitted y = on reg. line = $\hat{y}$

→ Parameters ↳ which we will optimize using our sample. $\hat{\beta}_0$ = constant $\hat{\beta}_1$ = slope coefficient.

* EQUATION OF FITTED LINE

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x \quad (\because y = mx + c) \quad \text{①}$$

($\hat{\beta}_0$ → c, $\hat{\beta}_1$ → m)

→ $y - \hat{y} = \hat{\varepsilon}$ → Error (Residual)

From ①.

$$y = \hat{\beta}_0 + \hat{\beta}_1 x + \hat{\varepsilon} \quad \text{②}$$

$\hat{\varepsilon}$ → Residual.

y → original ordinate value

$\hat{\beta}_0 + \hat{\beta}_1 x$ → Fitted/optimized intercept/slope

RESIDUAL/ERROR:

In the context of linear regression, the error or residual refers to the difference between the actual observed values (y-actual) and the predicted values (y-predicted) obtained from the fitted regression line for a given set of x values.

Mathematically, the error (e or ε) for each data point is given by:

e= (y-actual) – (y-predicted)

In the equation of the fitted line (y=mx+b), the residuals are the vertical distances between the observed data points and the corresponding points on the line. The goal of linear regression is to minimize the sum of the squared residuals, which is why it's often referred to as the least squares method.

The equation for the fitted line with consideration for the error term becomes:

$y=mx+b+\varepsilon$

Here, ε represents the random error or residual term that captures the variability in the data that isn't explained by the linear relationship between x and y. The linear regression process seeks to find the best-fitting values of m and b that minimize the sum of the squared residuals **($\sum\varepsilon^2$),** thereby providing the line that best represents the linear relationship between the variables.

In practice, after performing linear regression, we can analyse the residuals to assess the quality of the fit. A well-fitted model will have residuals that are close to zero and distributed randomly around the line. If you observe patterns or trends in the residuals, it could indicate that the linear model isn't capturing all the underlying relationships in the data.

Page No.
Date / 20

Q. At which value of $\hat{\beta}_1$ & $\hat{\beta}_0$ we can achieve min $\hat{\varepsilon}^2$?

(i) For β_0

$\sum \varepsilon^2$ ↓ sum of (error)^2

$\Rightarrow \frac{\delta}{\delta \hat{\beta}_0} \sum \hat{\varepsilon}^2 = 0$

$\Rightarrow \frac{\delta}{\delta \hat{\beta}_0} \sum (y - \hat{\beta}_0 - \hat{\beta}_1 x)^2 = 0$ From (2)

$\Rightarrow 2 \sum (y - \hat{\beta}_0 - \hat{\beta}_1 x)(-1) = 0$

$\Rightarrow \boxed{\sum (y - \hat{\beta}_0 - \hat{\beta}_1 x) = 0}$ → First Normal equation.

↓ → $\sum (y - \hat{y}) = 0$

$\boxed{\sum \hat{\varepsilon} = 0}$

$\Rightarrow \sum_i y_i - \sum_i \hat{\beta}_0 - \sum_i \hat{\beta}_1 x = 0$

$\Rightarrow \sum y - n\hat{\beta}_0 - \hat{\beta}_1 \sum x = 0$

$\Rightarrow \frac{1}{n} \sum y - \hat{\beta}_0 - \frac{\hat{\beta}_1}{n} \sum x = 0$

↓ → mean

$\Rightarrow \bar{y} - \hat{\beta}_0 - \hat{\beta}_1 \bar{x} = 0$

$\Rightarrow \boxed{\bar{y} = \hat{\beta}_0 + \hat{\beta}_1 \bar{x}}$ — (2)

$\boxed{\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 x}$ — (3)

Page No.
Date / 20

For $\hat{\beta}_1$

$$\Rightarrow \frac{\partial}{\partial \hat{\beta}_1} \Sigma (y - \hat{\beta}_0 - \hat{\beta}_1 x)^2 = 0$$

$$\Rightarrow 2 \Sigma (y - \hat{\beta}_0 - \hat{\beta} x)(-x) = 0$$

$$\Rightarrow \boxed{\Sigma (y - \hat{\beta}_0 - \hat{\beta}_1 x)(x) = 0} \rightarrow \text{2nd Normal form} - (4).$$

$$\Rightarrow \Sigma [y - (\hat{\beta}_0 + \hat{\beta}_1 x)](x) = 0$$

$$\Rightarrow \Sigma (y - \hat{y})(x) = 0$$

$$\Rightarrow \boxed{\Sigma \hat{\varepsilon}(x) = 0}$$

From (3) & (4).

$$\Rightarrow \Sigma (y - \bar{y} + \hat{\beta}_1 \bar{x} - \hat{\beta}_1 x)(x) = 0$$

$$\Rightarrow \Sigma (yx - \bar{y}x + \hat{\beta}_1 \bar{x} x - \hat{\beta}_1 x^2) = 0$$

$$\Rightarrow \Sigma yx - n\bar{y}\bar{x} - \hat{\beta}_1 (\Sigma x^2 - n\bar{x}^2) = 0$$

$$\Rightarrow \Sigma yx - n\bar{y}\bar{x} = \hat{\beta}_1 (\Sigma x^2 - nx^2)$$

$$\Rightarrow \hat{\beta}_1 = \frac{\Sigma yx - n\bar{y}\bar{x}}{\Sigma x^2 - n\bar{x}^2} = \frac{\frac{1}{n}\Sigma (y - \bar{y})(x - \bar{x})}{\frac{1}{n}\Sigma (x - \bar{x})^2}$$

$$\boxed{\hat{\beta}_1 = \frac{\text{Corr}(x,y)}{\text{Var}(x)}} \qquad \boxed{\beta_0 = \bar{y} - \left[\frac{\text{Corr}(x,y)}{\text{Var}(x)}\right](x)}$$

3.1.2 Classification Algorithm

As we know, the Supervised Machine Learning algorithm can be broadly classified into Regression and Classification Algorithms. In Regression algorithms, we have predicted the output for continuous values, but to predict the categorical values, we need Classification algorithms.

What is the Classification Algorithm?

The Classification algorithm is a Supervised Learning technique that is used to identify the category of new observations on the basis of training data. In Classification, a program learns from the given dataset or observations and then classifies new observation into a number of classes or groups. Such as, Yes or No, 0 or 1, Spam or Not Spam, cat or dog, etc. Classes can be called as targets/labels or categories.

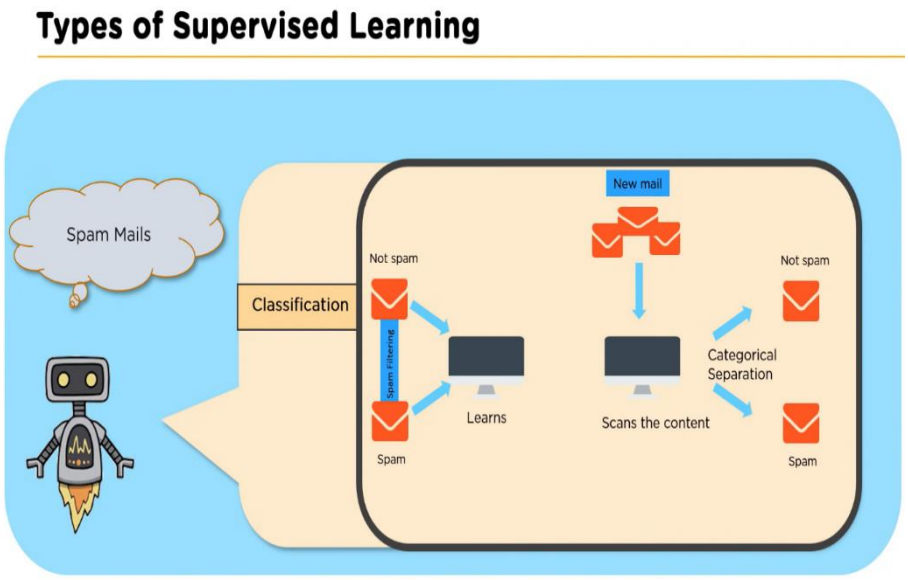
Unlike regression, the output variable of Classification is a category, not a value, such as "Green or Blue", "fruit or animal", etc. Since the Classification algorithm is a Supervised learning technique, hence it takes labeled input data, which means it contains input with the corresponding output.

In classification algorithm, a discrete output function(y) is mapped to input variable(x).

y=f(x), where y = categorical output

The best example of an ML classification algorithm is Email Spam Detector.

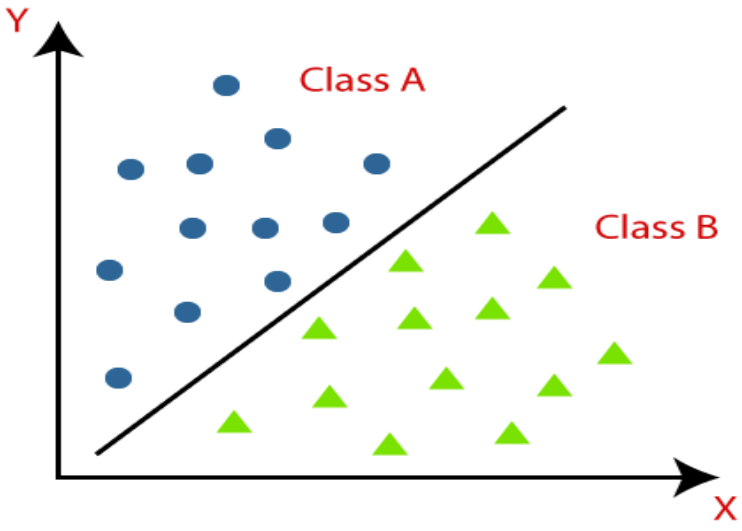
The main goal of the Classification algorithm is to identify the category of a given dataset, and these algorithms are mainly used to predict the output for the categorical data.



Classification algorithms can be better understood using the below diagram. In the below diagram, there are two classes, class A and

Class B. These classes have features that are similar to each other and dissimilar to other classes.

The algorithm which implements the classification on a dataset is known as a classifier.



There are two types of Classifications:

Binary Classifier: If the classification problem has only two possible outcomes, then it is called as Binary Classifier.
Examples: YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.

Multi-class Classifier: If a classification problem has more than two outcomes, then it is called as Multi-class Classifier.
Example: Classifications of types of crops, Classification of types of music.

Learners in Classification Problems:

In the classification problems, there are two types of learners:

Lazy Learners: Lazy Learner firstly stores the training dataset and wait until it receives the test dataset. In Lazy learner case, classification is done on the basis of the most related data stored in the training dataset. It takes less time in training but more time for predictions.
Example: K-NN algorithm, Case-based reasoning

Eager Learners: Eager Learners develop a classification model based on a training dataset before receiving a test dataset. Opposite to Lazy learners, Eager Learner takes more time in learning, and less time in prediction. Example: Decision Trees, Naïve Bayes, ANN.

Types of ML Classification Algorithms:

Classification Algorithms can be further divided into the Mainly two category:

- **Linear Models**
  - Logistic Regression
  - Support Vector Machines
- **Non-linear Models**
  - K-Nearest Neighbours
  - Kernel SVM
  - Naïve Bayes
  - Decision Tree Classification
  - Random Forest Classification

3.1.3 K-Nearest Neighbor(KNN) Algorithm:

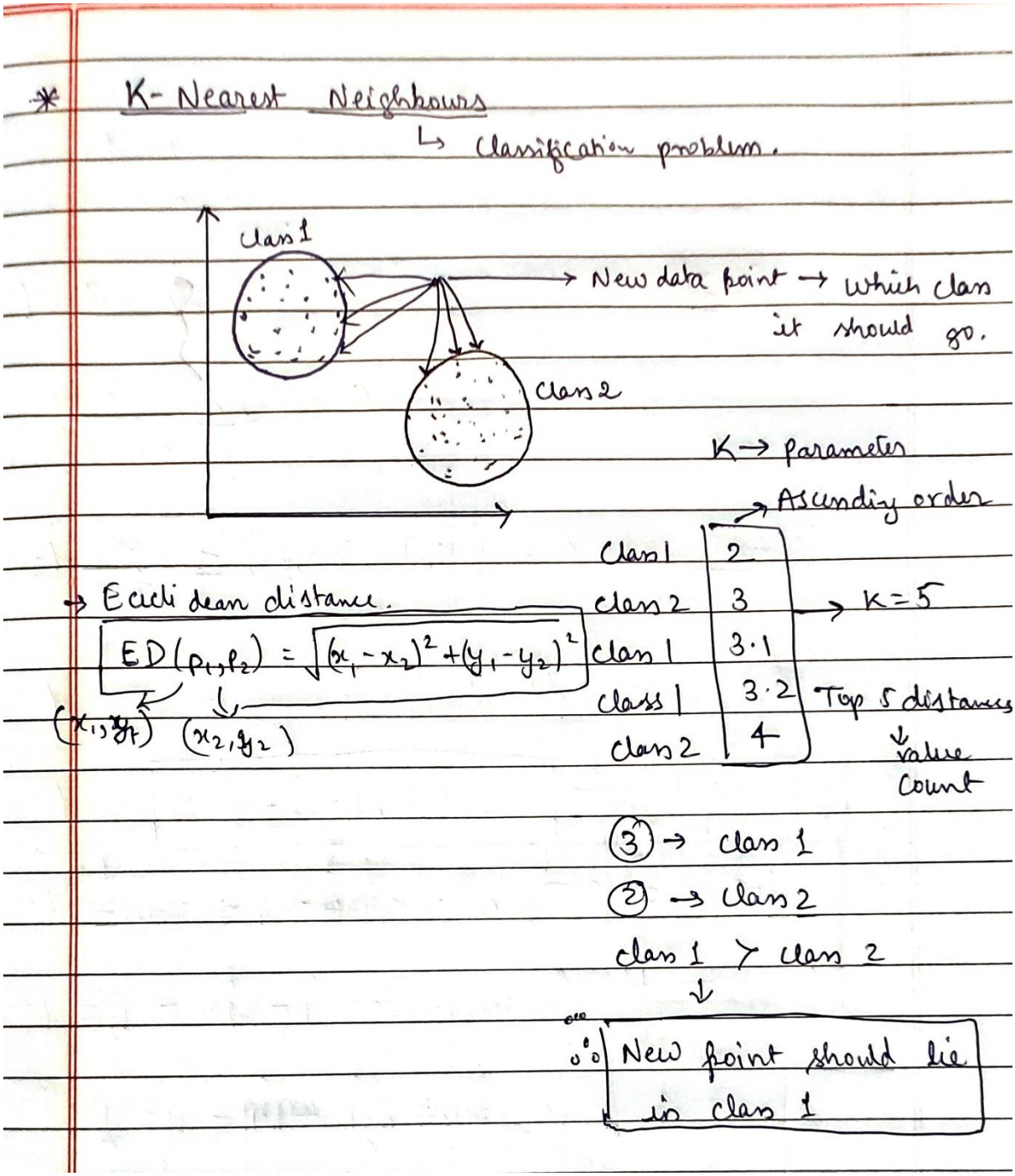
- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.
- It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

How does K-NN work?

The K-NN working can be explained on the basis of the below algorithm:

- **Step-1:** Select the number K of the neighbors

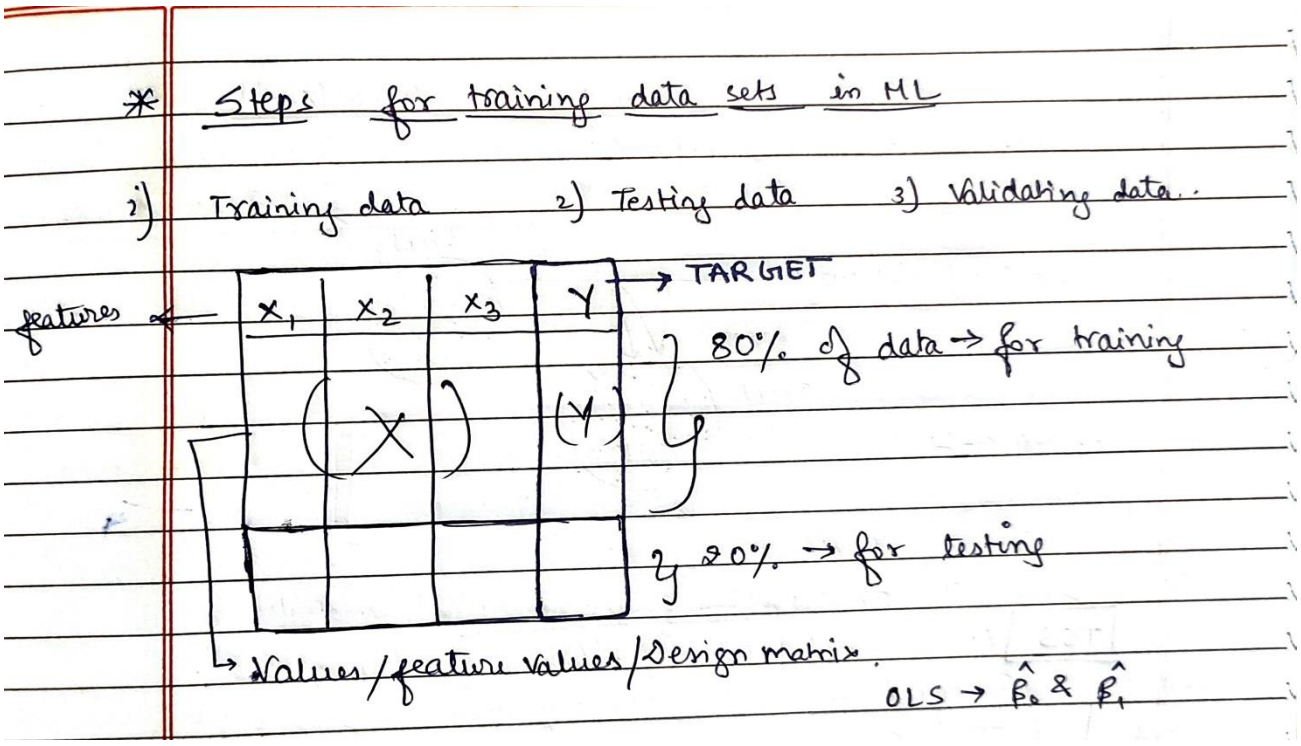
- **Step-2:** Calculate the Euclidean distance of **K number of neighbors**
- **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.
- **Step-4:** Among these k neighbors, count the number of the data points in each category.
- **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
- **Step-6:** Our model is ready.



Steps for training dataset model

In machine learning, it's common practice to divide your dataset into three main subsets: the training set, the validation set, and the testing set. This division helps in building, tuning, and evaluating your machine learning models effectively.

1. **Training Set:** The training set is the portion of your dataset used to train your machine learning model. This is where the model learns patterns, relationships, and features from the data. It's used to adjust the model's parameters and weights so that it can make accurate predictions on new, unseen data.
2. **Validation Set:** The validation set is used to fine-tune your model and make decisions about its architecture, hyperparameters, and other settings. It helps prevent overfitting by providing an independent dataset that the model has not seen during training. You use the validation set to evaluate the model's performance on data it hasn't been directly trained on and to choose the best combination of hyperparameters.
3. **Testing Set:** The testing set is the final evaluation step. After you have trained and tuned your model using the training and validation sets, you evaluate its performance on the testing set. This gives you a reliable estimate of how well your model will generalize to new, unseen data. It's important that the testing set remains completely separate from the training and validation sets to avoid biasing your evaluation.



Here's a typical workflow:

- **Initial Split:** Start by splitting your dataset into training and testing sets. A common split ratio is 70-80% for training and 20-30% for testing. It can be done manually: Linear train test split, random split or Using **train_test_split** function from **sklearn**.
- **Model Training:** Train your model using the training set. This involves feeding the model input features and the corresponding target values (labels) so that it learns to make predictions.
- **Validation:** Use the validation set to fine-tune the model's hyperparameters (e.g., learning rate, regularization strength) and architecture. This iterative process involves training the model with various combinations of hyperparameters and observing how well it performs on the validation set.
- **Model Evaluation:** After tuning, evaluate your model's performance using the testing set. This provides an unbiased assessment of how well your model generalizes to new, unseen data. Make sure to refrain from making any further changes to your model based on the testing results.
- **Iterative Process:** If the performance on the testing set is not satisfactory, you may need to go back to the training and validation stages, adjusting hyperparameters and model architecture as needed.

It's important to note that in some cases, you might also perform techniques like cross-validation, where you partition your data into multiple folds and use them for training, validation, and testing in a rotating manner. This can provide a more robust estimate of model performance, especially when you have limited data.

Evaluation / Performance Measurement of Models:

I. Regression Metrics/ Goodness of fit of Linear Regression:

The goodness of fit of a Linear Regression (LR) model is typically assessed using various evaluation metrics that quantify how well the model fits the observed data. Here are some commonly used metrics for evaluating the goodness of fit of a Linear Regression model:

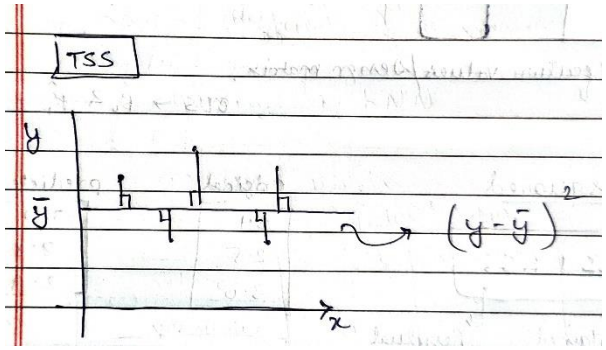
i. R-squared (Coefficient of Determination):

R-squared measures the proportion of the variance in the dependent variable (y) that is explained by the independent variables (features) in the model. It provides an indication of how well the model captures the variability in the data.

$R^2 = [(ESS)/(TSS)] = 1-[(RSS) / (TSS)]$

- **TSS (Total Sum of Squares):**

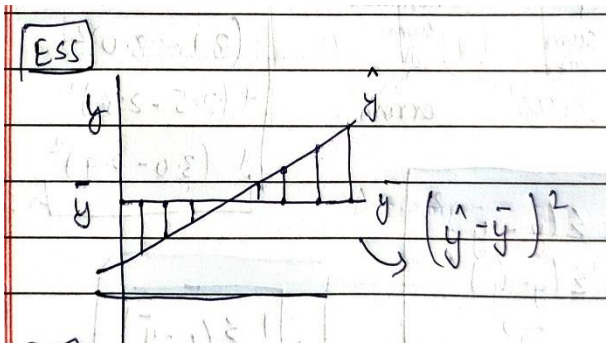
TSS represents the total variability in the dependent variable (y). It measures the total deviation of the actual y values from the mean of y.



$TSS=\sum(y_{actual} - y_{mean})^2$

- **ESS (Explained Sum of Squares):**

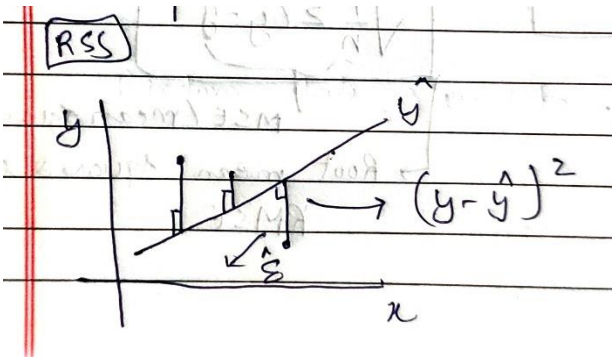
ESS represents the variability in the dependent variable (y) that is explained by the regression model. It measures the deviation of the predicted y values from the mean of y.



$ESS=\sum(y_{fitted} - y_{actual})^2$

- **RSS (Residual Sum of Squares):**

RSS represents the unexplained variability or residual error in the dependent variable (y). It measures the squared differences between the actual y values and the predicted y values.

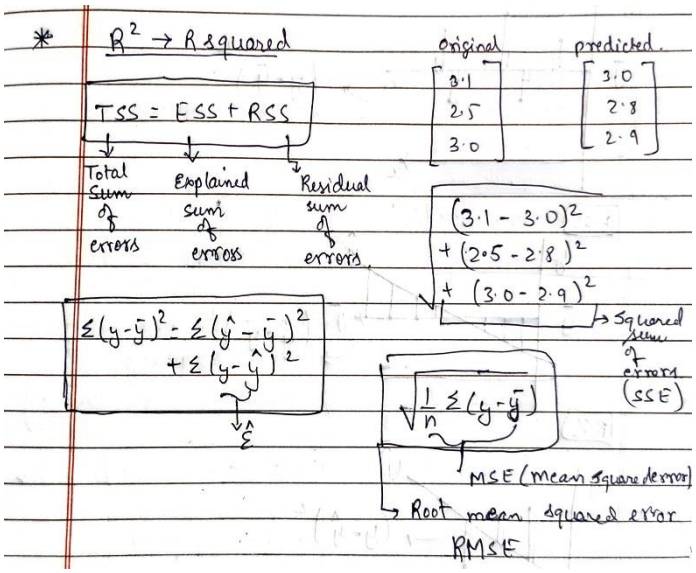


$RSS=\sum(y_{actual} - y_{fitted})$

An R-squared value close to 1 indicates that the model explains a large portion of the variance in the data, while a value close to 0 indicates that the model doesn't explain much of the variance.

ii. Adjusted R-squared:

Adjusted R-squared accounts for the number of predictors in the model. It penalizes the addition of irrelevant predictors that do not contribute significantly to the improvement of the model.

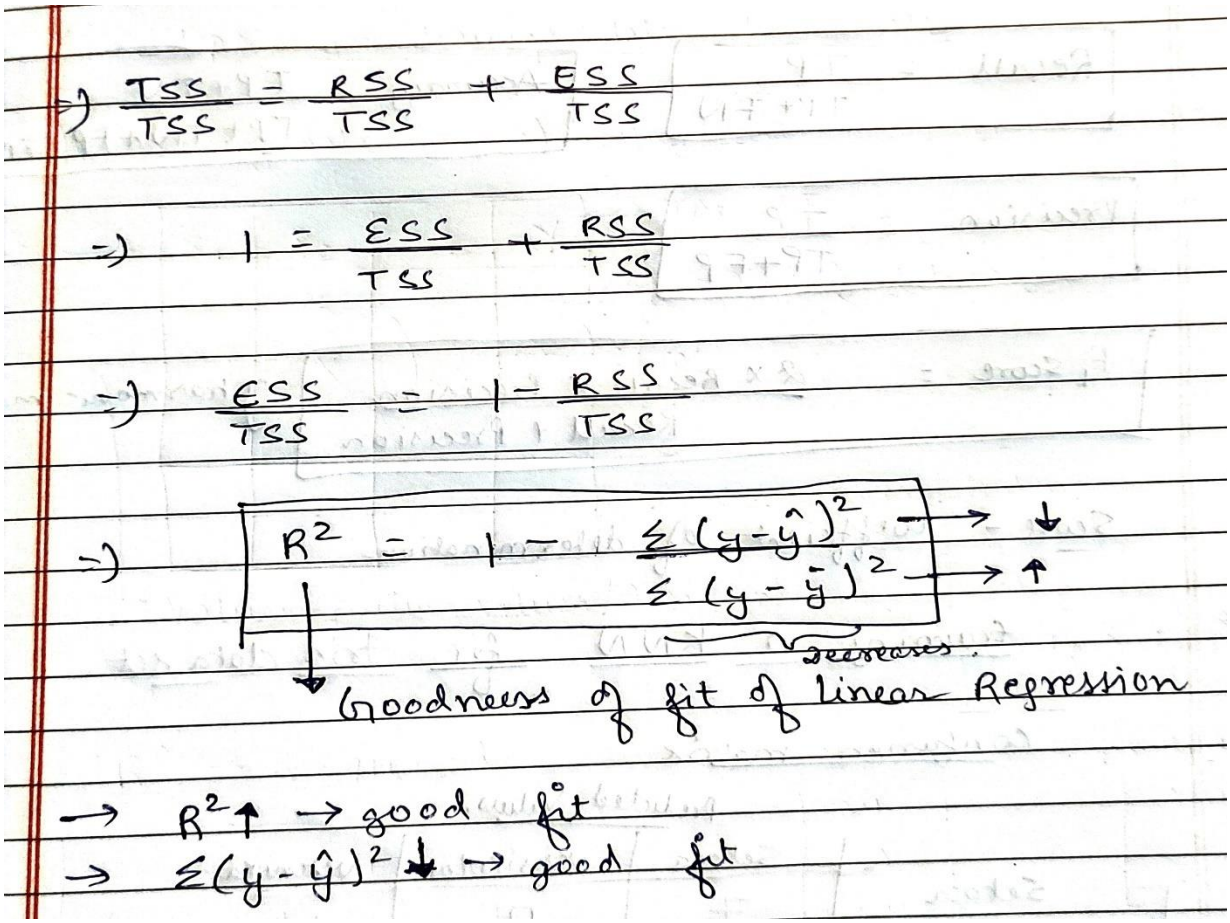


iii. Mean Squared Error (MSE):

MSE measures the average squared difference between the predicted and actual y values. Lower MSE values indicate better model performance.

iv. Root Mean Squared Error (RMSE):

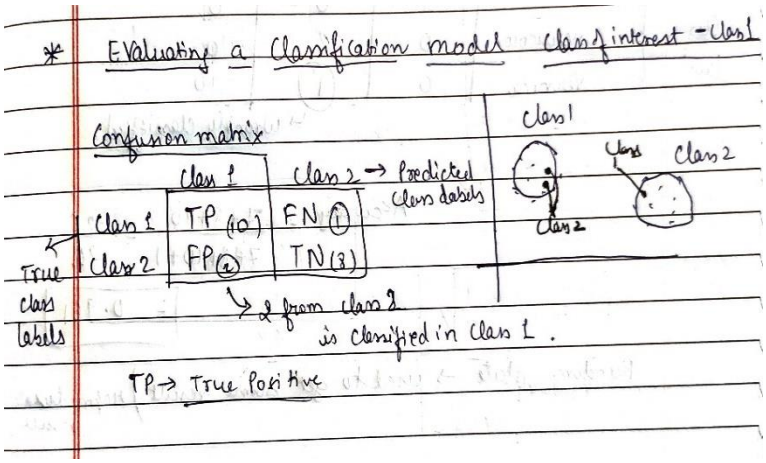
RMSE is the square root of the MSE. It provides a measure of the average magnitude of the prediction errors.



II. Classification Metrics:

- **Confusion Matrix:**

  A matrix showing the counts of true positive, true negative, false positive, and false negative predictions.



- **Accuracy:**

  Measures the proportion of correctly classified instances to the total instances.

- **Precision:**

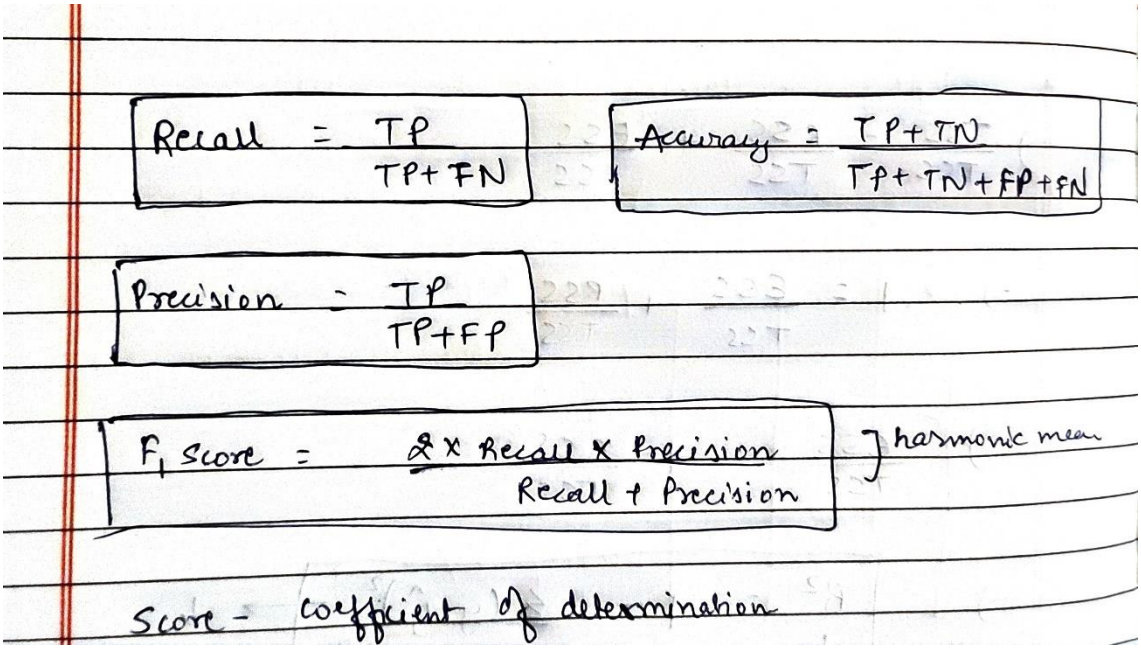
  Measures the proportion of true positive predictions among all positive predictions.

- **Recall (Sensitivity):**

  Measures the proportion of true positive predictions among all actual positive instances.

- **F1-Score:**

  The harmonic mean of precision and recall. It balances precision and recall especially for imbalanced datasets.



3.1.4 LOGISTIC REGRESSION:

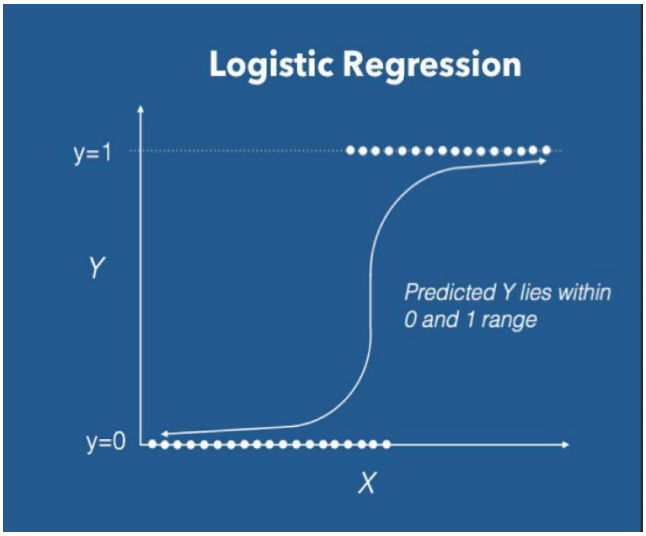
- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1**.
- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems**.
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:

Logistic Function (Sigmoid Function):

The sigmoid function is a mathematical function used to map the predicted values to probabilities.

It maps any real value into another value within a range of 0 and 1. o The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the “S” form.

The S-form curve is called the Sigmoid function or the logistic function.



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

In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

As shown above, the figure sigmoid function converts the continuous variable data into the probability i.e. between 0 and 1.

$\sigma(z)$ tends towards 1 as $z \to \infty$

$\sigma(z)$ tends towards 0 as $z \to -\infty$

$\sigma(z)$ is always bounded between 0 and 1

where the probability of being a class can be measured as:

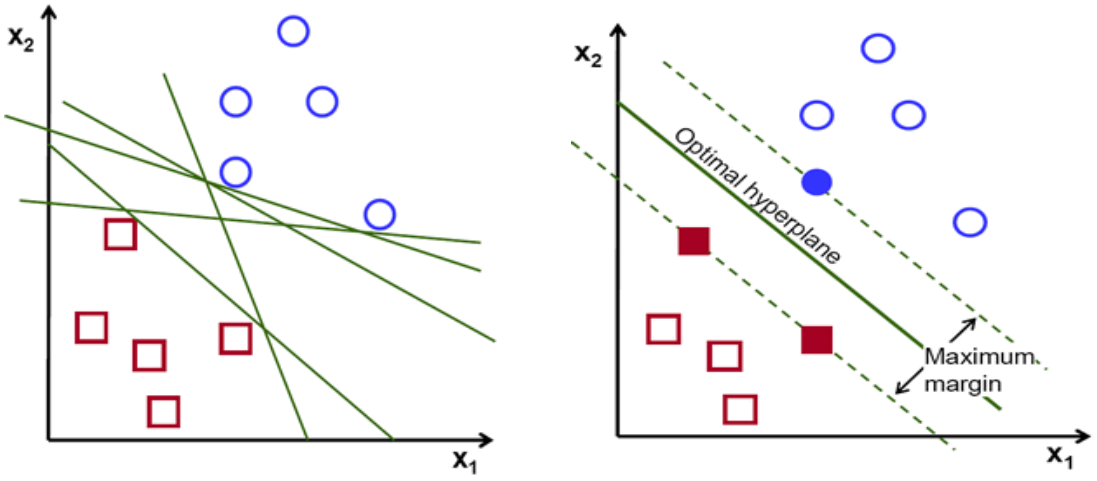
$$P(y=1) = \sigma(z)$$
$$P(y=0) = 1 - \sigma(z)$$

3.1.5 SUPPORT VECTOR MACHINE(SVM):

Support vector machine is highly preferred by many as it produces significant accuracy with less computation power. Support Vector Machine, abbreviated as SVM can be used for both regression and classification tasks. But, it is widely used in classification objectives.

What is Support Vector Machine?

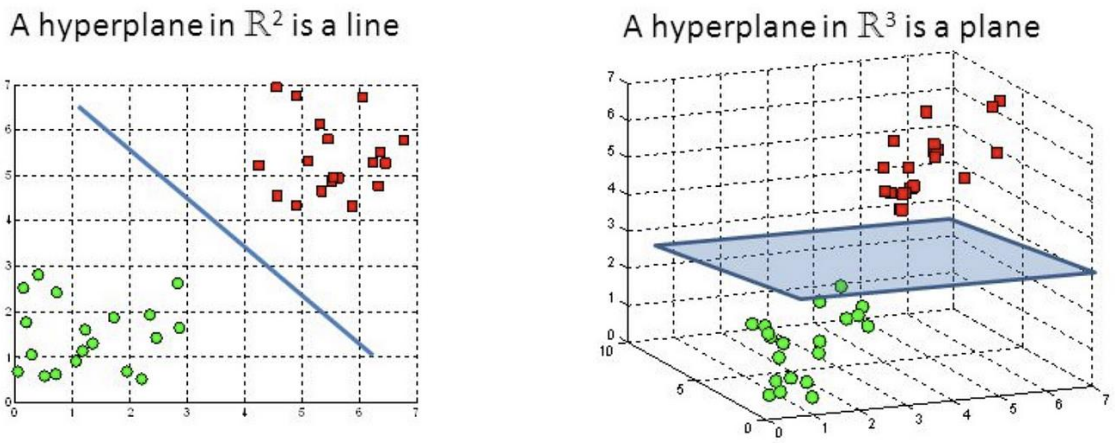
The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.



POSSIBLE HYPERPLANES

To separate the two classes of data points, there are many possible hyperplanes that could be chosen. Our objective is to find a plane that has the maximum margin, i.e the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.

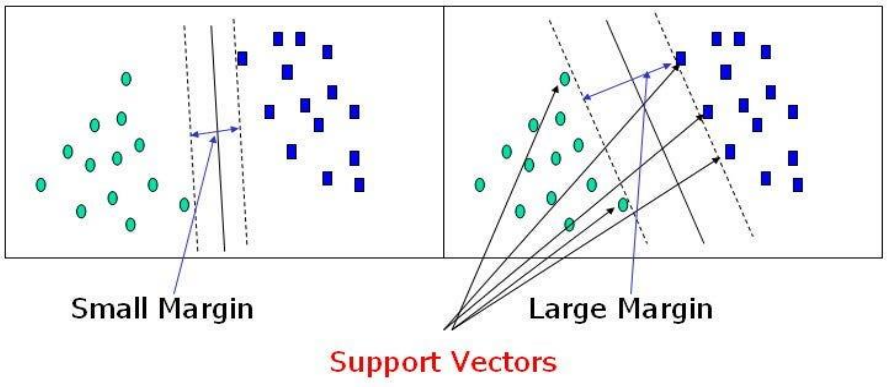
3.1.5.1 Hyperplanes and Support Vectors



Hyperplanes in 2D and 3D feature space

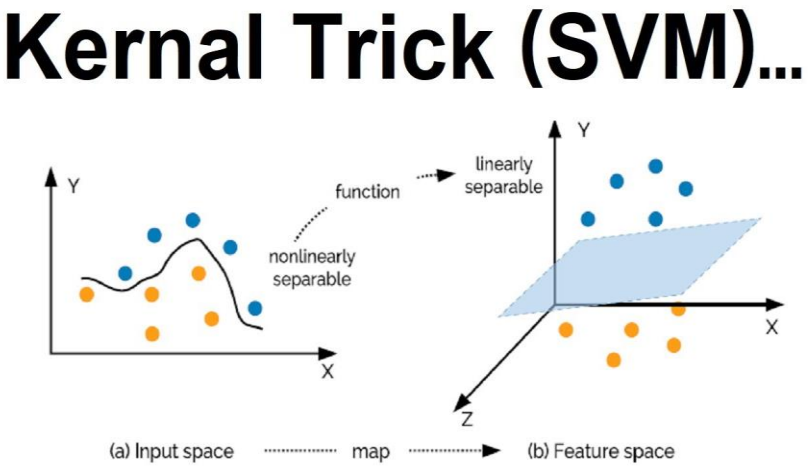
Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes. Also, the dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3.

Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the



margin of the classifier. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM.

3.1.5.2 KERNAL FUNCTION:



It is simply the mathematical function to transform data into higher dimensions.

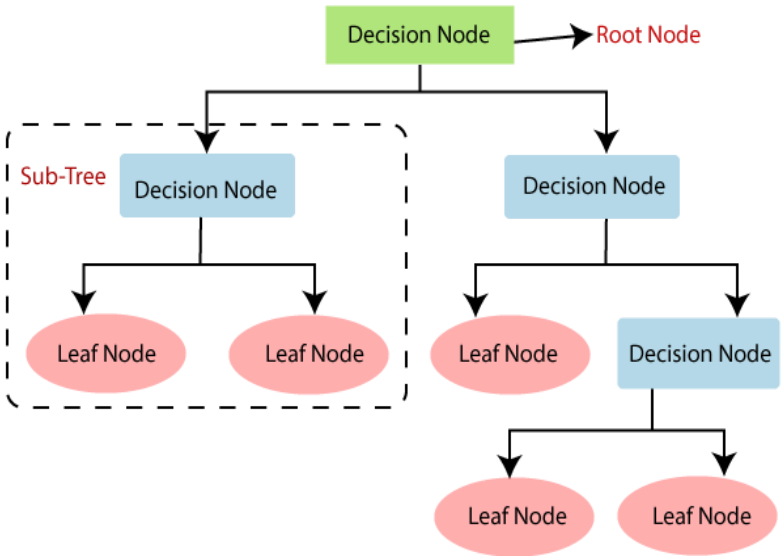
In kernel methods, such as Support Vector Machines (SVMs), the idea is to implicitly map the data into a higher-dimensional space using a kernel function. The data may not be explicitly transformed, but the algorithm operates in a higher-dimensional feature space. Popular kernel functions include the polynomial kernel, radial basis function (RBF) kernel, and sigmoid kernel.

3.1.6 DECISION TREES:

- Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**
- In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- ***It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.***

- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.**
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
- Below diagram explains the general structure of a decision tree:

Note: A decision tree can contain categorical data (YES/NO) as well as numeric data.



Why use Decision Trees?

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- The logic behind the decision tree can be easily understood because it shows a tree-like structure.

Decision Tree Terminologies

 Root Node: Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.

 Leaf Node: Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.

 Splitting: Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.

 Branch/Sub Tree: A tree formed by splitting the tree.

 Pruning: Pruning is the process of removing the unwanted branches from the tree.

 Parent/Child node: The root node of the tree is called the parent node, and other nodes are called the child nodes.

How does the Decision Tree algorithm Work?

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

- **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
- **Step-2:** Find the best attribute in the dataset using **Attribute Selection Measure (ASM).**
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
- **Step-4:** Generate the decision tree node, which contains the best attribute.
- **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Attribute Selection Measures

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as **Attribute selection measure or ASM.** By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

- **Information Gain**
- **Gini Index**

1. Information Gain:

- Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
- It calculates how much information a feature provides us about a class.
- According to the value of information gain, we split the node and build the decision tree.
- A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

1. **Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each feature)**

Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)

Where,

i. **S= Total number of samples**
ii. **P(yes)= probability of yes**
iii. **P(no)= probability of no**

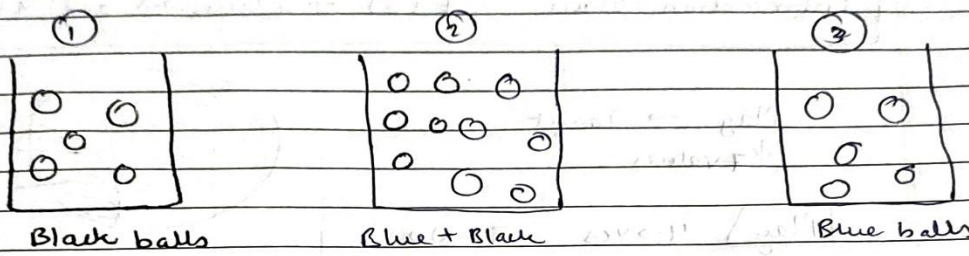
2. Gini Index:

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.
- An attribute with the low Gini index should be preferred as compared to the high Gini index.
- It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
- Gini index can be calculated using the below formula:

Gini Index= 1- $\sum_j P_j^2$

Date / 20

* DESICION TREES

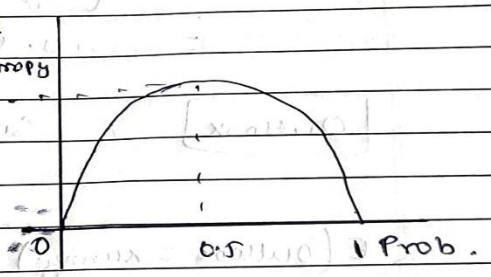


Probab. of black in ① = 1

P (black) in ② = 0.5

P(black) in ③ = 0

p → prob. of black
1−p → prob. of blue.



$$E(S) = -\sum p \cdot \log_2(p)$$

$E(S, p=0) = 0$ ✓

$E(S, p=1) = -[1 \cdot \log_2(1) + 0 \cdot \log 0] = 0$ ✓

$E(S, p=0.5) = -[0.5 \log(0.5) + 0.5 \log(0.5)]$

$= -0.5 [\log_2(\frac{1}{2}) + \log(1/2)]$

$= -0.5 [\log_2(1) - \log_2(2) + \log_2(1) + \log_2(2)]$

$= -0.5 (0-1+0-1) = -0.5 \times -2$

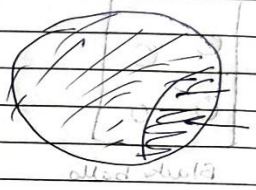
$E(S, p=0.5) = 1$ ✓

Outlook	Temp	Humidity	Windy	Play
sunny	Hot	High	FALSE	No
sunny	Hot	High	TRUE	No
overcast	Hot	High	FALSE	Yes
Rainy	Mild	High	FALSE	Yes
Rainy	Cool	Normal	FALSE	Yes
Rainy	Cool	Normal	TRUE	No
overcast	Cool	Normal	TRUE	Yes

IG

Information Gain = E(S) − [weighted avg × E(each feature)]

Play → Target
↓ 7 values



Play 4 → Yes 3 - No.

$$E(Play) = -\left[p(No)\cdot\log_2(p(No)) + p(Yes)\cdot\log_2(p(Yes))\right]$$

$$= -\left[\frac{3}{7}\cdot\log_2\left(\frac{3}{7}\right) + \frac{4}{7}\log_2\left(\frac{4}{7}\right)\right]$$

$$= 0.5240 + 0.4617 = 0.98$$

Outlook Sunny → 2 (↓ No) Overcast = 2 (↓ Yes) Rainy = 3 (↓ 2-Yes 1-No.)

$$E(Outlook = sunny) = -\left[0\log_2 0 + 1\log_2(1)\right]$$

$$= 0$$

$$E(Outlook = Overcast) = -\left[1\log_2(1) + 0\log_2(0)\right]$$

$$= 0$$

$$E(Outlook = Rainy) = -\left[\frac{2}{3}\log_2\left(\frac{2}{3}\right) + \frac{1}{3}\log_2\left(\frac{1}{3}\right)\right]$$

$$= 0.90$$

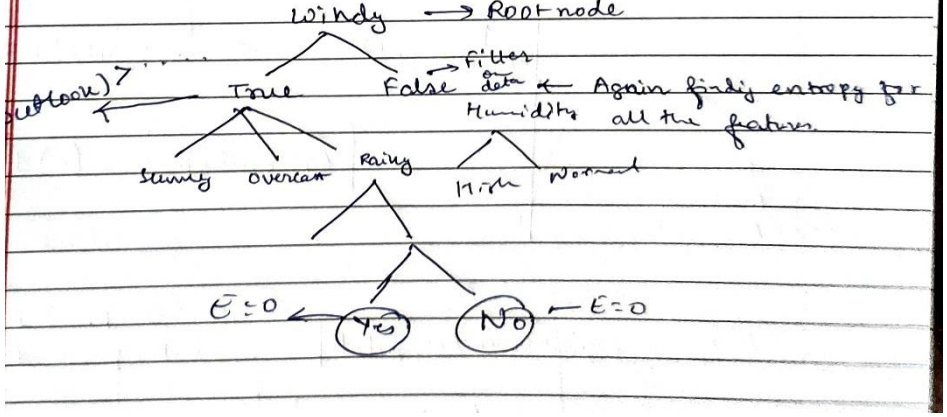
$$IG(Outlook) = E(S) - \frac{3}{7}E(Outlook = rainy)$$

$$= 0.98 - \frac{3}{7}\times 0.90$$

$$IG(Outlook) = 0.59$$

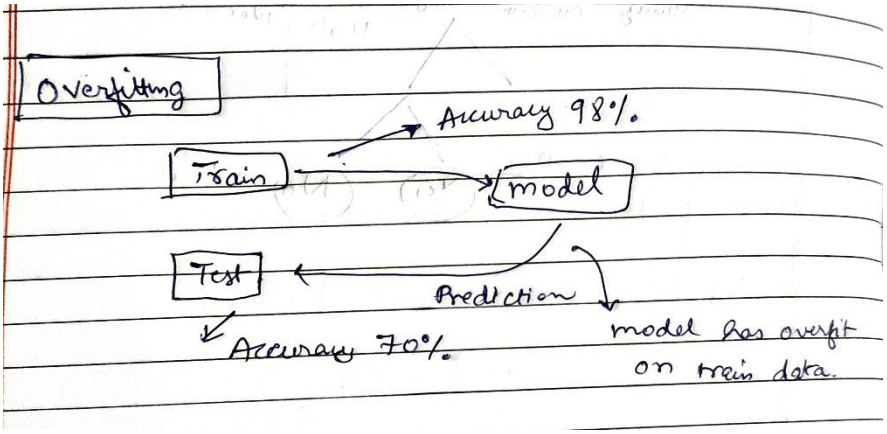
IG(Outlook) # IG(Windy) > IG(Humidity) > IG(Temp)

Greatest IG → 1st split



- **OVERFITTING:**

Overfitting is a common problem in machine learning where a model learns the training data too well and captures noise or random fluctuations in the data rather than the underlying pattern. This results in a model that performs very well on the training data but performs poorly on unseen or new data. In other words, the model has memorized the training data rather than generalizing from it.



CHAPTER- 04

ENSEMBLE MODELS:

An ensemble model is a machine learning technique that involves combining the predictions of multiple models to improve the accuracy and robustness of the prediction. Ensemble models work on the principle that a group of weak learners can work together to create a strong learner that is more accurate than any of the individual learners.

How Does Ensemble Model Work?

The goal of using ensemble methods is to improve the skill of predictions over that of any of the contributing members.

This objective is straightforward but it is less clear how exactly ensemble methods are able to achieve this.

It is important to develop an intuition for how ensemble techniques work as it will help you both choose and configure specific ensemble methods for a prediction task and interpret their results to come up with alternative ways to further improve performance.

These elements are how and ensemble methods work in the general sense, namely:

1. **Members learn different mapping functions for the same problem**. This is to ensure that models make different prediction errors.
2. **Predictions made by members are combined in some way**. This is to ensure that the differences in prediction errors are exploited.

We don't simply smooth out the prediction errors, although we can; instead, we smooth out the mapping function learned by the contributing members.

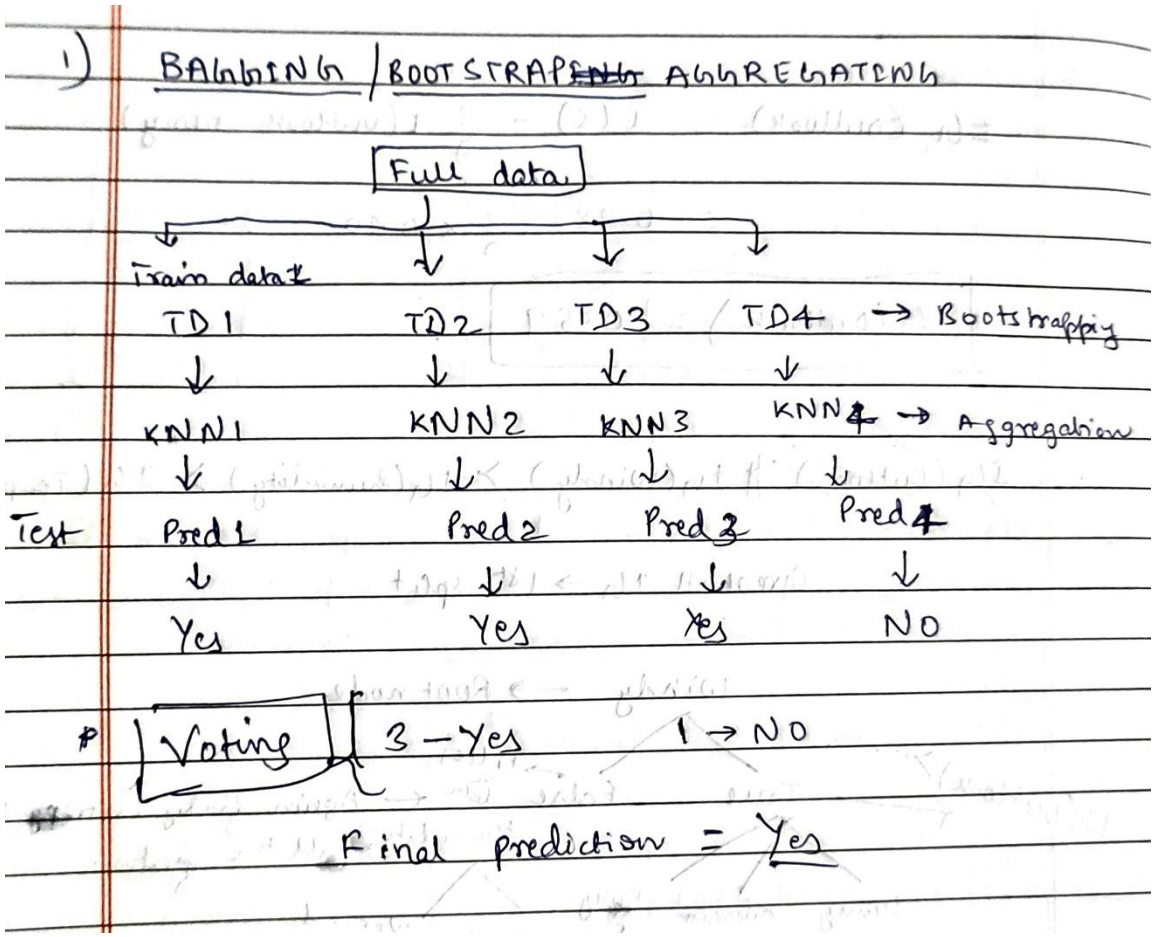
The improved mapping function allows better predictions to be made.

This is a deeper point and it is important that we understand it. Let's take a closer look at what it means for both classification and regression tasks.

Types of Ensemble Models

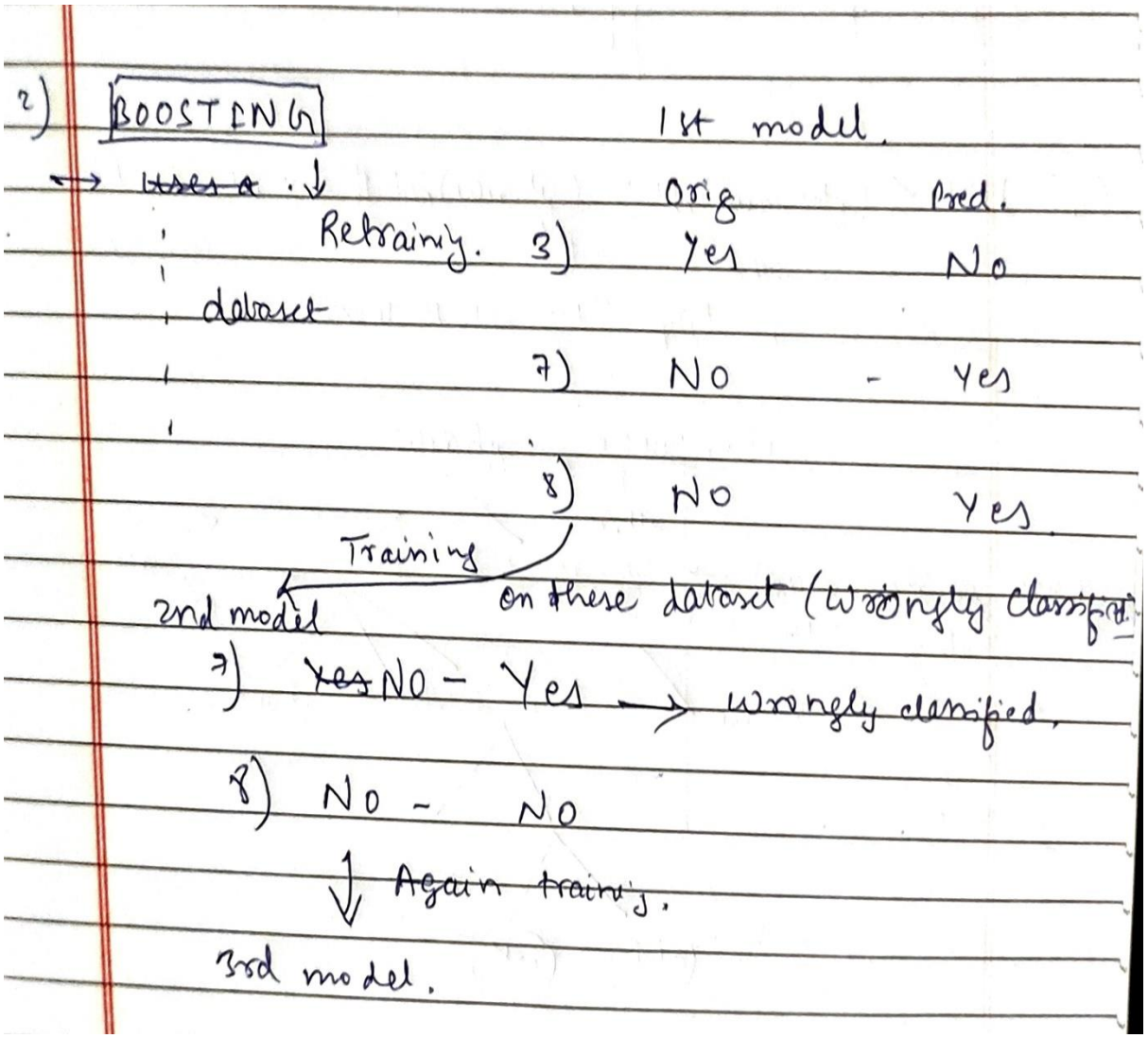
There are several types of ensemble models, including:

i. **Bagging:** Bagging, also known as Bootstrap aggregating, is an ensemble learning technique that helps to improve the performance and accuracy of machine learning algorithms. It is used to deal with bias-variance trade-offs and reduces the variance of a prediction model. Bagging avoids



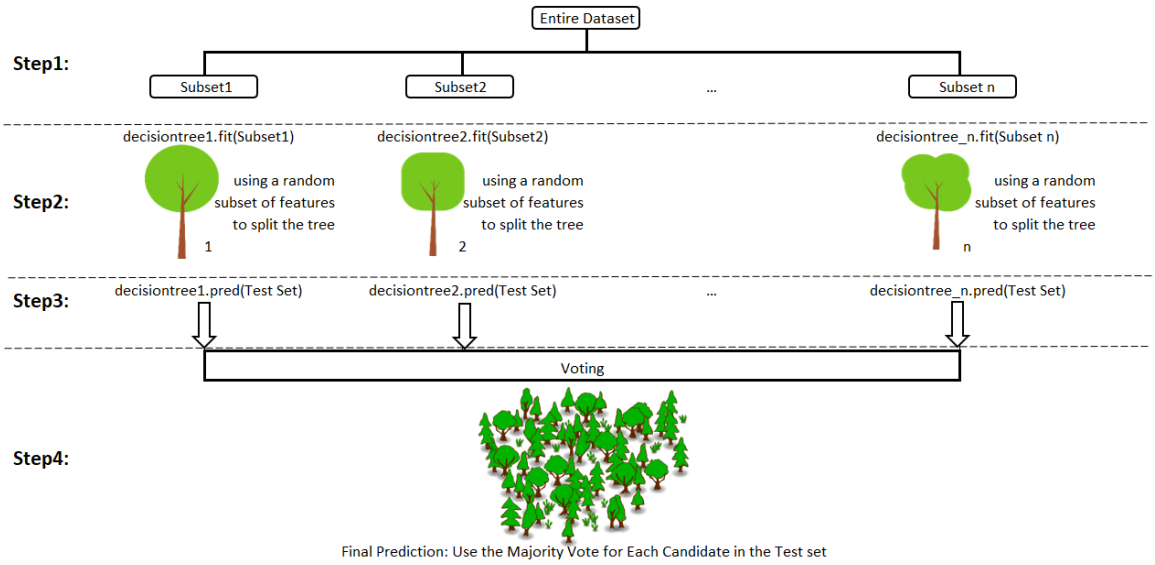
overfitting of data and is used for both regression and classification models, specifically for decision tree algorithms.

ii. **Boosting:** Uses a number of weak classifier to build a model. Retraining data again and again. Training a bunch of individual models in a sequential way. Each individual model learns from mistakes made by the previous model.



iii. **Random forest :** is an ensemble model using bagging as the ensemble method and decision tree as the individual model.

Let's take a closer look at **the magic of the randomness**:



Step 1: **Select n (e.g. 1000) random subsets** from the training set

Step 2: **Train n (e.g. 1000) decision trees**

- one random subset is used to train one decision tree
- the optimal splits for each decision tree are based on a random subset of features (e.g. 10 features in total, randomly select 5 out of 10 features to split)

Step 3: **Each individual tree predicts** the records/candidates in the test set, independently.

Step 4: **Make the final prediction**

For each candidate in the test set, Random Forest uses the class (e.g. cat or dog) with the **majority vote** as this candidate's final prediction.

Of course, our 1000 trees are the parliament here.

CHAPTER- 05

UNSUPERVISED MACHINE LEARNING

As the name suggests, unsupervised learning is a machine learning technique in which models are not supervised using training dataset. Instead, models itself find the hidden patterns and insights from the given data. It can be compared to learning which takes place in the human brain while learning new things. It can be defined as:

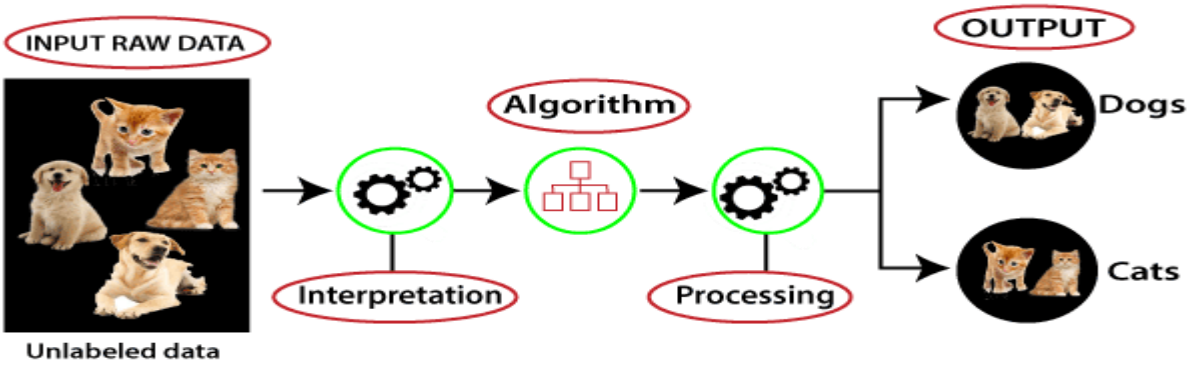
Unsupervised learning is a type of machine learning in which models are trained using unlabeled dataset and are allowed to act on that data without any supervision.

Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to **find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format**.

Why use Unsupervised Learning?

Below are some main reasons which describe the importance of Unsupervised Learning:

- Unsupervised learning is helpful for finding useful insights from the data.
- Unsupervised learning is much similar as a human learns to think by their own experiences, which makes it closer to the real AI.
- Unsupervised learning works on unlabeled and uncategorized data which make unsupervised learning more important.
- In real-world, we do not always have input data with the corresponding output so to solve such cases, we need unsupervised learning.



Working of Unsupervised Learning

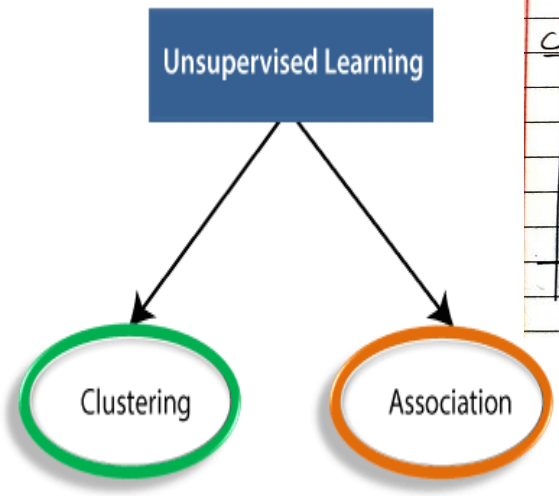
Working of unsupervised learning can be understood by the below diagram

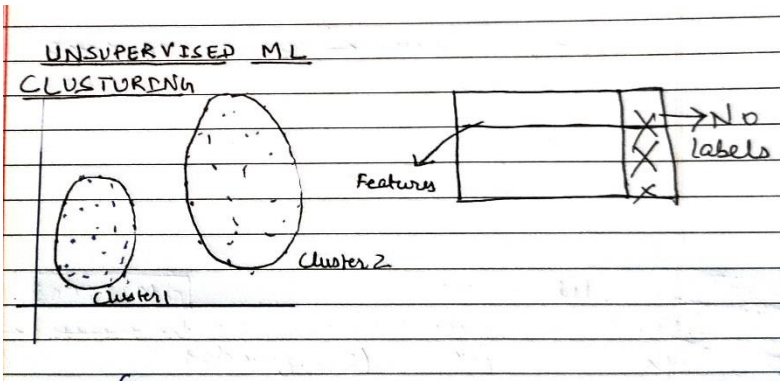
Here, we have taken an unlabeled input data, which means it is not categorized and corresponding outputs are also not given. Now, this unlabeled input data is fed to the machine learning model in order to train it. Firstly, it will interpret the raw data to find the hidden patterns from the data and then will apply suitable algorithms such as k-means clustering, Decision tree, etc.

Once it applies the suitable algorithm, the algorithm divides the data objects into groups according to the similarities and difference between the objects.

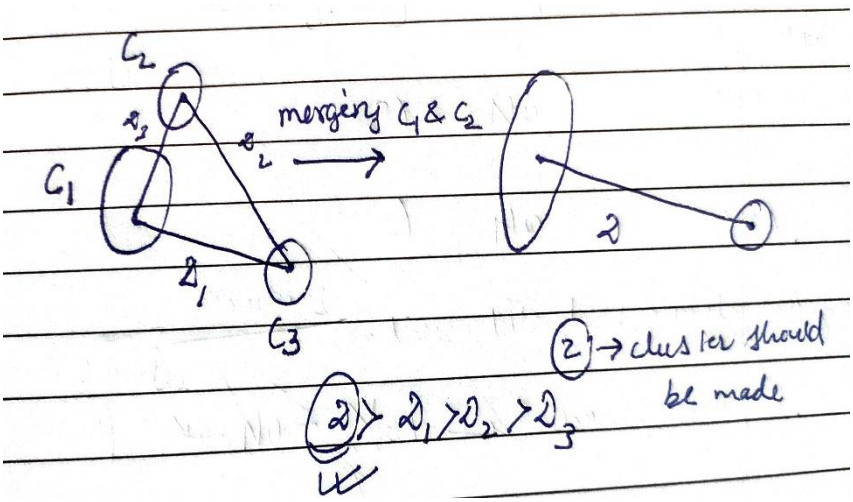
Types of Unsupervised Learning Algorithm:

The unsupervised learning algorithm can be further categorized into two types of problems:





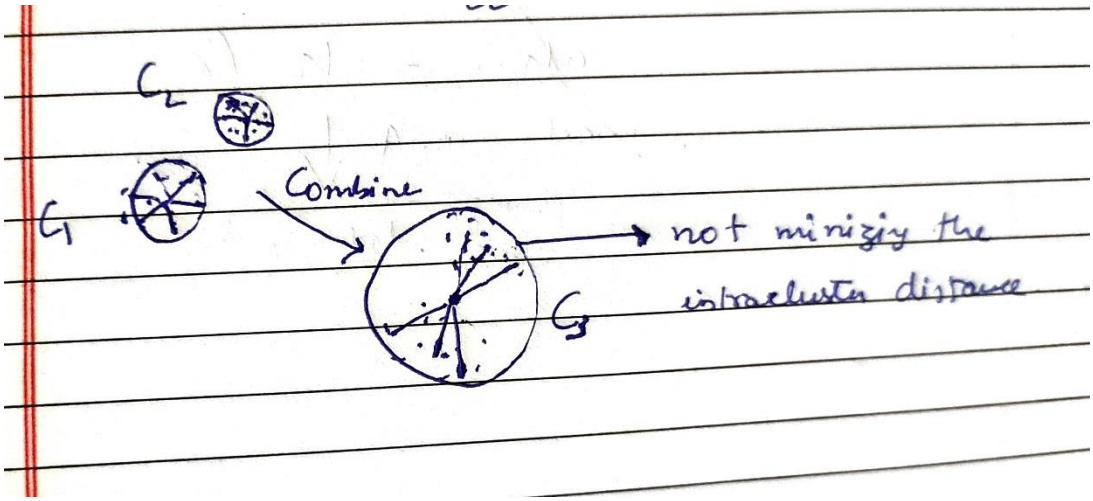
1. **Clustering**: Clustering is a method of grouping the objects into clusters such that objects with most similarities remains into a group and has less or no similarities with the objects of another group. Cluster analysis finds the commonalities between the data objects and categorizes them as per the presence and absence of those commonalities.



How to decide that how many clusters should we have:

- **Data Context**

- **Maximizing the inter-cluster distance:**
- **Minimizing the intra-cluster distance:**distance within the cluster



2. **Association**: An association rule is an unsupervised learning method which is used for finding the relationships between variables in the large database. It determines the set of items that occurs together in the dataset. Association rule makes marketing strategy more effective. Such as people who buy X item (suppose a bread) are also tend to purchase Y (Butter/Jam) item. A typical example of Association rule is Market Basket Analysis

Unsupervised Learning algorithms:

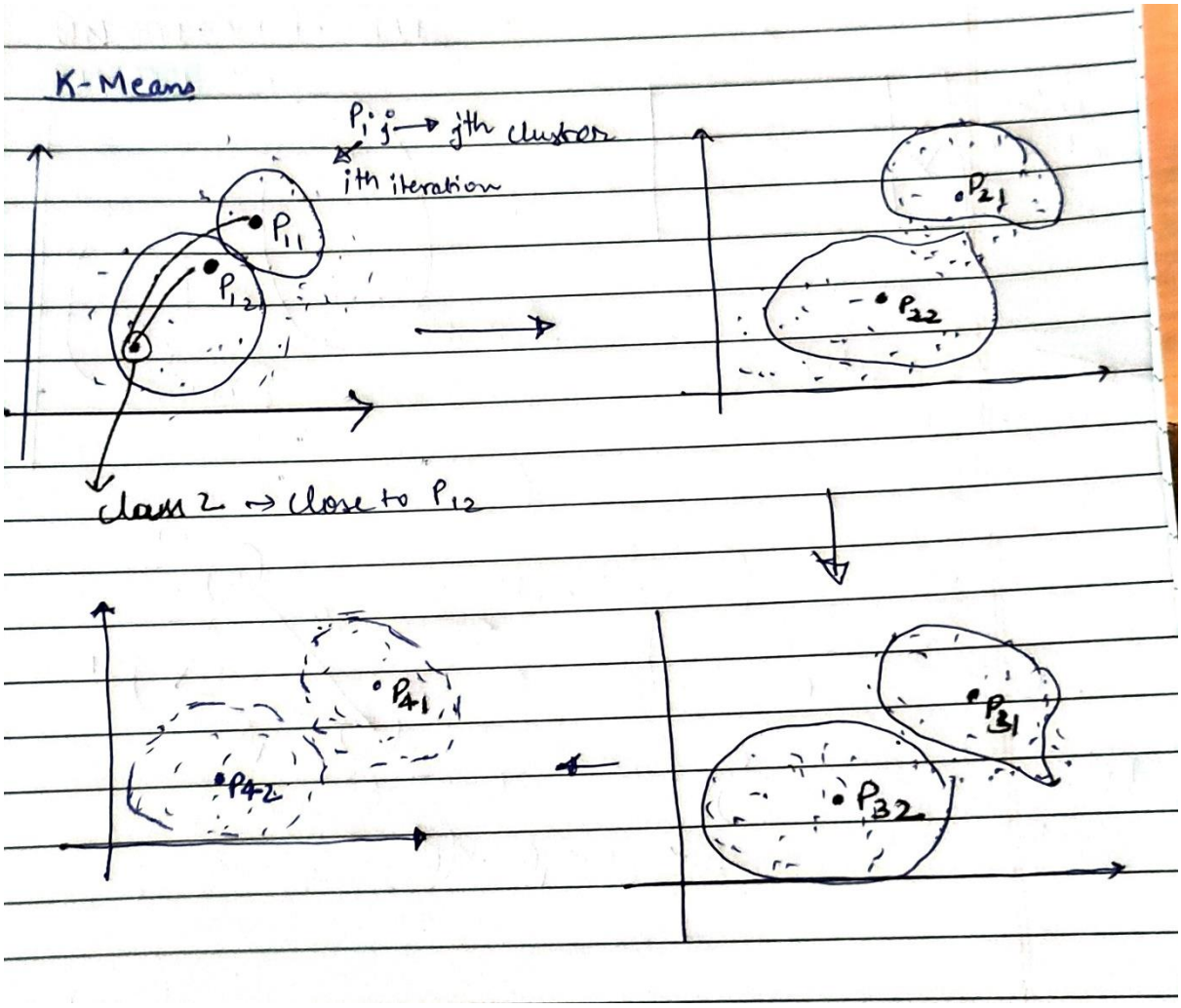
Below is the list of some popular unsupervised learning algorithms:

- **K-means clustering**

K-Means is a popular machine learning clustering algorithm used for partitioning a dataset into a set of distinct, non-overlapping groups or clusters. These clusters are formed by grouping similar data points together based on their feature similarities.

Steps:

a. Decide k=Number of clusters to be formed
b. Initialize k no. of points as the mean of the k-clusters(randomly).
c. Club points which are close to P_{11} as Class 1 and Club points close to P_{12} as Class 2.
d. After the 2 classes are created , re-calculate the 2 mean points.
e. Repeat steps c to e.



Advantages of Unsupervised Learning

- Unsupervised learning is used for more complex tasks as compared to supervised learning because, in unsupervised learning, we don't have labeled input data.
- Unsupervised learning is preferable as it is easy to get unlabeled data in comparison to labeled data.

Disadvantages of Unsupervised Learning

- Unsupervised learning is intrinsically more difficult than supervised learning as it does not have corresponding output.
- The result of the unsupervised learning algorithm might be less accurate as input data is not labeled, and algorithms do not know the exact output in advance.

CHAPTER-06

FINAL PROJECT

Python Implemention of different machine learning models for training testing and predictions:

Link for Jupyter Notebook: Final Project

Conclusion

In conclusion, this comprehensive report has provided an immersive journey through the intricate landscape of data science using Python. By seamlessly transitioning from fundamental Python concepts to advanced data analysis and machine learning techniques, this report has armed both novices and seasoned practitioners with a powerful arsenal of skills and tools.

The foundational understanding of Python programming, covered at the outset, serves as the bedrock upon which the rest of the report is built. Concepts such as variables, loops, functions, and data structures lay the groundwork for the more sophisticated data manipulation and analysis techniques explored later.

The journey into data analysis using NumPy and Pandas underscores the transformative power of these libraries in simplifying complex data operations. By leveraging multi-dimensional arrays and DataFrames, data cleaning, transformation, and aggregation become streamlined processes, freeing data scientists to focus on extracting meaningful insights.

Data visualization, facilitated by Plotly and Seaborn, emerges as a cornerstone of effective communication in data science. The ability to craft dynamic, interactive, and aesthetically pleasing visualizations elevates data-driven storytelling, making it easier to convey complex patterns and trends to a wide audience.

The report delves into the critical step of Exploratory Data Analysis (EDA), emphasizing its role in uncovering hidden insights and guiding subsequent analyses. EDA fosters hypothesis generation, outlier detection, and a deeper understanding of data distributions, paving the way for informed decision-making.

The comprehensive exploration of machine learning algorithms—K-Nearest Neighbors, Linear Regression, Logistic Regression, Support Vector Machines, and Decision Trees—provides a well-rounded understanding of their theoretical underpinnings and practical

implementations. This knowledge equips data scientists to select the appropriate algorithm for a given problem, train predictive models, and fine-tune parameters for optimal performance.

Model evaluation and metrics, exemplified through the Confusion Matrix, offer a tangible approach to quantifying the effectiveness of classification algorithms. By decoding the confusion matrix, data scientists can gain insights into the trade-offs between accuracy, precision, recall, and F1-score, enabling informed decisions in model selection and refinement.

The exploration of ensemble learning techniques—Bagging, Boosting, Voting, and Random Forest—sheds light on the power of combining models to achieve superior performance. Detailed implementations in Jupyter Notebook showcase the step-by-step process of creating and evaluating ensemble models, emphasizing practicality alongside theory.

The culmination of the report with K-Means clustering illustrates how unsupervised learning can unearth underlying structures in data. By employing Jupyter Notebook to demonstrate K-Means implementation, the report bridges the gap between theory and hands-on practice, enabling readers to grasp the mechanics of clustering algorithms.

In essence, this report stands as a comprehensive guide that empowers learners to embark on a rewarding journey through data science. By merging theoretical foundations with hands-on implementations using Jupyter Notebook, the report equips individuals to seamlessly transition from raw data to actionable insights. Armed with these skills, aspiring data scientists are poised to navigate the ever-evolving landscape of data science with confidence and innovation.

FUTURE PERSPECTIVES

Looking ahead, the field of data science is poised for exciting advancements and transformations. As technology continues to evolve and new challenges emerge, data scientists can anticipate several key trends and opportunities in the future:

1) **Exponential Growth of Data:** The volume of data generated worldwide is growing at an unprecedented rate. This data deluge presents both challenges and opportunities for data scientists, requiring innovative approaches to data storage, processing, and analysis.
2) **AI and Automation:** Artificial Intelligence (AI) and machine learning will play an increasingly integral role in automating routine data tasks, enabling data scientists to

focus on higher-level analysis and decision-making. AutoML (Automated Machine Learning) tools will become more sophisticated, simplifying model selection, hyperparameter tuning, and feature engineering.

3) **Ethics and Responsible AI:** With the increased reliance on AI-driven decision-making, ethical considerations will gain prominence. Data scientists will need to prioritize fairness, transparency, and accountability in their models and algorithms, addressing biases and potential negative impacts.
4) **Interdisciplinary Collaboration:** Data science will continue to intersect with various disciplines, including domain-specific knowledge, business acumen, and social sciences. Collaborations between data scientists and experts from other fields will be essential to address complex, real-world challenges.
5) **Big Data and Cloud Computing:** Cloud platforms will remain crucial for handling massive datasets and resource-intensive computations. Skills in deploying and managing data pipelines on cloud infrastructure will be valuable for efficient data processing.
6) **Edge Computing:** As IoT devices proliferate, data analysis at the edge (closer to where data is generated) will become more significant. Data scientists will need to adapt algorithms and models to work effectively on resource-constrained devices.
7) **Advanced Machine Learning Techniques**: Deep Learning and neural networks will continue to evolve, enabling more sophisticated analysis of unstructured data like images, audio, and text. Transfer learning and generative models will push the boundaries of what's possible.
8) **Data Privacy and Security:** With growing concerns about data privacy, data scientists will need to develop expertise in secure data handling, anonymization techniques, and compliance with data protection regulations.
9) **Explainable AI:** The need for interpretable and explainable AI models will become critical, especially in industries where decisions impact lives, such as healthcare and finance.
10) **Continuous Learning:** Given the rapidly evolving nature of technology, data scientists will need to embrace lifelong learning. Staying updated with the latest tools, techniques, and trends will be essential for remaining competitive in the field.
11) **Human-Centered Design:** Data science solutions will need to be user-centric, focusing not only on accurate predictions but also on creating actionable insights that resonate with stakeholders.

12) **Personalization and Recommendation Systems**: As businesses strive to provide personalized experiences, data scientists will play a key role in developing recommendation systems that enhance user engagement and satisfaction.

In essence, the future of data science is characterized by an ever-expanding landscape of possibilities. Data scientists who combine technical expertise with adaptability, ethical considerations, and domain knowledge will be well-positioned to drive innovation, solve complex problems, and contribute to meaningful advancements in various industries.

REFERENCES

1.Python Basics:

Official Python Documentation: https://docs.python.org/3/tutorial/

Python for Data Science Handbook by Jake VanderPlas: https://jakevdp.github.io/PythonDataScienceHandbook/

2.Data Analysis with NumPy and Pandas:

NumPy Documentation: https://numpy.org/doc/stable/

Pandas Documentation: https://pandas.pydata.org/docs/

Pandas Cookbook: https://pandas.pydata.org/pandas-docs/stable/user_guide/cookbook.html

3.Data Visualization with Plotly and Seaborn:

Plotly Express Documentation: https://plotly.com/python/plotly-express/

Seaborn Documentation: https://seaborn.pydata.org/tutorial.html

Interactive Data Visualization with Python (book): https://plotly.com/python/plotly-fundamentals/

4.Exploratory Data Analysis (EDA):

Exploratory Data Analysis with Python (book): https://www.oreilly.com/library/view/exploratory-data-analysis/9781491952931/

Practical EDA in Python: https://towardsdatascience.com/a-practical-introduction-to-exploratory-data-analysis-95f1d3c2f0bd

5.Machine Learning Algorithms:

Scikit-Learn Documentation: https://scikit-learn.org/stable/documentation.html

Introduction to Machine Learning with Python (book): https://www.oreilly.com/library/view/introduction-to-machine/9781449369880/

Machine Learning Mastery: https://machinelearningmastery.com/start-here/#algorithms

6.Model Evaluation and Metrics:

Scikit-Learn Metrics Documentation: https://scikit-learn.org/stable/modules/model_evaluation.html

Understanding Confusion Matrix: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

7.Ensemble Learning:

Ensemble Methods Documentation: https://scikit-learn.org/stable/modules/ensemble.html

Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (book): https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/

Ensemble Learning from Scratch: https://towardsdatascience.com/ensemble-learning-from-scratch-8ce166f9f84e

8.K-Means Clustering:

K-Means Clustering with Scikit-Learn: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

Introduction to K-Means Clustering: https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68