STOCK MARKET PREDICTION

Project Review Report Submitted
In Partial Fulfilment of the Requirements

For the Degree of

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

Submitted By

MOHD SABEEL UDDIN (1604-19-737-111)

SYED ARBAZ (1604-19-737-101)

AMENA SARAH (1609-19-737-093)

INFORMATION TECHNOLOGY DEPARTMENT
MUFFAKHAM JAH COLLEGE OF ENGINEERING & TECHNOLOGY
(Affiliated to Osmania University)
Mount Pleasant, 8-2-249, Road No. 3, Banjara Hills, Hyderabad-34
2022-2023

CERTIFICATE

It is certified that the work contained in the project report titled "stock market prediction." by

MOHD SABEEL UDDIN	**(1604-19-737-111)**
SYED ARBAZ	**(1604-19-737-101)**
AMENA SARAH	**(1609-19-737-093)**

has been carried out under my/our supervision and that this work has not been submitted elsewhere for a degree"

Project Guide

Dr.GOURI R. PATIL
Information Technology Dept.
MJ College of Engineering & Tech.
Hyderabad – 500034.

Signature of Head

Dr.MOUSMI AJAY CHAURASIA
Information Technology Dept.
MJ College of Engineering & Tech.
Hyderabad – 500034

STOCK MARKET PREDICTION

Graph-based methodologies are changing the analysis of a variety of real-world systems, including the stock market.

Individual stocks and stock market indices are linked, and when the stock market is shown as a graph, intriguing patterns emerge. In recent years, researchers have been adopting graph-based ways to analyse the stock market, and there is a demand for this information to examine their works from many angles We look at previous graph-based works from five different angles: I Formulation of stock market graphs, (ii) stock market graph filtering, (iii) stock market graph clustering, and (iv) stock movement portfolio optimization, and (v) forecasting. This research includes a succinct summary of the most important methodologies and algorithms. graph-based methods to the stock market are relevant.

CONTENTS

1. INTRODUCTION

1.1 Introduction

In present days so many people are interested in investing money in stock market for earning more in short period of time. Here, In stock market consist of many number of company shares along with prices in stock market every minute stock price will changes depending on the company environment and country economic structure decisions. In stock market there are many brokers for handing the stocks buying and selling between the clients and company. In previous year's there is difficult to predict the stock market because lack of technology and knowledge but in present days technology will increases day by day for that we can predict the stock market easily when compare to past here we can predict stock price by analyzing the previous data by using machine learning techniques. From these techniques we can use neural network it's means that it is interconnected with networking it look like human neural brain structure and simple moving average method.

A graph is a collection of two pieces of data: a group of nodes and a set of edges connecting them Biological structures and social networks, for example, are examples of real-life systems.Financial systems and communication networks are both susceptible to cyber-attacks.in the form of graphs To create and analyse graphs, researchers employ a variety of tools, techniques, and algorithms. Those are the tools, techniques, and algorithms that are considered methods based on graphs A financial system that can be visualised as a graph is the stock market. The implementation of the use of graph-based techniques to stock market analysis is fast growing.

1.2. Technologies used

Stock market prediction seems a complex problem because there are many factors that have yet tobe assumptions.Machine learning as such has many models but this paper focuses on two most important of them and made the predictions using them.

1.3. Objective

Stock market analysis enables investors to identify the intrinsic worth of a security even before investing in it. All stock market tips are formulated after thorough research by experts. Stock analysts try to find out activity of an instrument/sector/market in future.

By using stock analysis, investors and traders arrive at equity buying and selling decisions. Studying and evaluating past and current data helps investors and traders to gain an edge in the markets to make informed decisions. Fundamental Research and Technical Research are two types of research used to first analyze and then value a security

Performing a research before making an investment is a must. It is only after a thorough research that you can make some assumptions into the value and future performance of an investment. Even if you are following stock trading tips, it ideal to do some research, just to ensure that you are making an investment that's expected to get you maximum returns.When you invest in equity, you purchase some portions of a business expecting to make money upon increase in the value of the business. Before buying anything, be it a car or phone, you do some degree of research about its performance and quality. An investment is no different. It is your hard earned money that you are about to invest, so you must have a fair knowledge of what you are investing in.

2. LITERATURE SURVEY

Survey -01;-

Title Analysis of Investor Sentiment and Stock Market Volatility Trend Based on Big Data Strategy.

Year: 2019

Author : Du Peng (Dalian Ocean University; Liaoning Dalian 1

Abstract : This paper mainly studies the specific mechanism of investor sentiment affecting stock market volatility. With the help of Pollet and Wilson's theory of volatility decomposition, it performs a comparative analysis based on big data strategy and sources. This paper collects the data of web news emotion index, web search volume, social network emotion index, social network heat index, and establishes corresponding analysis index. After correlation analysis and Granger causality tests, it extracts the indicators which have significant correlation with the financial market and brings them into forecasting analysis. The model constructs market volatility index and analyzes the correlation between investor sentiment and stock price changes. In empirical study, the deviation between stock price and value is introduced as an explanatory variable, and the logarithmic return of stock is used to measure the volatility of stock price. It is found that the stock market volatility index compounded by the stock market sentiment index has a strong predictive ability for the stock market volatility turning point in the larger turbulent situation, especially for the one to two day decline turning point ahead of schedule, and it has a strong practical role for the stock market volatility prediction, as well as for financial market risk aversion.

Survey -02:

Title :- Stock Market Prediction using Supervised Machine Learning Techniques: An Overview

Year: 2020

Author :- Zaharaddeen Karami ,Hayati Yassin , Rufai Yusuf Zakar

Abstract :- Stock price prediction is one of the most extensively studied and challenging glitches, which is acting so many academicians and industries experts from many fields comprising of economics, and business, arithmetic, and computational science. Predicting the stock market is not a simple task, mainly as a magnitude of the close to random-walk behavior of a stock time series. Millions of people across the globe are investing in stock market daily. A good stock price prediction model will help investors, management and decision makers in making correct and effective decisions. In this paper, we review studies on supervised machine learning models in stock market predictions. The study discussed how supervised machine learning techniques are applied to improve accuracy of stock market predictions. Support Vector Machine (SVM) was found to be the most frequently used technique for stock price prediction due to its good performance and accuracy. Other techniques like Artificial Neural Network (ANN), K-Nearest Neighbor (KNN), Naïve Bayes, Random Forest, Linear Regression and Support Vector Regression (SVR) also showed a promising prediction result.

Survey -03

Title :- Stock Market Prediction for Time-series Forecasting using Prophet upon ARIMA

Year: 2020

Author :- CH.RAGA MADHURI ,MUKESH CHINTA,N V PHANI KUMAR

Abstract :- Since the beginning, the fundamental goal of man is to make life easy to live. The whole world believes that wealth would make life comfortable and luxurious. One of the most common notion among humans is that one of the best way to make money is to invest in stock markets which are expected to have tremendous results. There is a requirement to develop an intelligent system to perform predictions based on various indicators like fundamental, statistical and technical trends. However, there is no one good predictive model that has been successful to beat the trends in market continuously. Traditionally for time series data, the predictions are in general performed based on past historical data and market trends, historical correlation data and projections can be calculated.

Above all said, there is no such system that calculates the predictions based on users selection on investment type and on risk criteria user is willing to take. So in this paper, we tried to demonstrate the technique(s) to get most accurate results.

Survey -04:-

Title: Stock Price Prediction Using CNN and LSTMBased Deep Learning Models

Year :2020

Author :- Sidra Mehtab, Jaydip Sen

Abstract :- Designing robust and accurate predictive models for stock price prediction has been an active area of research over a long time. While on one side, the supporters of the efficient market hypothesis claim that it is impossible to forecast stock prices accurately, many researchers believe otherwise. There exist propositions in the literature that have demonstrated that if properly designed and optimized, predictive models can very accurately and reliably predict future values of stock prices. This paper presents a suite of deep learning-based models for stock price prediction. We use the historical records of the NIFTY 50 index listed in the National Stock Exchange (NSE) of India, during the period from December 29, 2008 to July 31, 2020 for training and testing the models. Our proposition includes two regression models built on convolutional neural networks (CNNs), and three long-and-short-term memory (LSTM) network-based predictive models. For the purpose of forecasting the open values of the NIFTY 50 index records, we adopted a multi-step prediction technique with walk-forward validation. In this approach, the open values of the NIFTY 50 index are predicted on a time horizon of one week, and once a week is over, the actual index values are included in the training set before the model is trained again, and the forecasts for the next week are made. We present detailed results on the forecasting accuracies for all our proposed models. The results show that while all the models are very accurate in forecasting the NIFTY 50 open values, the univariate encoder-decoder convolutional LSTM with previous two weeks' data as the input is the most accurate model. On the other hand, a univariate CNN model with previous one week's data as the input is found to be the fastest model in terms of its execution speed.

Survey -05:-

Title : Analysis of Stock Market using Streaming data Framework

Year: 2018

Author :- Umadevi.K.S*, Abhijitsingh Gaonka, umadeviks R. Jagadeesh Kannan

Abstract :-

It has been proved that the decisions in stock market exchange may bring influence on the investors, financial institutions, banking sectors etc. The stock market is a highly composite system in addition often concealed with mystery, it is therefore, very difficult to analyze all the impacting factors before making a decision. In this research, we have tried to design a stock market prediction model which is considers different parameters of a particular stock. Analysis is performed after obtaining the stock scores. This analysis involves visualization of stock scores in the form of various plots and prediction of the scores using a time series model known as ARIMA (auto regressive moving average). The results shows that the time series model performed a descent prediction of the market scores with considerably high accuracy. Each factor was studied independently to find out its association with market performance. Furthermore the results suggests that behavior of market can be predicted using machine learning techniques

3. SYSTEM ANALYSIS

3.1Existing System

In existing method analysis we used deep learning algorithm of convolutional neural network(CNN) for the classification. But, we got less training accurate results for the given data set. We should improvise the accuracy to get less false output results for the future predictions.

3.2 Problem with Existing System

1.Less amount of accuracy score

2.Small level data-set

3.Applicable on small level prediction work.

3.3 Proposed System

1. In our proposed system, we are implementing K-means clustering.

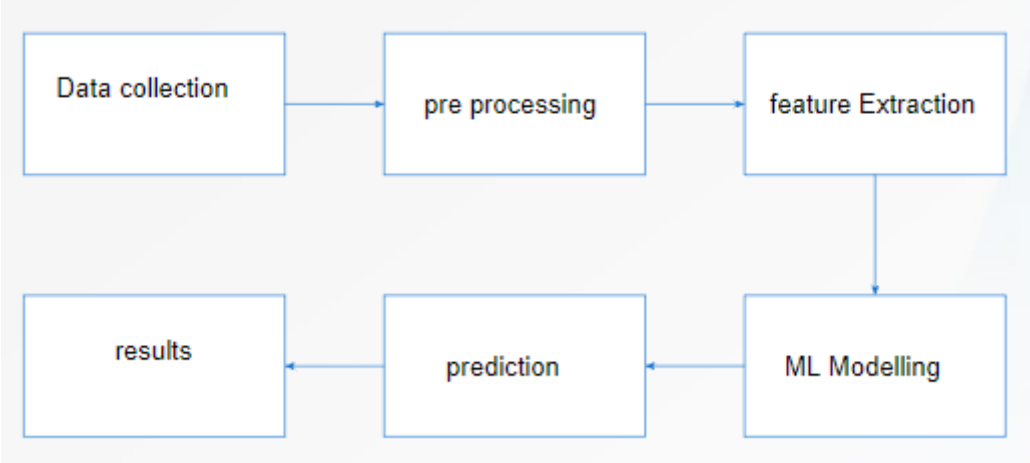
2. We are collecting the past data sets for future prediction by using random forest algorithm.

Advantages of Proposed System

1. Accuracy level is high.

2. Time consumption is less.

3.3.1. System Architecture:



3.3.2. System Requirements:

Hardware

Windows 7,8,9,10,11.

RAM 4GB.

Software

Anaconda Navigator

Python Language

Jupyter notebook

3.3.3. IMPLEMENTATION :

Feasibility study in the sense it's a practical approach of implementing the proposed model of system . Here for a machine learning projects .we generally collect the input from online websites and filter the input data and visualize them in graphical format and then the data is divided for training and testing . That training is testing data is given to the algorithms to predict the data .

1. First, we take stock market dataset.

2. Filter dataset according to requirements and create a new dataset which has attribute according to analysis to be done

3. Perform Pre-Processing on the dataset

4. Split the data into training and testing

5. Train the model with training data then analyze testing dataset over classification algorithm

6. Finally you will get results as accuracy metrics.

Algorithms description:

k-means clustering:

K-means clustering algorithm computes the centroids and iterates until we it finds optimal centroid. It assumes that the number of clusters are already known. It is also called **flat clustering** algorithm. The number of clusters identified from data by algorithm is represented by 'K' in K-means.

In this algorithm, the data points are assigned to a cluster in such a manner that the sum of the squared distance between the data points and centroid would be minimum. It is to be understood that less variation within the clusters will lead to more similar data points within same cluster.

Working of K-Means Algorithm

We can understand the working of K-Means clustering algorithm with the help of following steps −

Step 1 − First, we need to specify the number of clusters, K, need to be generated by this algorithm.

Step 2 − Next, randomly select K data points and assign each data point to a cluster. In simple words, classify the data based on the number of data points.

Step 3 − Now it will compute the cluster centroids.

Step 4 − Next, keep iterating the following until we find optimal centroid which is the assignment of data points to the clusters that are not changing any more −

4.1 − First, the sum of squared distance between data points and centroids would be computed.

4.2 − Now, we have to assign each data point to the cluster that is closer than other cluster (centroid).

4.3 − At last compute the centroids for the clusters by taking the average of all data points of that cluster.

K-means follows Expectation-Maximization approach to solve the problem. The Expectation-step is used for assigning the data points to the closest cluster and the Maximization-step is used for computing the centroid of each cluster.

While working with K-means algorithm we need to take care of the following things −

While working with clustering algorithms including K-Means, it is recommended to standardize the data because such algorithms use distance-based measurement to determine the similarity between data points.

Due to the iterative nature of K-Means and random initialization of centroids, K-Means may stick in a local optimum and may not converge to global optimum. That is why it is recommended to use different initializations of centroids.

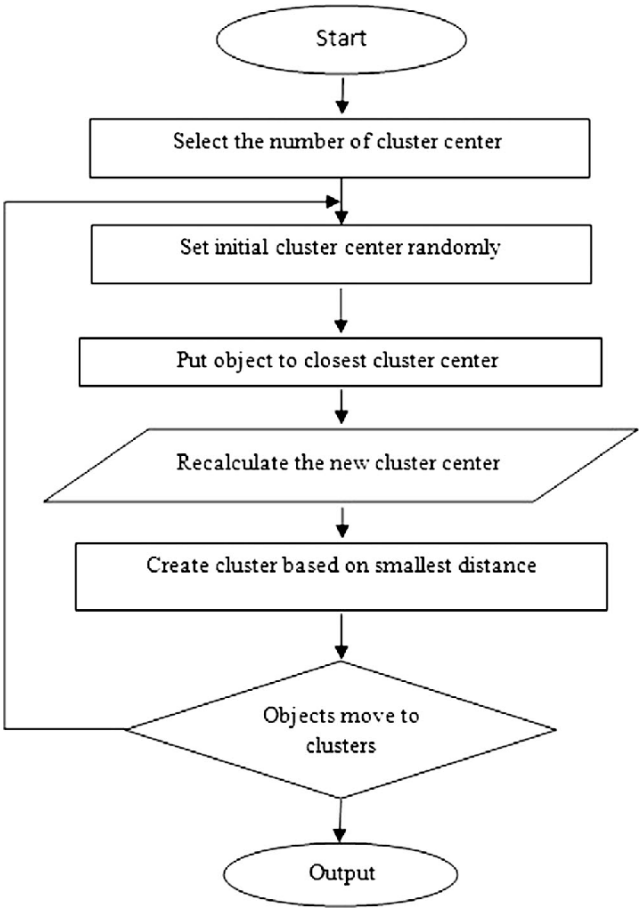
Advantages:

The following are some advantages of K-Means clustering algorithms −

1. It is very easy to understand and implement.
2. If we have large number of variables then, K-means would be faster than Hierarchical clustering.
3. On re-computation of centroids, an instance can change the cluster.
4. Tighter clusters are formed with K-means as compared to Hierarchical clustering.

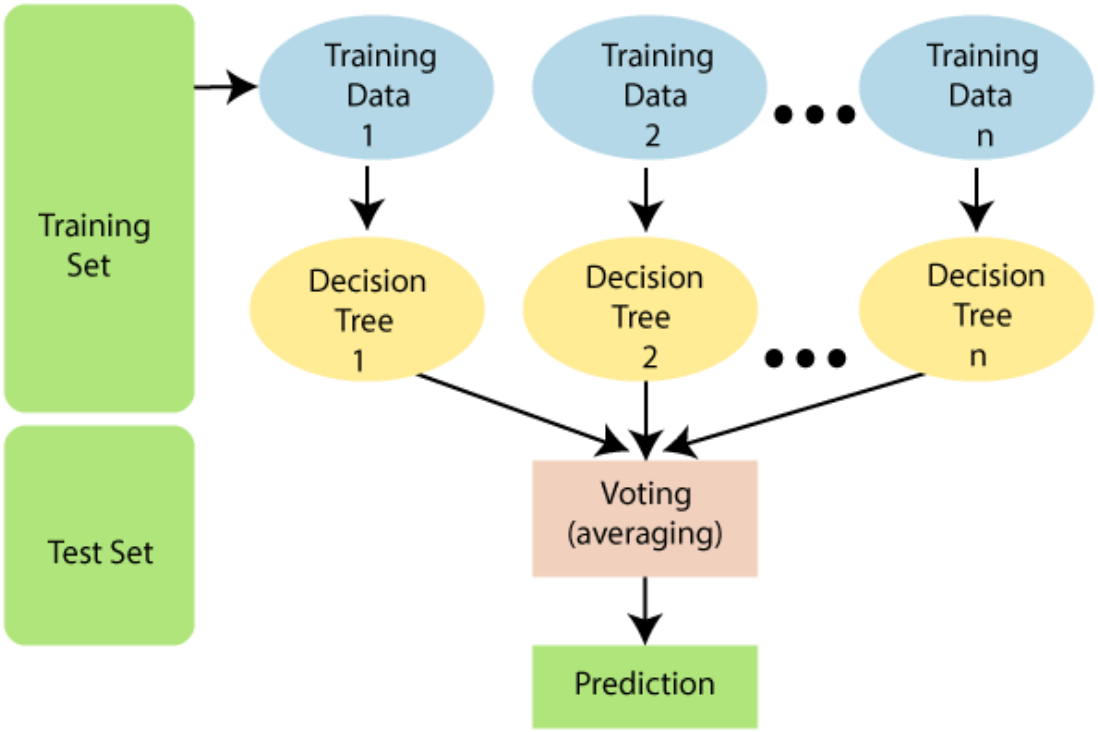
The main goals of cluster analysis are −

- To get a meaningful intuition from the data we are working with.
- Cluster-then-predict where different models will be built for different subgroups.
- To fulfill the above-mentioned goals, K-means clustering is performing well enough. It can be used in following applications −
  1. Market segmentation
  2. Document Clustering
  3. Image segmentation
  4. Image compression
  5. Customer segmentation
  6. Analyzing the trend on dynamic data.



RANDOM FOREST ALGORITHM :

Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset. Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.



HOW DOES RANDOM FOREST ALGORITHM WORK?

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

Step-1: Select random K data points from the training set.

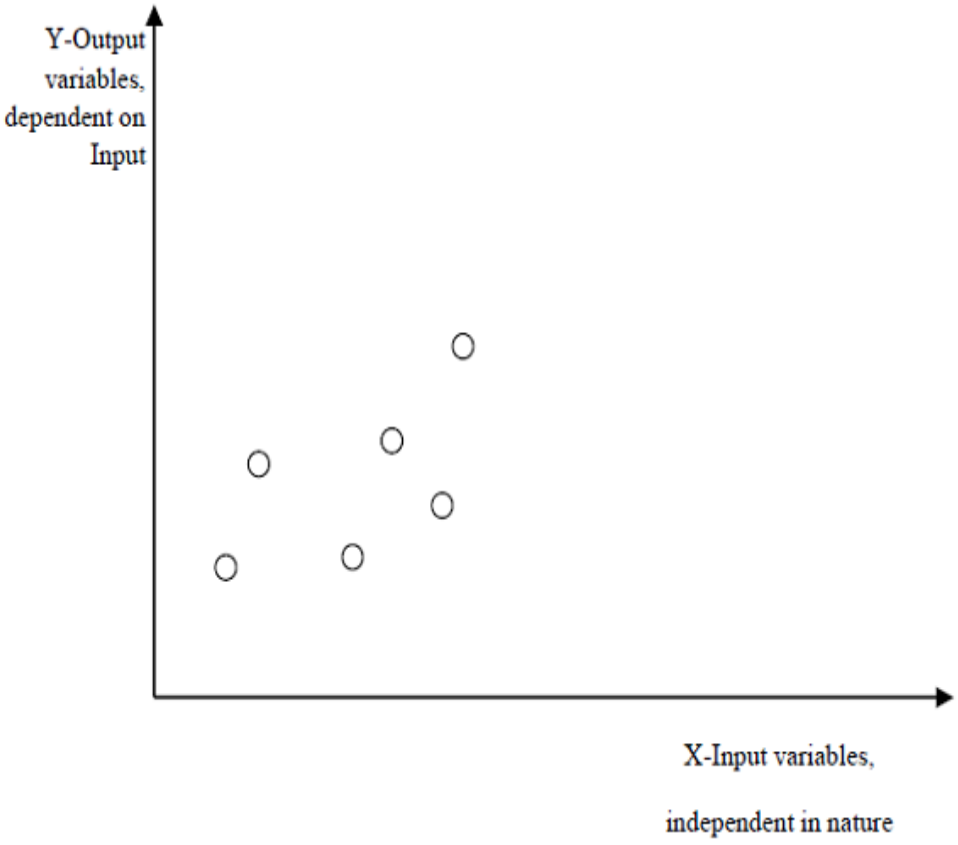
Step-2: Build the decision trees associated with the selected data points (Subsets).

Step-3: Choose the number N for decision trees that you want to build.

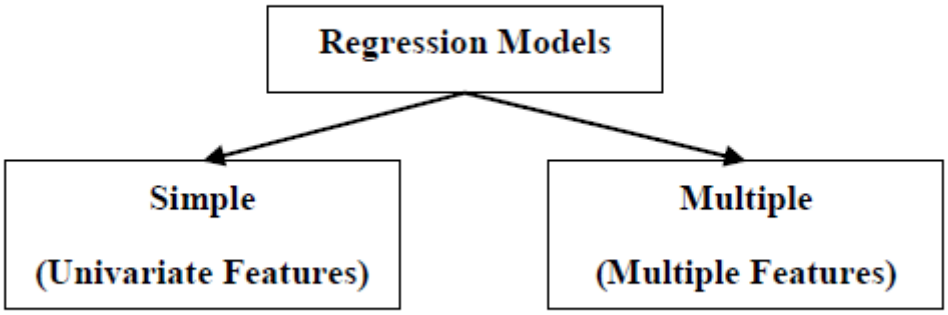
Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

Regression is another important and broadly used statistical and machine learning tool. The key objective of regression-based tasks is to predict output labels or responses which are continues numeric values, for the given input data. The output will be based on what the model has learned in training phase. Basically, regression models use the input data features (independent variables) and their corresponding continuous numeric output values (dependent or outcome variables) to learn specific association between inputs and corresponding outputs.



Types of Regression Models



Regression models are of following two types −

Simple regression model − This is the most basic regression model in which predictions are formed from a single, univariate feature of the data.

Multiple regression model − As name implies, in this regression model the predictions are formed from multiple features of the data.

Building a Regressor in Python

Regressor model in Python can be constructed just like we constructed the classifier. Scikit-learn, a Python library for machine learning can also be used to build a regressor in Python.

3.3.5. UML Diagrams

The Unified Modeling Language (UML) is used to specify, visualize, modify, construct and document the artifacts of an object-oriented software intensive system under development. UML offers a standard way to visualize a system's architectural blueprints, including elements such as:

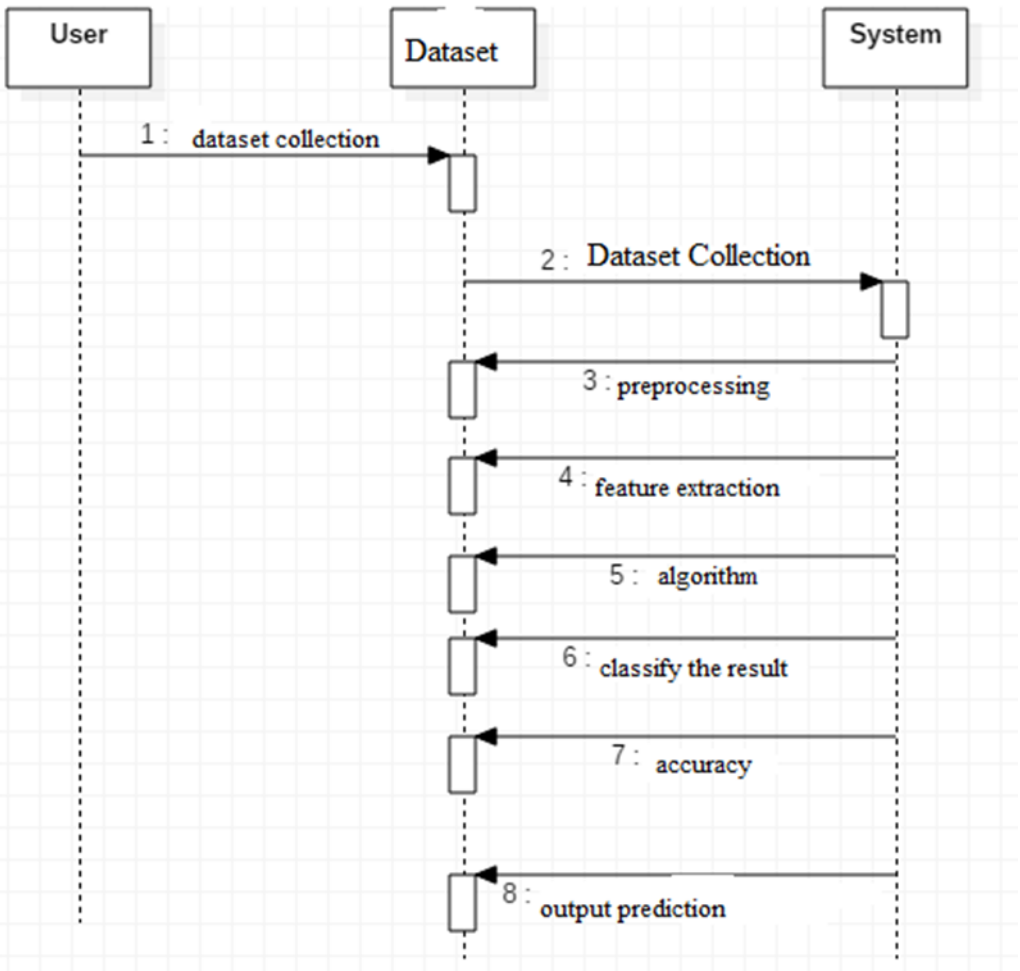
1. actors
2. business processes
3. (logical) components
4. activities

5. programming language statements
6. database schemas, and
7. Reusable software components.

UML combines best techniques from data modeling (entity relationship diagrams), business modeling (work flows), object modeling, and component modeling. It can be used with all processes, throughout the software development life cycle, and across different implementation technologies. UML has synthesized the notations of the Booch method, the Object-modeling technique (OMT) and Object-oriented software engineering (OOSE) by fusing them into a single, common and widely usable modeling language. UML aims to be a standard modeling language which can model concurrent and distributed systems.

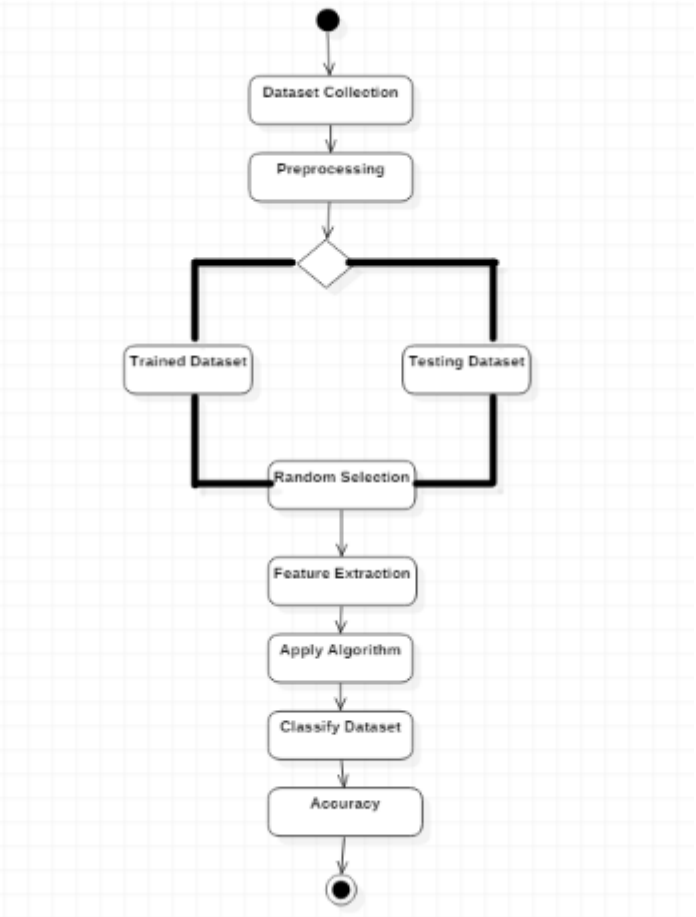
3.3.5.1. Sequence Diagram:

Sequence Diagrams Represent the objects participating the interaction horizontally and time vertically. A Use Case is a kind of behavioral classifier that represents a declaration of an offered behavior. Each use case specifies some behavior, possibly including variants that the subject can perform in collaboration with one or more actors. Use cases define the offered behavior of the subject without reference to its internal structure. These behaviors, involving interactions between the actor and the subject, may result in changes to the state of the subject and communications.



3.3.5.2. Activity Diagrams-:

Activity diagrams are graphical representations of Workflows of stepwise activities and actions with support for choice, iteration and concurrency.In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



3.3.5.3. Usecase diagram:

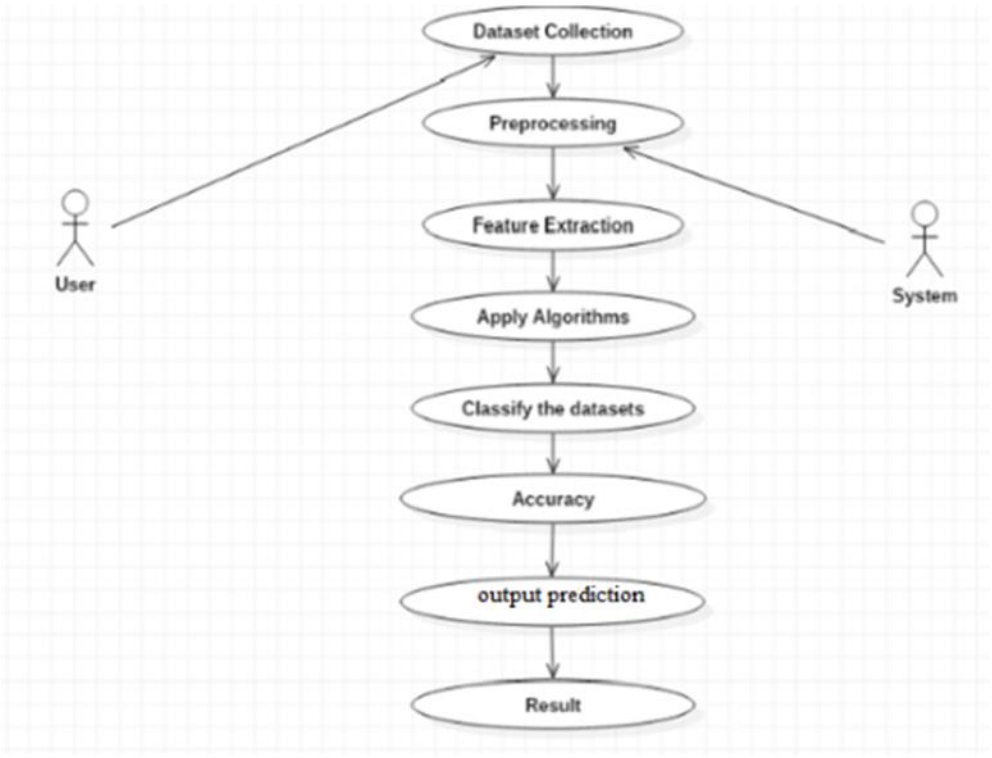
UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.

UML was created by Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997.

OMG is continuously putting effort to make a truly industry standard.

UML stands for Unified Modeling Language.

UML is a pictorial language used to make software blue prints



3.3.5.4. Class diagram

The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling.[1] The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

In the diagram, classes are represented with boxes that contain three compartments:

The top compartment contains the name of the class. It is printed in bold and centered, and the first letter is capitalized.

The middle compartment contains the attributes of the class. They are left-aligned and the first letter is lowercase.

The bottom compartment contains the operations the class can execute. They are also left-aligned and the first letter is lowercase.

4.SYSTEM DESIGN

Data Flow Diagrams

DFD is the abbreviation for **Data Flow Diagram**. The flow of data of a system or a process is represented by DFD. It also gives insight into the inputs and outputs of each entity and the process itself. DFD does not have control flow and no loops or decision rules are present. Specific operations depending on the type of data can be explained by a flowchart.

It is a graphical tool, useful for communicating with users ,managers and other personnel. it is useful for analyzing existing as well as proposed system.

It provides an overview of

- What data is system processes.
- What transformation are performed.
- What data are stored.
- What results are produced , etc.

Data Flow Diagram can be represented in several ways. The DFD belongs to structured-analysis modeling tools. Data Flow diagrams are very popular because they help us to visualize the major steps and data involved in software-system processes.

The Data Flow Diagram has 4 components:

Process Input to output transformation in a system takes place because of process function. The symbols of a process are rectangular with rounded corners, oval, rectangle or a circle. The process is named a short sentence, in one word or a phrase to express its essence

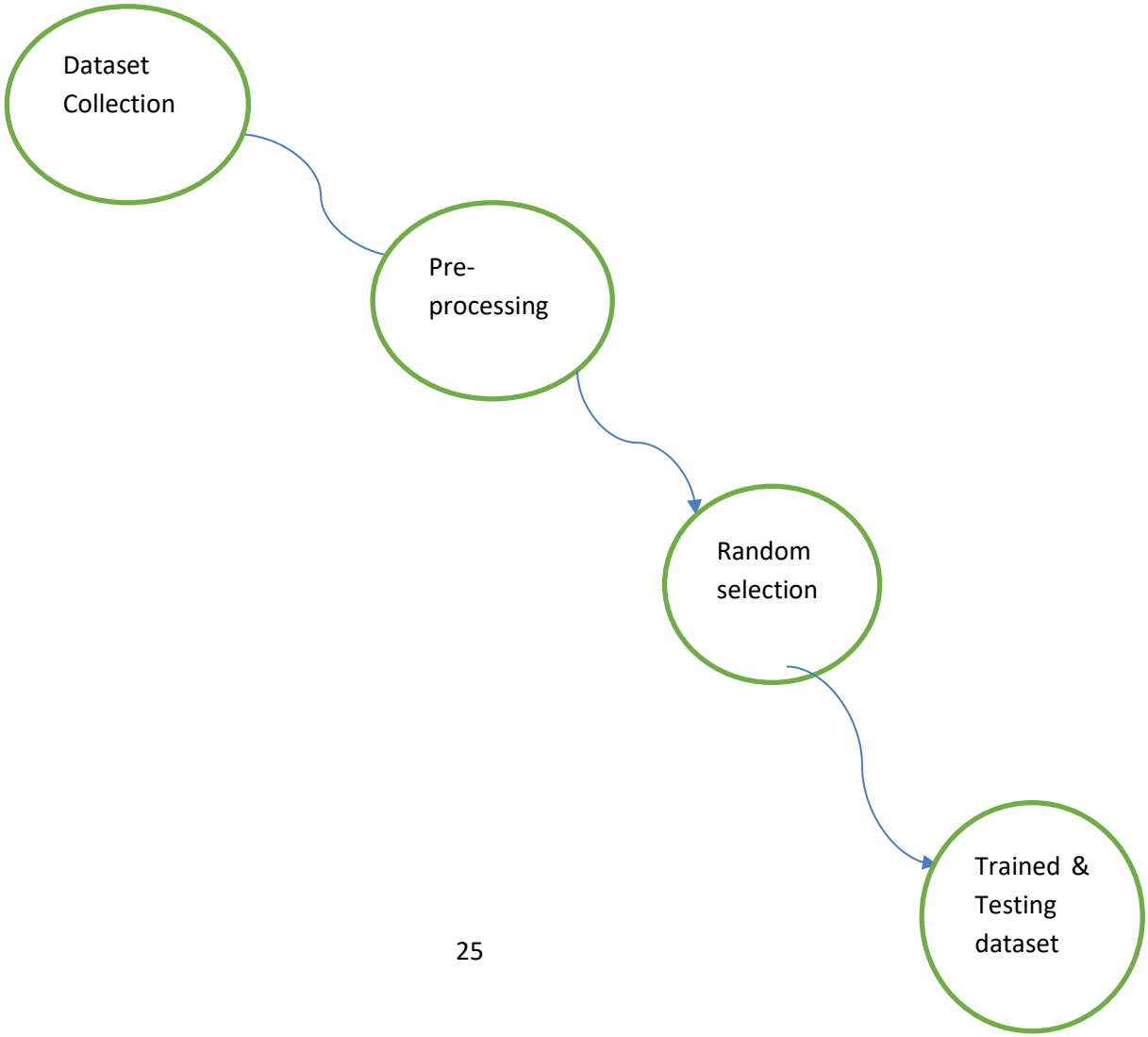
Data Flow Data flow describes the information transferring between different parts of the systems. The arrow symbol is the symbol of data flow. A relatable name should be given to the flow to determine the information which is being moved. Data flow also represents material along with information that is being moved. Material shifts are modeled in systems that are not merely informative. A given flow should only transfer a single type of information. The direction of flow is represented by the arrow which can also be bi-directional.

Warehouse The data is stored in the warehouse for later use. Two horizontal lines represent the symbol of the store. The warehouse is simply not restricted to being a data file rather it can be anything like a folder with documents, an optical disc, a filing cabinet. The data warehouse can be viewed independent of its implementation. When the data flow from the warehouse it is considered as data reading and when data flows to the warehouse it is called data entry or data updating.

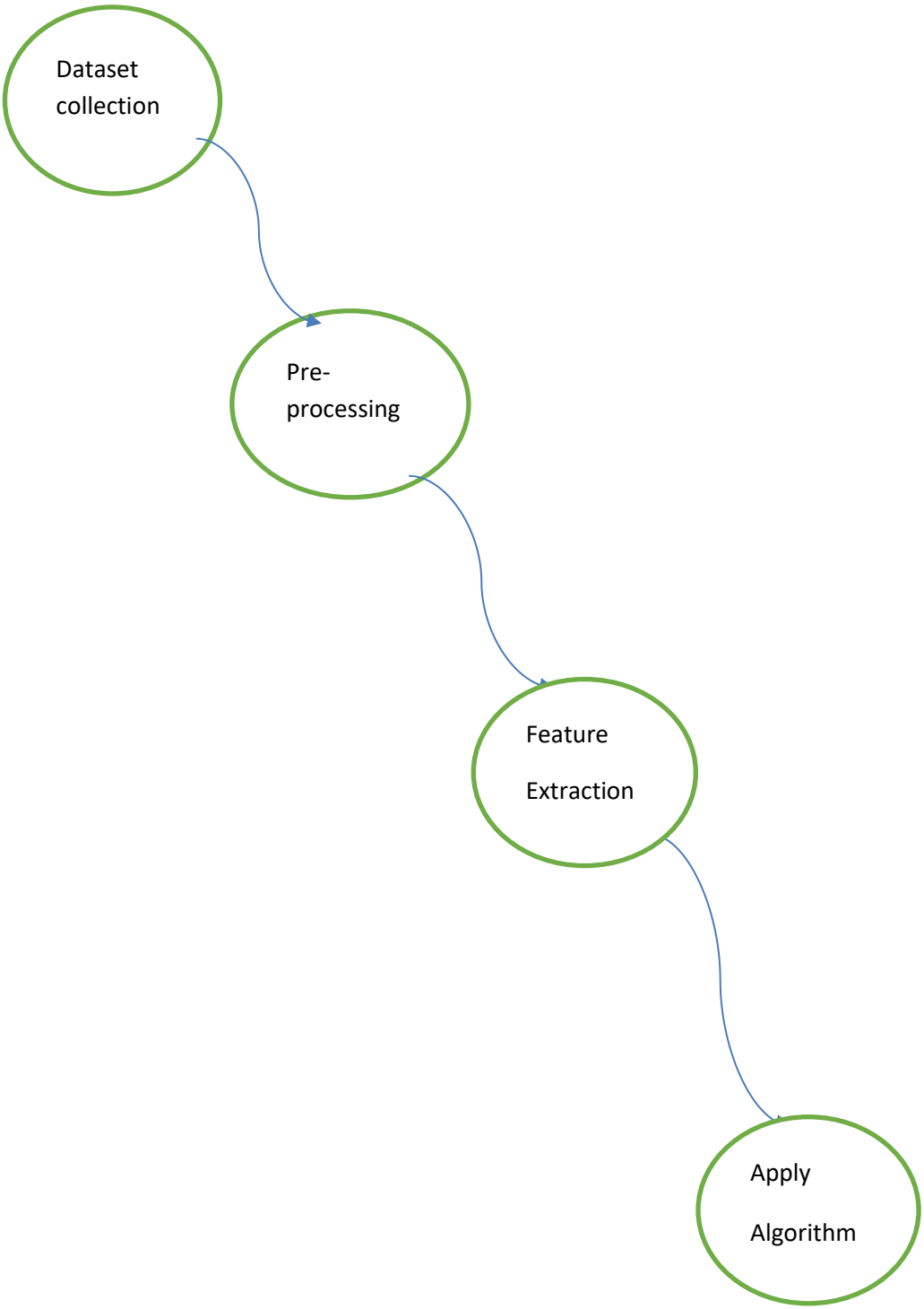
Terminator The Terminator is an external entity that stands outside of the system and communicates with the system. It can be, for example, organizations like banks, groups of people like customers or different departments of the same organization, which is not a part of the model system and is an external entity. Modeled systems also communicate with terminator.

DATA FLOW DIAGRAM

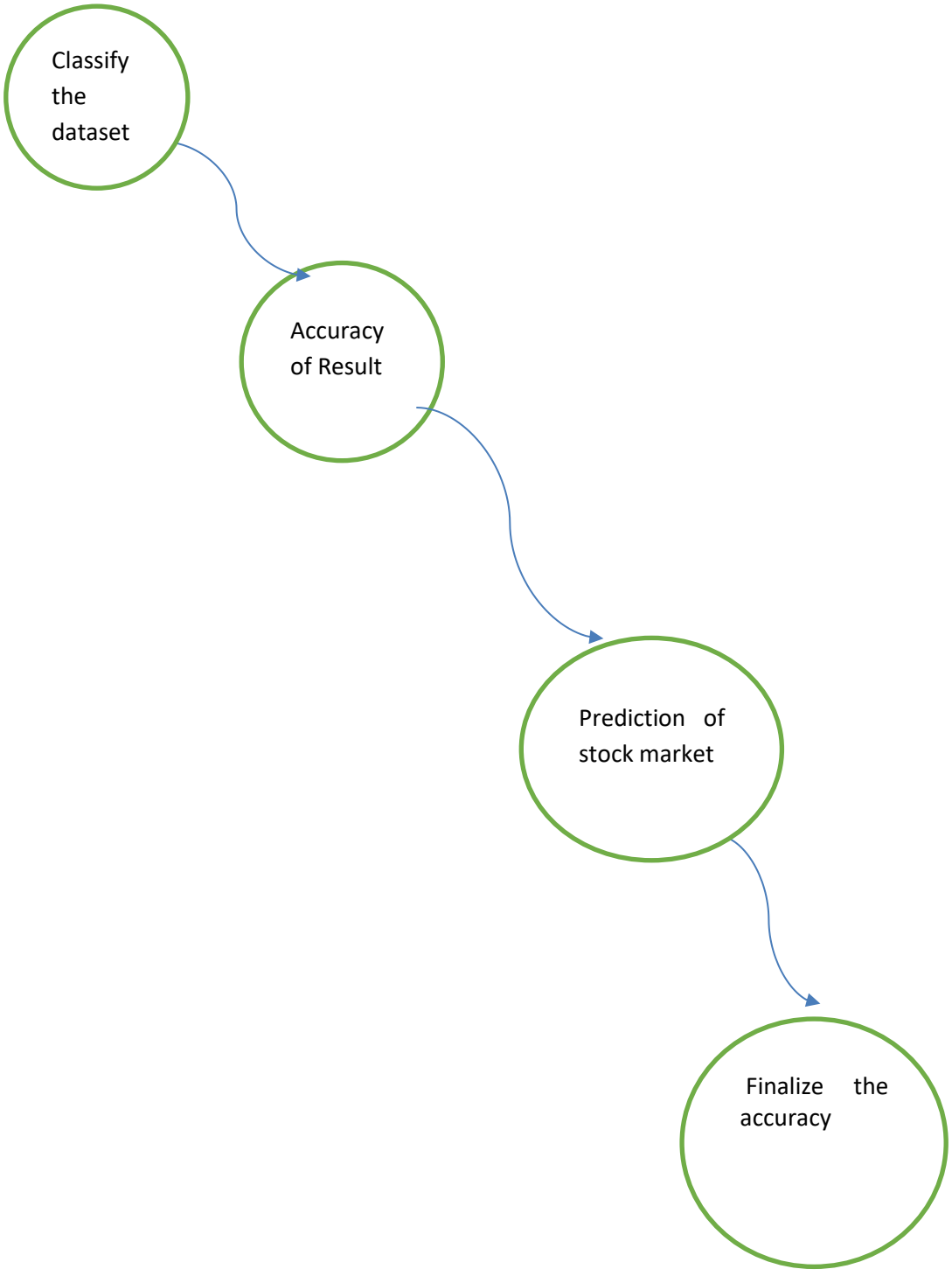
LEVEL 0



LEVEL 1


Dataset collection
Pre-processing
Feature Extraction
Apply Algorithm

LEVEL 2

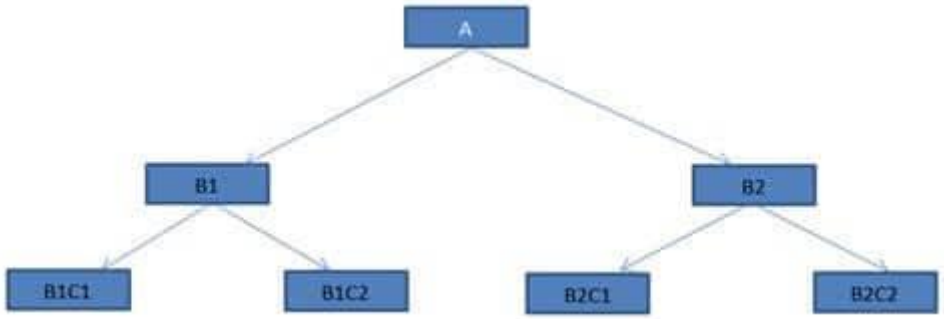

Classify the dataset
Accuracy of Result
Prediction of stock market
Finalize the accuracy

Test Integration Approaches

There are fundamentally 2 approaches for doing test integration:

1.Bottom-up approach

2.Top-down approach.



Bottom-up approach

Bottom-up testing, as the name suggests starts from the lowest or the innermost unit of the application, and gradually moves up. The Integration testing starts from the lowest module and gradually progresses towards the upper modules of the application. This integration continues till all the modules are integrated and the entire application is tested as a single unit.

Top-down approach

This technique starts from the topmost module and gradually progress towards the lower modules. Only the top module is unit tested in isolation. After this, the lower modules are integrated one by one. The process is repeated until all the modules are integrated and tested.

5.SYSTEM IMPLEMENTATION

1. **DATA COLLECTION**
2. **DATA PRE-PROCESSING**
3. **FEATURE EXTRATION**
4. **EVALUATION MODEL**
5. **FLASK**
6. **STREAMLIT**

DATA COLLECTION

Data collection is a process in which information is gathered from many sources which is later used to develop the machine learning models. The data should be stored in a way that makes sense for problem. In this step the data set is converted into the understandable format which can be fed into machine learning models.

Data used in this paper is a set of stock market dataset with some features . This step is concerned with selecting the subset of all available data that you will be working with. ML problems start with data preferably, lots of data (examples or observations) for which you already know the target answer. Data for which you already know the target answer is called *labelled data*.

DATA PRE-PROCESSING

Organize your selected data by formatting, cleaning and sampling from it.

Three common data pre-processing steps are:

Formatting: The data you have selected may not be in a format that is suitable for you to work with. The data may be in a relational database and you would like it in a flat file, or the data may be in a proprietary file format and you would like it in a relational database or a text file.

Cleaning: Cleaning data is the removal or fixing of missing data. There may be data instances that are incomplete and do not carry the data you believe you need to address the problem. These instances may need to be removed. Additionally, there may be sensitive information in some of

the attributes and these attributes may need to be anonymized or removed from the data entirely.

Sampling: There may be far more selected data available than you need to work with. More data can result in much longer running times for algorithms and larger computational and memory requirements. You can take a smaller representative sample of the selected data that may be much faster for exploring and prototyping solutions before considering the whole dataset.

FEATURE EXTRATION

Next thing is to do Feature extraction is an attribute reduction process. Unlike feature selection, which ranks the existing attributes according to their predictive significance, feature extraction actually transforms the attributes. The transformed attributes, or features, are linear combinations of the original attributes. Finally, our models are trained using Classifier algorithm. We use classify module on Natural Language Toolkit library on Python. We use the labelled dataset gathered. The rest of our labelled data will be used to evaluate the models. Some machine learning algorithms were used to classify pre-processed data. The chosen classifiers were Random forest. These algorithms are very popular in text classification tasks.

EVALUATION MODEL

Model Evaluation is an integral part of the model development process. It helps to find the best model that represents our data and how well the chosen model will work in the future. Evaluating model performance with the data used for training is not acceptable in data science because it can easily generate overoptimistic and over fitted models. There are two methods of evaluating models in data science, Hold-Out and Cross-Validation. To avoid over fitting, both methods use a test set (not seen by the model) to evaluate model performance.

Performance of each classification model is estimated base on its averaged. The result will be in the visualized form. Representation of classified data in the form of graphs.

Accuracy is defined as the percentage of correct predictions for the test data. It can be calculated easily by dividing the number of correct predictions by the number of total predictions.

FLASK

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

Features:-

- Development server and debugger
- Integrated support for unit testing
- RESTful request dispatching
- Uses Jinja templating
- Support for secure cookies (client side sessions)
- 100% WSGI 1.0 compliant
- Unicode-based
- Complete documentation
- Google App Engine compatibility
- Extensions available to extend functionality

Example:

The following code shows a simple web application that displays "Hello World!" when visited:

```
from flask import Flaskapp = Flask(__name__)
@app.route("/")def hello() -> str:
    return "Hello World"

if __name__ == "__main__":
    app.run(debug=False)
```

Templates are files that contain static data as well as placeholders for dynamic data. A template is rendered with specific data to produce a final document. Flask uses the Jinja template library to render templates.

In your application, you will use templates to render HTML which will display in the user's browser. In Flask, Jinja is configured to autoescape any data that is rendered in HTML templates. This means that it's safe to render user input; any characters they've entered that could mess with the HTML, such as `<` and `>` will be escaped with safe values that look the same in the browser but don't cause unwanted effects.

Jinja looks and behaves mostly like Python. Special delimiters are used to distinguish Jinja syntax from the static data in the template. Anything between `{{` and `}}` is an expression that will be output to the final document. `{%` and `%}` denotes a control flow statement like `if` and `for`. Unlike Python, blocks are denoted by start and end tags rather than indentation since static text within a block could change indentation.

Each page in the application will have the same basic layout around a different body. Instead of writing the entire HTML structure in each template, each template will extend a base template and override specific sections.

```
<!doctype html>
<title>{% block title %}{% endblock %} - Flaskr</title>
<link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
<nav>
  <h1>Flaskr</h1>
  <ul>
    {% if g.user %}
      <li><span>{{ g.user['username'] }}</span>
      <li><a href="{{ url_for('auth.logout') }}">Log Out</a>
    {% else %}
      <li><a href="{{ url_for('auth.register') }}">Register</a>
      <li><a href="{{ url_for('auth.login') }}">Log In</a>
    {% endif %}
  </ul>
</nav>
<section class="content">
  <header>
    {% block header %}{% endblock %}
  </header>
  {% for message in get_flashed_messages() %}
    <div class="flash">{{ message }}</div>
  {% endfor %}
  {% block content %}{% endblock %}
</section>
```

g is automatically available in templates. Based on if `g.user` is set (from `load_logged_in_user`), either the username and a log out link are displayed, or links to register and log in are displayed. **url_for()** is also automatically available, and is used to generate URLs to views instead of writing them out manually.

STREAMLIT:

Streamlit is an open-source (free) Python library, which provides a fast way to build interactive web apps. It is a relatively new package but has been growing tremendously. It is designed and built especially for machine learning engineers or other data science professionals. Once you are done with your analysis in Python, you can quickly turn those scripts into web apps/tools to share with others.

As long as you can code in Python, it should be straightforward for you to write Streamlit apps. Imagine the app as the canvas. We can write Python scripts from top to bottom to add all kinds of elements, including text, charts, widgets, tables, etc.

Streamlit is the easiest way especially for people with no front-end knowledge to put their code into a web application:

- No front-end (html, js, css) experience or knowledge is required.
- You don't need to spend days or months to create a web app, you can create a really beautiful machine learning or data science app in only a few hours or even minutes.
- It is compatible with the majority of Python libraries (e.g. pandas, matplotlib, seaborn, plotly, Keras, PyTorch, SymPy(latex)).
- Less code is needed to create amazing web apps.
- Data caching simplifies and speeds up computation pipelines.

NumPy:

NumPy (Numerical Python) is an open source Python library that's used in almost every field of science and engineering. It's the universal standard for working with numerical data in Python, and it's at the core of the scientific Python and PyData ecosystems. NumPy users include everyone from beginning coders to experienced researchers doing state-of-the-art scientific and industrial research and development. The NumPy API is used extensively in Pandas, SciPy, Matplotlib, scikit-learn, scikit-image and most other data science and scientific Python packages. The NumPy library contains multidimensional array and matrix data structures (you'll find more information about this in later sections). It provides ndarray, a homogeneous n-dimensional array object, with methods to efficiently operate on it. NumPy can be used to perform a wide variety of mathematical operations on arrays. It adds powerful data structures to Python that guarantee efficient calculations with arrays and matrices and it supplies an enormous library of high-level mathematical functions that operate on these arrays and matrices.

Installing NumPy:

```
pip install numpy
```

How to import NumPy:

```
import numpy as np
```

An array is a central data structure of the NumPy library. An array is a grid of values and it contains information about the raw data, how to locate an element, and how to interpret an element. It has a grid of elements that can be indexed in various ways. The elements are all of the same typef nonnegative integers, by booleans, by another array, or by integers.

The rank of the array is the number of dimensions. The shape of the array is a tuple of integers giving the size of the array along each dimension.

Pandas:, referred to as the array dtype.

An array can be indexed by a tuple o

pandas is a Python package that provides fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis / manipulation tool available in any language. It is already well on its way towards this goal.

Main Features

- Easy handling of missing data (represented as `NaN`, `NA`, or `NaT`) in floating point as well as non-floating point data
- Size mutability: columns can be inserted and deleted from DataFrame and higher dimensional objects
- Automatic and explicit data alignment: objects can be explicitly aligned to a set of labels, or the user can simply ignore the labels and let `Series`, `DataFrame`, etc. automatically align the data for you in computations
- Powerful, flexible group by functionality to perform split-apply-combine operations on data sets, for both aggregating and transforming data
- Make it easy to convert ragged, differently-indexed data in other Python and NumPy data structures into DataFrame objects

Install Pandas:

```
pip install pandas
```

Matplotlib:
Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002. One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc. Installation : Windows, Linux and macOS distributions have matplotlib and most of its dependencies as wheel packages.

Installation:

pip install matplotlib

Run the following command to install matplotlib package:

python -mpip install -U matplotlib

Matplotlib comes with a wide variety of plots. Plots helps to understand trends, patterns, and to make correlations. They're typically instruments for reasoning about quantitative information.

Seaborn:
Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data. One has to be familiar with Numpy and Matplotlib and Pandas to learn about Seaborn.Seaborn offers the following functionalities:

- Dataset oriented API to determine the relationship between variables.
- Automatic estimation and plotting of linear regression plots.
- It supports high-level abstractions for multi-plot grids.
- Visualizing univariate and bivariate distribution**.**

To initialize the Seaborn library, the command used is:

```
import seaborn as sns
```

To get a little overview here are a few popular plotting libraries:

1. **Matplotlib:** low level, provides lots of freedom
2. **Pandas Visualization:** easy to use interface, built on Matplotlib
3. **Seaborn:** high-level interface, great default styles
4. **ggplot:** based on R's ggplot2, uses Grammar of Graphics
5. **Plotly:** can create interactive plots

In this article, we will learn how to create basic plots using Matplotlib, Pandas visualization and Seaborn as well as how to use some specific features of each library. This article will focus on the syntax and not on interpreting the graphs.

Matplotlib

Matplotlib is the most popular python plotting library. It is a low level library with a Matlab like interface which offers lots of freedom at the cost of having to write more code.

1. To install Matplotlib pip and conda can be used.

2. pip install matplotlib

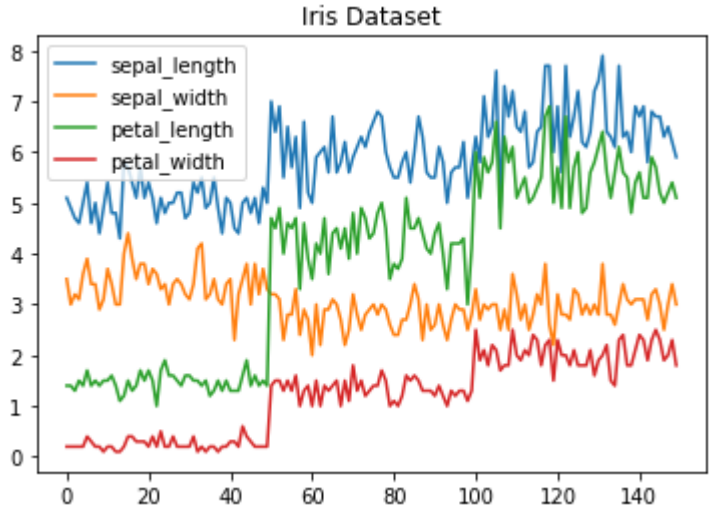
3. conda install matplotlib

Matplotlib is specifically good for creating basic graphs like line charts, bar charts, histograms and many more. It can be imported by typing:

- **import matplotlib.pyplot as plt**

Line Chart

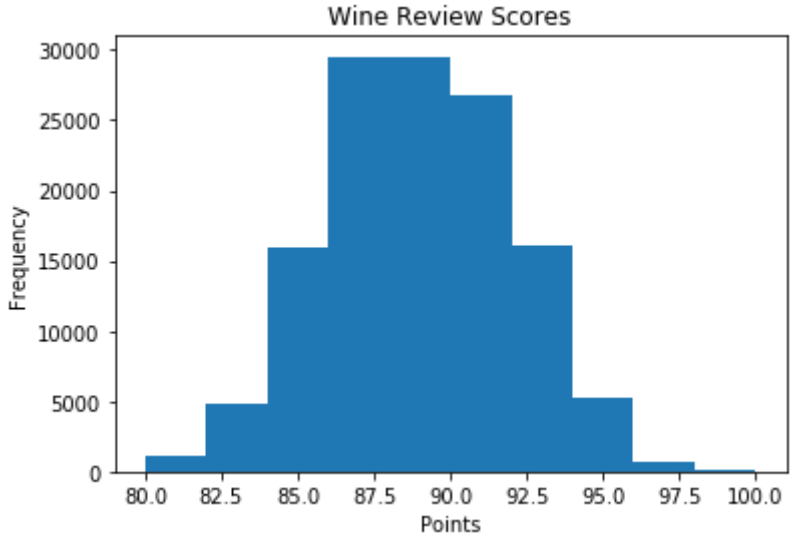
In Matplotlib we can create a line chart by calling the plot method. We can also plot multiple columns in one graph, by looping through the columns we want, and plotting each column on the same axis.



Line Chart

Histogram

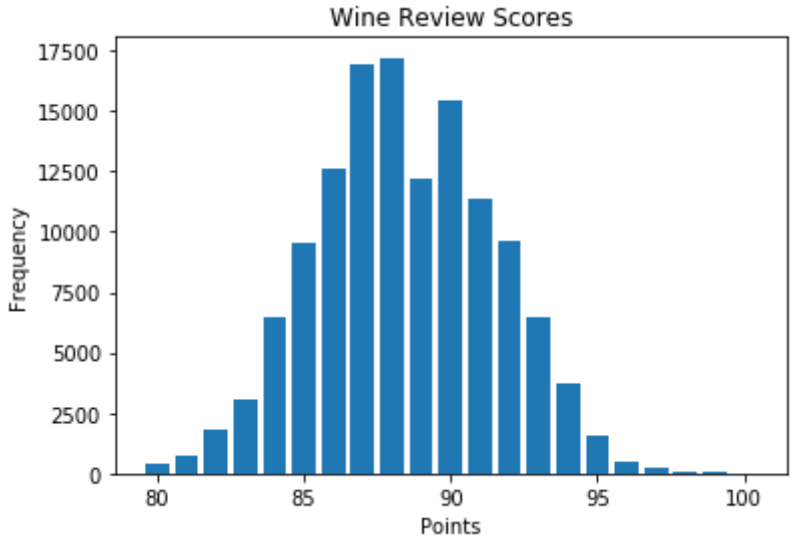
In Matplotlib we can create a Histogram using the hist method. If we pass it categorical data like the points column from the wine-review dataset it will automatically calculate how often each class occurs.



Histogram

Bar Chart

A bar-chart can be created using the bar method. The bar-chart isn't automatically calculating the frequency of a category so we are going to use pandas value_counts function to do this. The bar-chart is useful for categorical data that doesn't have a lot of different categories (less than 30) because else it can get quite messy.



Bar-Chart

Pandas Visualization

Pandas is a open source high-performance, easy-to-use library providing data structures, such as dataframes, and data analysis tools like the visualization tools we will use in this article.

Pandas Visualization makes it really easy to create plots out of a pandas dataframe and series. It also has a higher level API than Matplotlib and therefore we need less code for the same results.

- **Pandas can be installed using either pip or conda.**

- **pip install pandas**

- **conda install pandas**

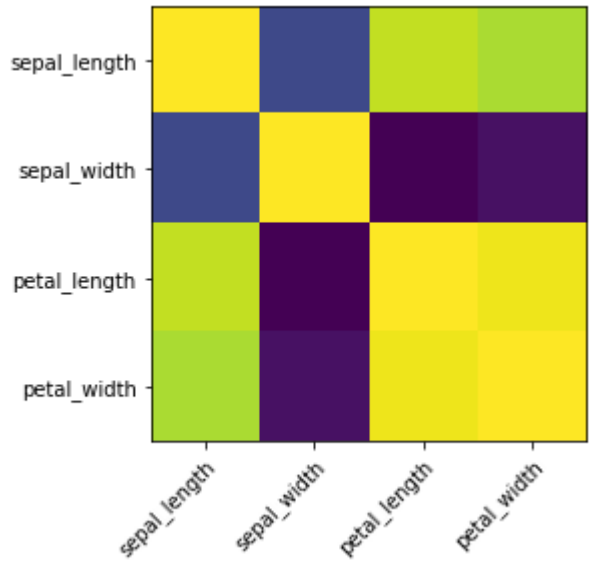
Heatmap

A Heatmap is a graphical representation of data where the individual values contained in a matrix are represented as colors. Heatmaps are perfect for exploring the correlation of features in a dataset.

To get the correlation of the features inside a dataset we can call <dataset>.corr() , which is a Pandas dataframe method. This will give use the correlation matrix.

We can now use either Matplotlib or Seaborn to create the heatmap.

Matplotlib:



Heatmap without annotations

Data visualization is the discipline of trying to understand data by placing it in a visual context, so that patterns, trends and correlations that might not otherwise be detected can be exposed.

Python offers multiple great graphing libraries that come packed with lots of different features. In this article we looked at Matplotlib, Pandas visualization and Seaborn.

Scikit-learn:

Scikit-learn is an open source data analysis library, and the gold standard for Machine Learning (ML) in the Python ecosystem

Concepts And Features:

Classification: identifying and categorizing data based on patterns

Regression: predicting or projecting data values based on the average mean of existing and planned data.

Clustering: automatic grouping of similar data into datasets.

Algorithms that support predictive analysis ranging from simple linear regression to neural network pattern recognition.

Interoperability with NumPy, pandas, and matplotlib libraries.

ML is a technology that enables computers to learn from input data and to build/train a predictive model without explicit programming. ML is a subset of Artificial Intelligence (AI).

Whether you are just looking for an introduction to ML, want to get up and running fast, or are looking for the latest ML research tool, you will find that scikit-learn is both well-documented and easy to learn/use. As a high-level library, it lets you define a predictive data model in just a few lines of code, and then use that model to fit your data. It's versatile and integrates well with other Python libraries, such **as** matplotlib for plotting**,** numpy for array vectorization, and pandas for data frames**.**

6.Source code:

Random forest algorithm:

```
# Importing all the required libraries

import numpy as np
import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt
from matplotlib import style
from sklearn import preprocessing
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.model_selection import train_test_split
from datetime import datetime, timedelta

%matplotlib inline

# The historical data of stock prices of Apple was downloaded from Yahoo! Finance Website in the .csv format
#Website - https://finance.yahoo.com/quote/AAPL/history?p=AAPL

# Reading the CSV file

df = pd.read_csv('stock.csv')
df.set_index('Date', inplace=True)
df.tail()


# Visualizing the stock prices
df['Adj Close'].plot(label='AAPL', figsize=(15, 9), title='Adjusted Closing Price', color='red', linewidth=1.0, grid=True)
plt.legend()

# Rolling Mean / Moving Average to remove the noise in the graph and smoothen it
```

```
close_col = df['Adj Close']
mvag = close_col.rolling(window=100).mean()      # Taking an average over the window size of 100.
# Increasing the window size can make it more smoother, but less informative and vice-versa.

# Visualizing Rolling Mean and Adjusted Closing Price together

df['Adj Close'].plot(label='AAPL', figsize=(15,10), title='Adjusted Closing Price vs Moving Average', color='red', linewidth=1.0, grid=True)
mvag.plot(label='MVAG', color='blue')
plt.legend()

# Return Deviation measures the Mean of the Probability Distribution of Investment Returns if it has a positive/negative Average Net Outcome

rd = close_col / close_col.shift(1) - 1
rd.plot(label='Return', figsize=(15, 10), title='Return Deviation', color='red', linewidth=1.0, grid=True)
plt.legend()
# Number of days for which to predict the stock prices

predict_days = 30

# Shifting by the Number of Predict days for Prediction array

df['Prediction'] = df['Adj Close'].shift(-predict_days)
# print(df['Prediction'])
# print(df['Adj Close'])
df

# Dropping the Prediction Row

X = np.array(df.drop(['Prediction'], axis = 1))
X = X[:-predict_days]      # Size upto predict days
# print(X)
print(X.shape)
```

```
# Creating the Prediction Row

y = np.array(df['Prediction'])
y = y[:-predict_days]      # Size upto predict_days
# print(y)
print(y.shape)


# Splitting the data into Training data & Testing data

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
#Splitting the data into 80% for training & 20% for testing
print(X_train.shape)
print(y_train.shape)
print(X_test.shape)
print(y_test.shape)

## 1. Random forest Regression
Building First model of regression - Linear Regression

from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor(n_estimators=100, random_state=0)
regressor.fit(X_train, y_train)

regressor_score = regressor.score(X_test, y_test)
print('regressor score:', regressor_score)

X_predict = np.array(df.drop(['Prediction'], 1))[-predict_days:]

regressor_predict_prediction = regressor.predict(X_predict)
regressor_real_prediction                                    =
regressor.predict(np.array(df.drop(['Prediction'], 1)))
```

```
# Defining some Parameters
predicted_dates = []
recent_date = df.index.max()
display_at = 1000
alpha = 0.5

for i in range(predict_days):
    recent_date += str(timedelta(days=1))
    predicted_dates.append(recent_date)

# Plotting the Actual and Prediction Prices

plt.figure(figsize=(15, 9))
plt.plot(df.index[display_at:],regressor_real_prediction[display_at:],
label='Linear Prediction', color='blue', alpha=alpha)
plt.plot(predicted_dates,regressor_predict_prediction,label='Forecast',
color='green', alpha=alpha)
plt.plot(df.index[display_at:],df['Close'][display_at:],          label='Actual',
color='red')
plt.legend()
```

k- means clustering:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn import decomposition
from sklearn import datasets
import seaborn as sns
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
%matplotlib inline

df = pd.read_csv('stock.csv')
df.head()
```

```
df1 = df.drop("Date", axis='columns')

df1.shape

from sklearn.preprocessing import StandardScaler
features = df1.values
sc = StandardScaler()
X_scaled = sc.fit_transform(features)

X_scaled.shape
```

Determining optimal number of components for PCA looking at the explained variance as a function of the components

```
sns.set_style('whitegrid')
pca = PCA().fit(X_scaled)
plt.plot(np.cumsum(pca.explained_variance_ratio_))
plt.xlabel('number of components')
plt.ylabel('cumulative explained variance')
plt.show()
```

Apply PCA to reduce the number of dimensions from 502 to 2 dimensions for better data visualization.

```
pca = PCA(n_components=2)
pca.fit(X_scaled)
print('explained variance :')
print('----------------------------------------------------------------------')
print(pca.explained_variance_)
print('----------------------------------------------------------------------')
print('PCA Components : ')
print('----------------------------------------------------------------------')
print(pca.components_)
print('----------------------------------------------------------------------')
X_transformed = pca.transform(X_scaled)
print('Transformed Feature values first five rows :')
```

```
print('--------------------------------------------------------------------')
print(X_transformed[:5,:])
print('--------------------------------------------------------------------')
print('Transformed Feature shape :')
print('--------------------------------------------------------------------')
print(X_transformed.shape)
print('--------------------------------------------------------------------')
print('Original Feature shape :')
print('--------------------------------------------------------------------')
print(X_scaled.shape)
print('--------------------------------------------------------------------')
print('Restransformed Feature shape :')
print('--------------------------------------------------------------------')
X_retransformed = pca.inverse_transform(X_transformed)
print(X_retransformed.shape)
print('--------------------------------------------------------------------')
print('Retransformed Feature values first five rows :')
print('--------------------------------------------------------------------')
print(X_retransformed[:5,:])
print('--------------------------------------------------------------------')
```

Problem 1:¶

There are various stocks for which we have collected a data set, which all stocks are apparently similar in performance

Finding optimum number of clusters for KMEANS cluster

```
X_transformed.shape
```

```
!pip install scikit-plot
```

Elbow method

```
import scikitplot
scikitplot.cluster.plot_elbow_curve(KMeans(),X_transformed,cluster_ranges=range(1,20))
```

Optimum number of cluster from the elbow method is determined to be 5

Applying K-Means Clustering to find stocks which are similar in performance

```
X_train = pd.DataFrame(X_transformed)
X_train.rename(columns = {0:"Stock-1", 1:"Stock-2"})
```

Kmean Algorithm

```
k_means = KMeans(n_clusters=5,random_state=0,init='k-means++')
k_means.fit(X_train)
y_kmeans = k_means.fit_predict(X_train)
labels = k_means.labels_
```

```
len(labels), X_train.shape
```

```
plt.scatter(X_transformed[y_kmeans == 0, 0], X_transformed[y_kmeans == 0, 1], s = 100, c = 'red', label = 'Cluster 1')
plt.scatter(X_transformed[y_kmeans == 1, 0], X_transformed[y_kmeans == 1, 1], s = 100, c = 'blue', label = 'Cluster 2')
plt.scatter(X_transformed[y_kmeans == 2, 0], X_transformed[y_kmeans == 2, 1], s = 100, c = 'green', label = 'Cluster 3')
plt.scatter(X_transformed[y_kmeans == 3, 0], X_transformed[y_kmeans == 3, 1], s = 100, c = 'cyan', label = 'Cluster 4')
plt.scatter(X_transformed[y_kmeans == 4, 0], X_transformed[y_kmeans == 4, 1], s = 100, c = 'magenta', label = 'Cluster 5')
plt.scatter(k_means.cluster_centers_[:, 0], k_means.cluster_centers_[:, 1], s = 300, c = 'yellow', label = 'Centroids')
plt.title('Clusters of stocks')
plt.xlabel('Principal Component (1)')
plt.ylabel('Principal Component (2)')
plt.legend()
plt.show()
```

The above 5 clusters shows the stocks which are similar in stock performance

```
import pickle
```

```
filename = 'final_model_Kmean_clustring.pkl'
pickle.dump(k_means, open(filename, 'wb'))
```

7.RESULTS

7.1 Screenshots

i. Importing packages:

```
# Importing all the required libraries

import numpy as np
import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt
from matplotlib import style
from sklearn import preprocessing
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.model_selection import train_test_split
from datetime import datetime, timedelta
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn import decomposition
from sklearn import datasets
import seaborn as sns
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
%matplotlib inline
```

ii. **Data Collection**

```
df = pd.read_csv('stock.csv')
```

```
df.head()
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2021-07-12	146.210007	146.320007	144.000000	144.500000	143.679138	76299700
1	2021-07-13	144.029999	147.460007	143.630005	145.639999	144.812653	100827100
2	2021-07-14	148.100006	149.570007	147.679993	149.149994	148.302704	127050800
3	2021-07-15	149.240005	150.000000	147.089996	148.479996	147.636520	106820300
4	2021-07-16	148.460007	149.759995	145.880005	146.389999	145.558380	93251400

iii. **Data preprocessing**

```
# Dropping the Prediction Row

X = np.array(df.drop(['Prediction'], axis = 1))
X = X[:-predict_days]      # Size upto predict days
# print(X)
print(X.shape)
```

```
(221, 6)
```

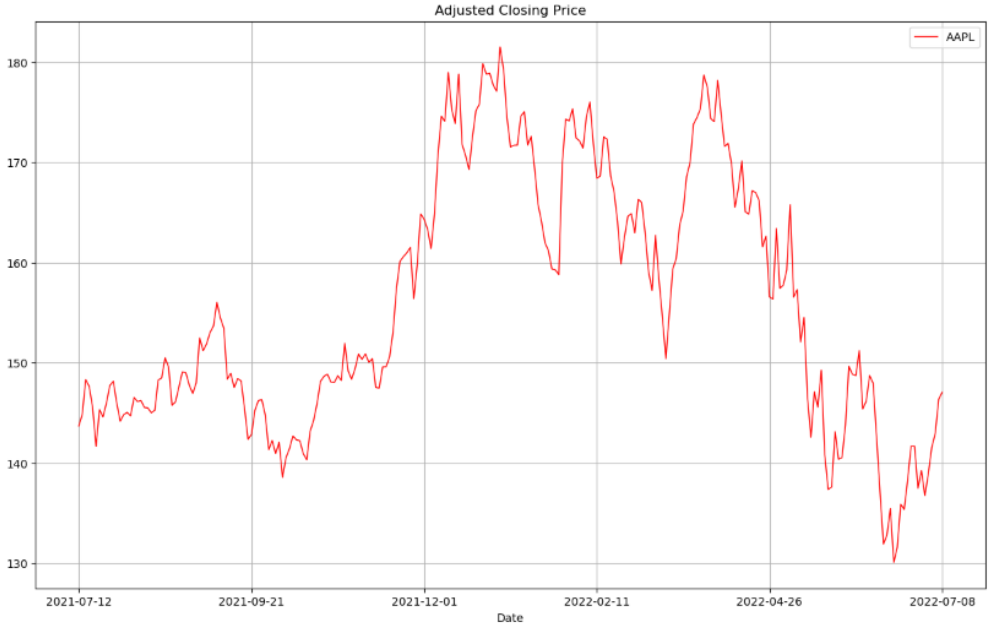
```
# Creating the Prediction Row

y = np.array(df['Prediction'])
y = y[:-predict_days]      # Size upto predict_days
# print(y)
print(y.shape)
```

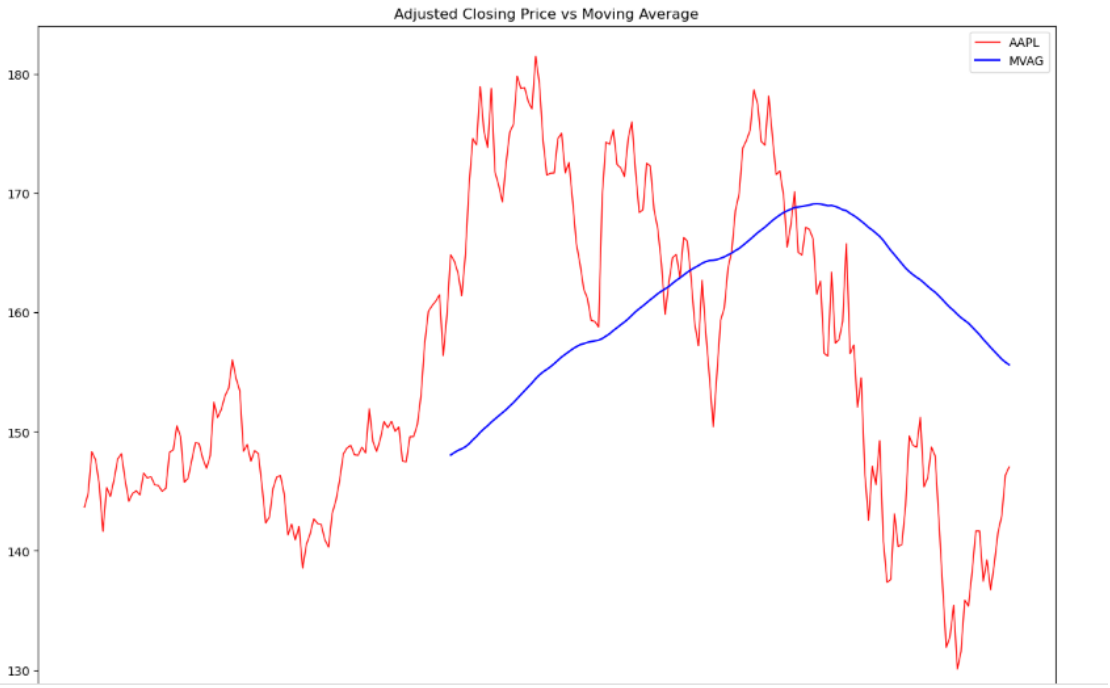
```
(221,)
```

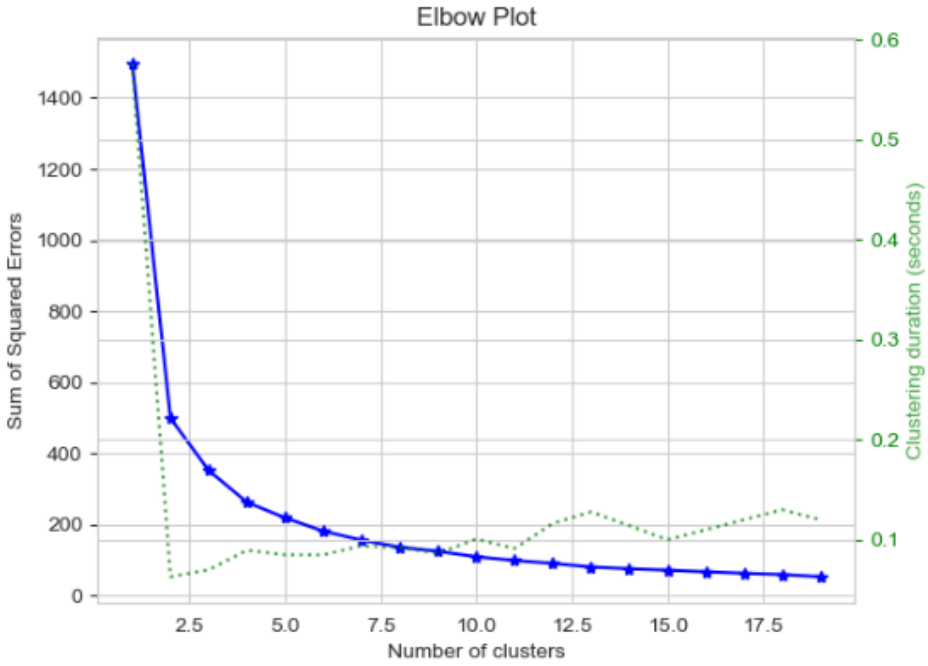
```
# Splitting the data into Training data & Testing data

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)      #Splitting the data into 80% for training & 20% for testing
print(X_train.shape)
print(y_train.shape)
print(X_test.shape)
print(y_test.shape)
```

```
(176, 6)
(176,)
```

iv. **Feature Extraction**







v. Training and Testing

```
# Creating the Prediction Row

y = np.array(df['Prediction'])
y = y[:-predict_days]      # Size upto predict_days
# print(y)
print(y.shape)
```

```
(221,)
```

```
# Splitting the data into Training data & Testing data

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)      #Splitting the data into 80% for training & 20% for testing
print(X_train.shape)
print(y_train.shape)
print(X_test.shape)
print(y_test.shape)
```

```
(176, 6)
(176,)
(45, 6)
(45,)
```

vi. **Evaluation model**

1. Random forest Regression

Building First model of regression - Linear Regression

```
[ ] from sklearn.ensemble import RandomForestRegressor
    regressor = RandomForestRegressor(n_estimators=100, random_state=0)
    regressor.fit(X_train, y_train)
```

```
RandomForestRegressor(random_state=0)
```

```
[ ] regressor_score = regressor.score(X_test, y_test)
    print('regressor score:', regressor_score)
```

```
regressor score: 0.1737670708094945
```

```
[ ] X_predict = np.array(df.drop(['Prediction'], 1))[-predict_days:]

    regressor_predict_prediction = regressor.predict(X_predict)
    regressor_real_prediction = regressor.predict(np.array(df.drop(['Prediction'], 1)))
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_8692\3361663058.py:1: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'l
  X_predict = np.array(df.drop(['Prediction'], 1))[-predict_days:]
C:\Users\admin\AppData\Local\Temp\ipykernel_8692\3361663058.py:4: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'l
  regressor_real_prediction = regressor.predict(np.array(df.drop(['Prediction'], 1)))
```

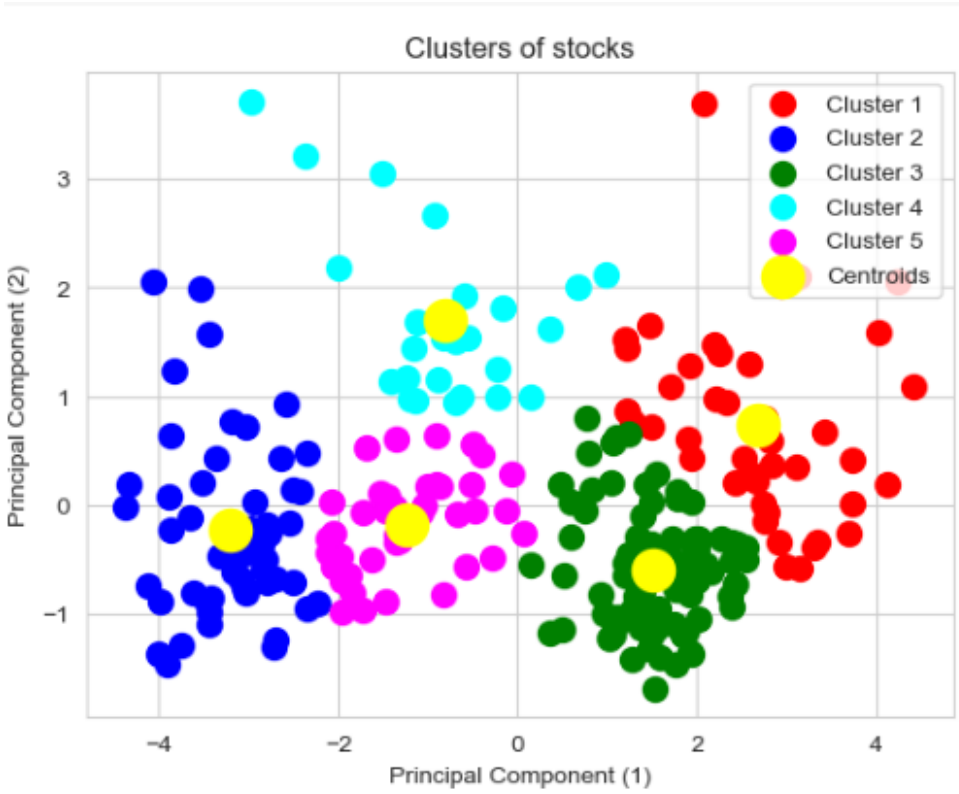
Kmean Algorithm

```
[ ] k_means = KMeans(n_clusters=5,random_state=0,init='k-means++')
    k_means.fit(X_train)
    y_kmeans = k_means.fit_predict(X_train)
    labels = k_means.labels_
```

```
[ ] len(labels), X_train.shape
```

```
(251, (251, 2))
```

```
[ ] plt.scatter(X_transformed[y_kmeans == 0, 0], X_transformed[y_kmeans == 0, 1], s = 100, c = 'red', label = 'Cluster 1')
    plt.scatter(X_transformed[y_kmeans == 1, 0], X_transformed[y_kmeans == 1, 1], s = 100, c = 'blue', label = 'Cluster 2')
    plt.scatter(X_transformed[y_kmeans == 2, 0], X_transformed[y_kmeans == 2, 1], s = 100, c = 'green', label = 'Cluster 3')
    plt.scatter(X_transformed[y_kmeans == 3, 0], X_transformed[y_kmeans == 3, 1], s = 100, c = 'cyan', label = 'Cluster 4')
    plt.scatter(X_transformed[y_kmeans == 4, 0], X_transformed[y_kmeans == 4, 1], s = 100, c = 'magenta', label = 'Cluster 5')
    plt.scatter(k_means.cluster_centers_[:, 0], k_means.cluster_centers_[:, 1], s = 300, c = 'yellow', label = 'Centroids')
    plt.title('Clusters of stocks')
    plt.xlabel('Principal Component (1)')
    plt.ylabel('Principal Component (2)')
    plt.legend()
    plt.show()
```



8.TESTING

8.1 Testing

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software Testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks at implementation of the software. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs.

Software Testing can also be stated as the process of validating and verifying that a software program/application/product:

- Meets the business and technical requirements that guided its design and Development.

- Works as expected and can be implemented with the same characteristics.

TESTING METHODS

Functional Testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- Functions: Identified functions must be exercised.
- Output: Identified classes of software outputs must be exercised.
- Systems/Procedures: system should work properly

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

Test Case for Excel Sheet Verification:

Here in machine learning we are dealing with dataset which is in excel sheet format so if any test case we need means we need to check excel file. Later on classification will work on the respective columns of dataset .

Test Case 1 :

SL #	TEST CASE NAME	DESCRIPTION	STEP NO	ACTION TO BE TAKEN (DESIGN STEPS)	EXPECTED (DESIGN STEP)	Test Execution Result (PASS/FAIL)
1	Excel Sheet verification	**Objective:** There should be an excel sheet. Any number of rows can be added to the sheet.	Step 1	Excel sheet should be available	Excel sheet is available	Pass
			Step 2	Excel sheet is created based on the template	The excel sheet should always be based on the template	Pass
			Step 3	Changed the name of excel sheet	Should not make any modification on the name of excel sheet	Fail
			Step 4	Added 10000 or above records	Can add any number of records	Pass

Domain Specification

MACHINE LEARNING

Machine Learning is a system that can learn from example through self-improvement and without being explicitly coded by programmer. The breakthrough comes with the idea that a machine can singularly learn from the data (i.e., example) to produce accurate results.

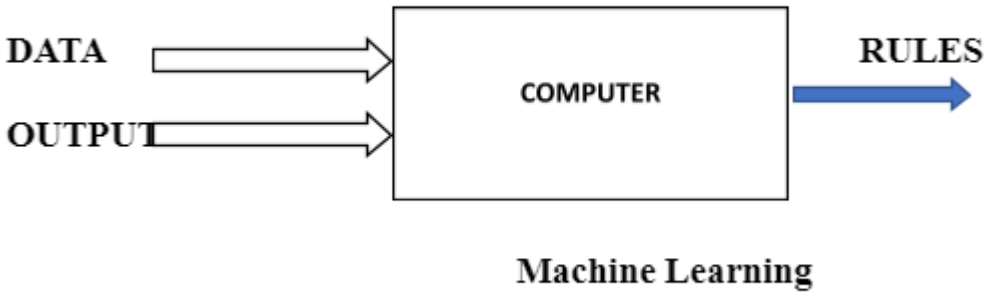
Machine learning combines data with statistical tools to predict an output. This output is then used by corporate to makes actionable insights. Machine learning is closely related to data mining and Bayesian predictive modeling. The machine receives data as input, use an algorithm to formulate answers.

A typical machine learning tasks are to provide a recommendation. For those who have a Netflix account, all recommendations of movies or series are based on the user's historical data. Tech companies are using unsupervised learning to improve the user experience with personalizing recommendation.

Machine learning is also used for a variety of task like fraud detection, predictive maintenance, portfolio optimization, automatize task and so on.

Machine Learning vs. Traditional Programming

Traditional programming differs significantly from machine learning. In traditional programming, a programmer code all the rules in consultation with an expert in the industry for which software is being developed. Each rule is based on a logical foundation; the machine will execute an output following the logical statement. When the system grows complex, more rules need to be written. It can quickly become unsustainable to maintain.



Machine Learning

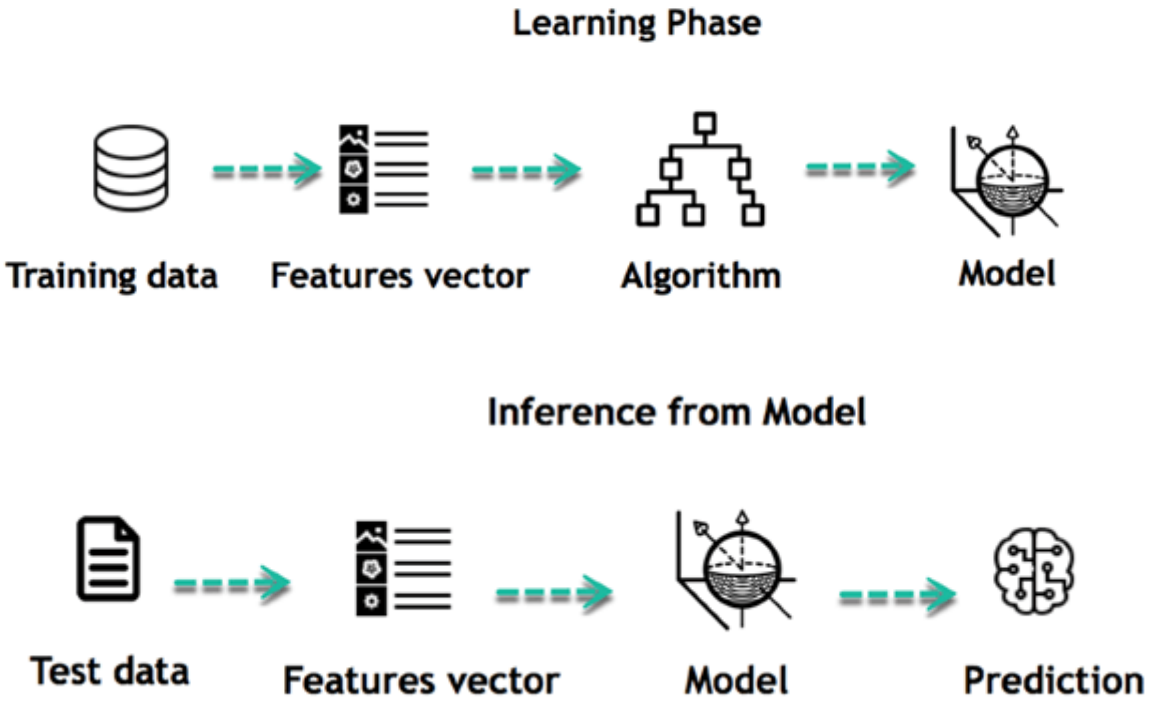
Machine Learning

How does Machine learning work?

Machine learning is the brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if its feed a previously unseen example, the machine has difficulties to predict.

The core objective of machine learning is the **learning** and **inference**. First of all, the machine learns through the discovery of patterns. This discovery is made thanks to the **data**. One crucial part of the data scientist is to choose carefully which data to provide to the machine. The list of attributes used to solve a problem is called a **feature vector.** You can think of a feature vector as a subset of data that is used to tackle a problem.

The machine uses some fancy algorithms to simplify the reality and transform this discovery into a **model**. Therefore, the learning stage is used to describe the data and summarize it into a model.



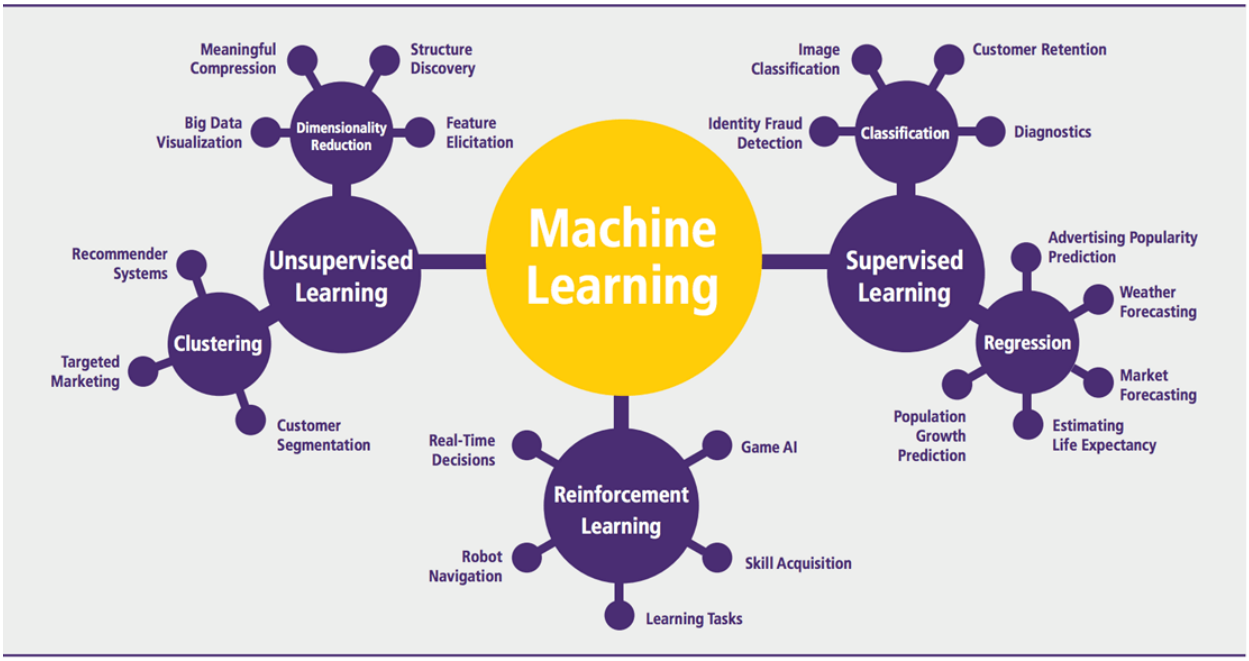
The life of Machine Learning programs is straightforward and can be summarized in the following points:

- Define a question
- Collect data
- Visualize data
- Train algorithm
- Test the Algorithm
- Collect feedback
- Refine the algorithm

- Loop 4-7 until the results are satisfying
- Use the model to make a prediction

Once the algorithm gets good at drawing the right conclusions, it applies that knowledge to new sets of data.

Machine learning Algorithms and where they are used?



Machine learning can be grouped into two broad learning tasks: Supervised and Unsupervised. There are many other algorithms

Supervised learning

An algorithm uses training data and feedback from humans to learn the relationship of given inputs to a given output. For instance, a practitioner can use marketing expense and weather forecast as input data to predict the sales of cans.

You can use supervised learning when the output data is known. The algorithm will predict new data.

There are two categories of supervised learning:

Algorithm Name	Description	Type
Linear regression	Finds a way to correlate each feature to the output to help predict future values.	Regression
Logistic regression	Extension of linear regression that's used for classification tasks. The output variable 3is binary (e.g., only black or white) rather than continuous (e.g., an infinite list of potential colors)	Classification
Decision tree	Highly interpretable classification or regression model that splits data-feature values into branches at decision nodes (e.g., if a feature is a color, each possible color becomes a new branch) until a final decision output is made	Regression Classification
Naive Bayes	The Bayesian method is a classification method that makes use of the Bayesian theorem. The theorem updates the prior knowledge of an event with the independent probability of each feature that can affect the event.	Regression Classification
Support vector machine	Support Vector Machine, or SVM, is typically used for the classification task. SVM algorithm finds a hyperplane that	Regression (not very common) Classification

	optimally divided the classes. It is best used with a non-linear solver.	
Random forest	The algorithm is built upon a decision tree to improve the accuracy drastically. Random forest generates many times simple decision trees and uses the 'majority vote' method to decide on which label to return. For the classification task, the final prediction will be the one with the most vote; while for the regression task, the average prediction of all the trees is the final prediction.	Regression Classification
AdaBoost	Classification or regression technique that uses a multitude of models to come up with a decision but weighs them based on their accuracy in predicting the outcome	Regression Classification
Gradient-boosting trees	Gradient-boosting trees is a state-of-the-art classification/regression technique. It is focusing on the error committed by the previous trees and tries to correct it.	Regression Classification

- Classification task
- Regression task

Classification

Imagine you want to predict the gender of a customer for a commercial. You will start gathering data on the height, weight, job, salary, purchasing basket, etc. from your customer database. You know the gender of each of your customer, it can only be male or female. The objective of the classifier will be to assign a probability of being a male or a female (i.e., the label) based on the information (i.e., features you have collected). When the model learned how to recognize male or female, you can use new data to make a prediction. For instance, you just got new information from an unknown customer, and you want to know if it is a male or female. If the classifier predicts male = 70%, it means the algorithm is sure at 70% that this customer is a male, and 30% it is a female.

The label can be of two or more classes. The above example has only two classes, but if a classifier needs to predict object, it has dozens of classes (e.g., glass, table, shoes, etc. each object represents a class)

Regression

When the output is a continuous value, the task is a regression. For instance, a financial analyst may need to forecast the value of a stock based on a range of feature like equity, previous stock performances, macroeconomics index. The system will be trained to estimate the price of the stocks with the lowest possible error.

Unsupervised learning

In unsupervised learning, an algorithm explores input data without being given an explicit output variable (e.g., explores customer demographic data to identify patterns)

You can use it when you do not know how to classify the data, and you want the algorithm to find patterns and classify the data for you

Algorithm	Description	Type
K-means clustering	Puts data into some groups (k) that each contains data with similar characteristics (as determined by the model, not in advance by humans)	Clustering

Gaussian mixture model	A generalization of k-means clustering that provides more flexibility in the size and shape of groups (clusters	Clustering
Hierarchical clustering	Splits clusters along a hierarchical tree to form a classification system. Can be used for Cluster loyalty-card customer	Clustering
Recommender system	Help to define the relevant data for making a recommendation.	Clustering
PCA/T-SNE	Mostly used to decrease the dimensionality of the data. The algorithms reduce the number of features to 3 or 4 vectors with the highest variances.	Dimension Reduction

Application of Machine learning
Augmentation:

Machine learning, which assists humans with their day-to-day tasks, personally or commercially without having complete control of the output. Such machine learning is used in different ways such as Virtual Assistant, Data analysis, software solutions. The primary user is to reduce errors due to human bias.

Automation:

Machine learning, which works entirely autonomously in any field without the need for any human intervention. For example, robots performing the essential process steps in manufacturing plants.

Finance Industry

Machine learning is growing in popularity in the finance industry. Banks are mainly using ML to find patterns inside the data but also to prevent fraud.

Government organization

The government makes use of ML to manage public safety and utilities. Take the example of China with the massive face recognition. The government uses Artificial intelligence to prevent jaywalker.

Healthcare industry

Healthcare was one of the first industry to use machine learning with image detection.

Marketing

Broad use of AI is done in marketing thanks to abundant access to data. Before the age of mass data, researchers develop advanced mathematical tools like Bayesian analysis to estimate the value of a customer. With the boom of data, marketing department relies on AI to optimize the customer relationship and marketing campaign.

Example of application of Machine Learning in Supply Chain

Machine learning gives terrific results for visual pattern recognition, opening up many potential applications in physical inspection and maintenance across the entire supply chain network.

Unsupervised learning can quickly search for comparable patterns in the diverse dataset. In turn, the machine can perform quality inspection throughout the logistics hub, shipment with damage and wear.

For instance, IBM's Watson platform can determine shipping container damage. Watson combines visual and systems-based data to track, report and make recommendations in real-time.

In past year stock manager relies extensively on the primary method to evaluate and forecast the inventory. When combining big data and

machine learning, better forecasting techniques have been implemented (an improvement of 20 to 30 % over traditional forecasting tools). In term of sales, it means an increase of 2 to 3 % due to the potential reduction in inventory costs.

Example of Machine Learning Google Car

For example, everybody knows the Google car. The car is full of lasers on the roof which are telling it where it is regarding the surrounding area. It has radar in the front, which is informing the car of the speed and motion of all the cars around it. It uses all of that data to figure out not only how to drive the car but also to figure out and predict what potential drivers around the car are going to do. What's impressive is that the car is processing almost a gigabyte a second of data.

Deep Learning

Deep learning is a computer software that mimics the network of neurons in a brain. It is a subset of machine learning and is called deep learning because it makes use of deep neural networks. The machine uses different layers to learn from the data. The depth of the model is represented by the number of layers in the model. Deep learning is the new state of the art in term of AI. In deep learning, the learning phase is done through a neural network.

Reinforcement Learning

Reinforcement learning is a subfield of machine learning in which systems are trained by receiving virtual "rewards" or "punishments," essentially learning by trial and error. Google's DeepMind has used reinforcement learning to beat a human champion in the Go games. Reinforcement learning is also used in video games to improve the gaming experience by providing smarter bot.

One of the most famous algorithms are:

- Q-learning
- Deep Q network
- State-Action-Reward-State-Action (SARSA)
- Deep Deterministic Policy Gradient (DDPG)

Applications/ Examples of deep learning applications

AI in Finance: The financial technology sector has already started using AI to save time, reduce costs, and add value. Deep learning is changing the lending industry by using more robust credit scoring. Credit decision-makers can use AI for robust credit lending applications to achieve faster, more accurate risk assessment, using machine intelligence to factor in the character and capacity of applicants.

Underwrite is a Fintech company providing an AI solution for credit makers company. underwrite.ai uses AI to detect which applicant is more likely to pay back a loan. Their approach radically outperforms traditional methods.

AI in HR: Under Armour, a sportswear company revolutionizes hiring and modernizes the candidate experience with the help of AI. In fact, Under Armour Reduces hiring time for its retail stores by 35%. Under Armour faced a growing popularity interest back in 2012. They had, on average, 30000 resumes a month. Reading all of those applications and begin to start the screening and interview process was taking too long. The lengthy process to get people hired and on-boarded impacted Under Armour's ability to have their retail stores fully staffed, ramped and ready to operate.

At that time, Under Armour had all of the 'must have' HR technology in place such as transactional solutions for sourcing, applying, tracking and onboarding but those tools weren't useful enough. Under armour choose **HireVue**, an AI provider for HR solution, for both on-demand and live interviews. The results were bluffing; they managed to decrease by 35% the time to fill. In return, the hired higher quality staffs.

AI in Marketing: AI is a valuable tool for customer service management and personalization challenges. Improved speech recognition in call-center management and call routing as a result of the application of AI techniques allows a more seamless experience for customers.

For example, deep-learning analysis of audio allows systems to assess a customer's emotional tone. If the customer is responding poorly to the AI chatbot, the system can be rerouted the conversation to real, human operators that take over the issue.

Difference between Machine Learning and Deep Learning

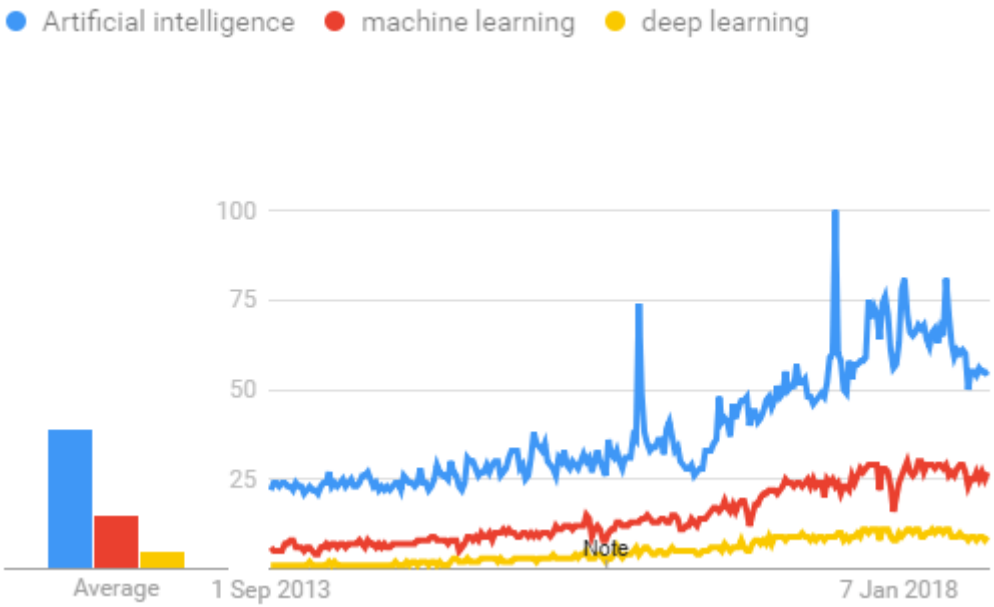
	Machine Learning	Deep Learning
Data Dependencies	Excellent performances on a small/medium dataset	Excellent performance on a big dataset
Hardware dependencies	Work on a low-end machine.	Requires powerful machine, preferably with GPU: DL performs a significant amount of matrix multiplication
Feature engineering	Need to understand the features that represent the data	No need to understand the best feature that represents the data
Execution time	From few minutes to hours	Up to weeks. Neural Network needs to compute a significant number of weights
Interpretability	Some algorithms are easy to interpret (logistic, decision tree), some are almost impossible (SVM, XGBoost)	Difficult to impossible

When to use ML or DL?

In the table below, we summarize the difference between machine learning and deep learning.

	Machine learning	Deep learning
Training dataset	Small	Large
Choose features	Yes	No
Number of algorithms	Many	Few
Training time	Short	Long

With machine learning, you need fewer data to train the algorithm than deep learning. Deep learning requires an extensive and diverse set of data to identify the underlying structure. Besides, machine learning provides a faster-trained model. Most advanced deep learning architecture can take days to a week to train. The advantage of deep learning over machine learning is it is highly accurate. You do not need to understand what features are the best representation of the data; the neural network learned how to select critical features. In machine learning, you need to choose for yourself what features to include in the model.



TensorFlow

the most famous deep learning library in the world is Google's TensorFlow. Google product uses machine learning in all of its products to improve the search engine, translation, image captioning or recommendations.

To give a concrete example, Google users can experience a faster and more refined the search with AI. If the user types a keyword a the search bar, Google provides a recommendation about what could be the next word.

Google wants to use machine learning to take advantage of their massive datasets to give users the best experience. Three different groups use machine learning:

- Researchers
- Data scientists
- Programmers.

They can all use the same toolset to collaborate with each other and improve their efficiency.

Google does not just have any data; they have the world's most massive computer, so TensorFlow was built to scale. TensorFlow is a library developed by the Google Brain Team to accelerate machine learning and deep neural network research.

It was built to run on multiple CPUs or GPUs and even mobile operating systems, and it has several wrappers in several languages like Python, C++ or Java.

In this tutorial, you will learn

TensorFlow Architecture

Tensor flow architecture works in three parts:

- Pre processing the data
- Build the model
- Train and estimate the model

It is called Tensor flow because it takes input as a multi-dimensional array, also known as **tensors**. You can construct a sort of **flowchart** of operations (called a Graph) that you want to perform on that input. The input goes in at one end, and then it flows through this system of multiple operations and comes out the other end as output.

This is why it is called TensorFlow because the tensor goes in it flows through a list of operations, and then it comes out the other side.

Where can Tensor flow run?

TensorFlow can hardware, and software requirements can be classified into

Development Phase: This is when you train the mode. Training is usually done on your Desktop or laptop.

Run Phase or Inference Phase: Once training is done Tensorflow can be run on many different platforms. You can run it on

- Desktop running Windows, macOS or Linux
- Cloud as a web service
- Mobile devices like iOS and Android

You can train it on multiple machines then you can run it on a different machine, once you have the trained model.

The model can be trained and used on GPUs as well as CPUs. GPUs were initially designed for video games. In late 2010, Stanford researchers found that GPU was also very good at matrix operations and algebra so that it makes them very fast for doing these kinds of calculations. Deep learning relies on a lot of matrix multiplication. TensorFlow is very fast at

computing the matrix multiplication because it is written in C++. Although it is implemented in C++, TensorFlow can be accessed and controlled by other languages mainly, Python.

Finally, a significant feature of Tensor Flow is the Tensor Board. The Tensor Board enables to monitor graphically and visually what TensorFlow is doing.

List of Prominent Algorithms supported by TensorFlow

- Linear regression: tf. estimator .Linear Regressor
- Classification :tf. Estimator .Linear Classifier
- Deep learning classification: tf. estimator. DNN Classifier
- Booster tree regression: tf.estimator.BoostedTreesRegressor
- Boosted tree classification: tf.estimator.BoostedTreesClassifier

PYTHON OVERVIEW

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted:** Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive:** You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented:** Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language:** Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, Unix shell, and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Python Features

Python's features include:

Easy-to-learn: Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
Easy-to-read: Python code is more clearly defined and visible to the eyes.
Easy-to-maintain: Python's source code is fairly easy-to-maintain.
A broad standard library: Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
Interactive Mode: Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
Portable: Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
Extendable: You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
Databases: Python provides interfaces to all major commercial databases.
GUI Programming: Python supports GUI applications that can be created and ported to many system calls, libraries, and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
Scalable: Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below:

- IT supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python is available on a wide variety of platforms including Linux and Mac OS X. Let's understand how to set up our Python environment.

ANACONDA NAVIGATOR

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows you to launch applications and easily manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository. It is available for Windows, mac OS and Linux.

Why use Navigator?

In order to run, many scientific packages depend on specific versions of other packages. Data scientists often use multiple versions of many packages, and use multiple environments to separate these different versions.

The command line program conda is both a package manager and an environment manager, to help data scientists ensure that each version of each package has all the dependencies it requires and works correctly.

Navigator is an easy, point-and-click way to work with packages and environments without needing to type conda commands in a terminal

window. You can use it to find the packages you want, install them in an environment, run the packages and update them, all inside Navigator.

WHAT APPLICATIONS CAN I ACCESS USING NAVIGATOR?

The following applications are available by default in Navigator:

- Jupyter Lab
- Jupyter Notebook
- QT Console
- Spyder
- VS Code
- Glue viz
- Orange 3 App
- Rodeo
- RStudio

Advanced conda users can also build your own Navigator applications

How can I run code with Navigator?

The simplest way is with Spyder. From the Navigator Home tab, click Spyder, and write and execute your code.

You can also use Jupyter Notebooks the same way. Jupyter Notebooks are an increasingly popular system that combine your code, descriptive text, output, images and interactive interfaces into a single notebook file that is edited, viewed and used in a web browser.

What's new in 1.9?

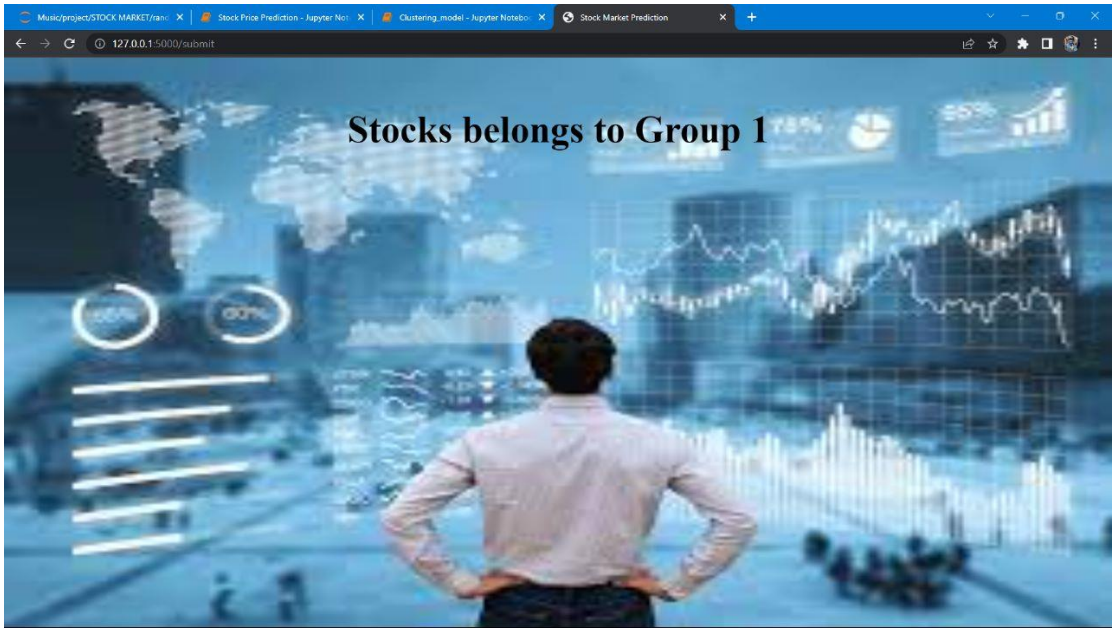
- Add support for **Offline Mode** for all environment related actions.
- Add support for custom configuration of main windows links.

Numerous bug fixes and performance enhancements.

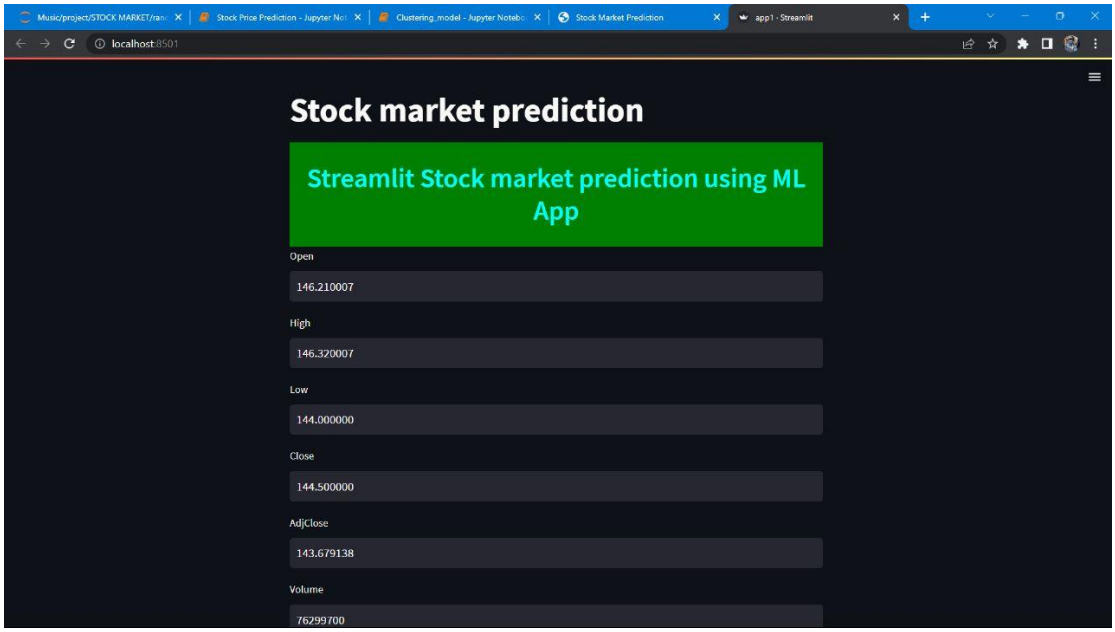
9.OUTPUT

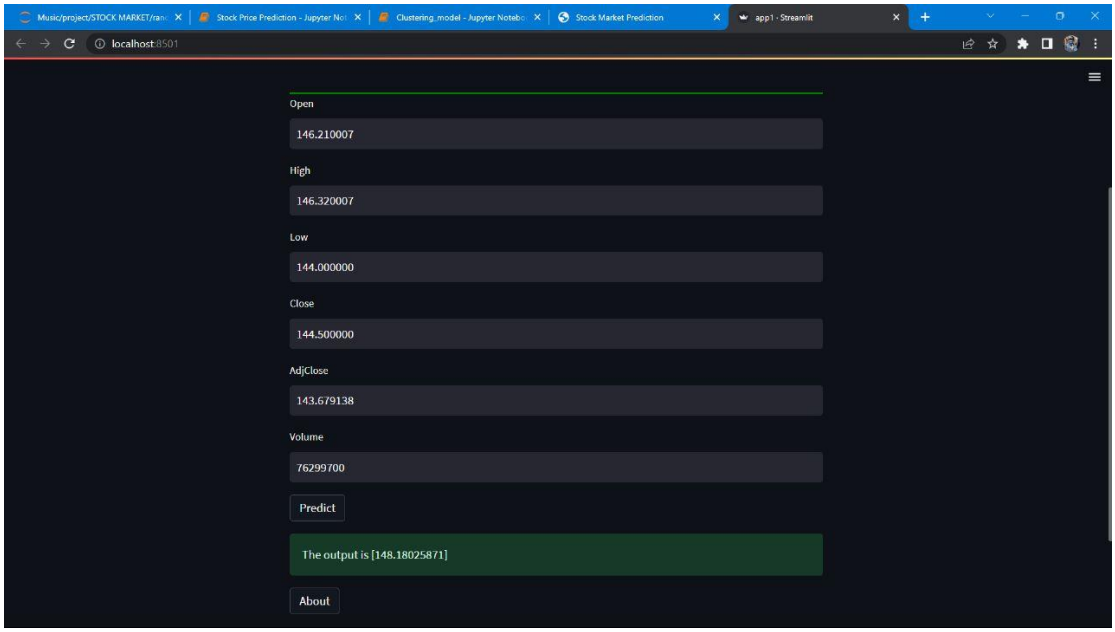
Screenshots :

K-MEANS CLUSTERING :



RANDOM FOREST ALGORITHM :





CONCLUSION:

cluster analysis is used to classify objects into groups where objects in one group are more similar to each other and different from objects in other groups. It is normally used for exploratory data analysis and as a method of discovery by solving classification issues.

In the present work clustering is performed on stock data obtained which produces the name of the best companies as output. Then comparison between partitioning based, hierarchical, model based and density based techniques are performed with the help of validation index such as c-index, Jaccard index, rand index and silhouette index. K-means algorithm in partitioning based technique and EM algorithm in model based technique shows better performance than hierarchical and density based technique. Then the clustered result is given to multiple regression which is one of the regression technique for getting the future stock price.

FUTURE ENHANCEMENTS:

Future work can be considered as using traditional time series analysis to perform forecast price of stock markets. It is an area of continuous research as investors and researchers strive to work with the market with an ultimate reason of acquiring higher returns. It is so unlikely that the new theoretical results will come out with the above projected works.

10. REFERENCES

[1] Madge, Saahil, and Swati Bhatt. "Predicting stock price direction using support vector machines." Independent work report spring, 2015.

[2] Hegazy, Osman, Omar S. Soliman, and Mustafa Abdul Salam. "A machine learning model for stock market prediction." arXiv preprint arXiv:1402.7351,2014

[3] Lakshminarayanan, Sai Krishna, and John McCrae. "A Comparative Study of SVM and LSTM Deep Learning Algorithms for Stock Market Prediction." In AICS, pp. 446-457. 2019.

[4] E. Studies, "A Stock Market Prediction Method Based on Support Vector Machines (SVM) and Independent Component Analysis (ICA)," pp. 12-21, 2000.

[5] M. A. Ghazanfar, S. A. Alahmari, Y. Fahad, A. Mustaqeem, and M. A. Azam, "Using Machine Learning Classifiers to Predict Stock Exchange Index," vol. 7, no. 2, pp. 24-29, doi: 10.18178/ijmlc.2017.7.2.614.

[6] Shah, Dev, Haruna Isah, and Farhana Zulkernine. "Stock market analysis: A review and taxonomy of prediction techniques." International Journal of Financial Studies vol 7, no. 2 pp 26., 2019

[7] S. Jyothirmayee, V. D. Kumar, C. S. Rao, and R. S. Shankar, "Predicting Stock Exchange using Supervised Learning Algorithms," no. 1, pp. 4081-4090, doi: 10.35940/ijitee.A4144.119119.

[8] H. Snyder, "Literature review as a research methodology: An overview and guidelines," J. Bus. Res., vol. 104, no. 2019, pp. 333-339, doi: 10.1016/j.jbusres.2019.07.039.

[9] N. S. Selamat, "An Overview of Big Data Usage in Disaster Management," vol. 11, no. 1, pp. 35-40.

[10] R. P. Schumaker and H. Chen, "Textual Analysis of Stock Market Prediction Using Breaking Financial News : The AZFinText System," no. 2006, pp. 1 -29.

[11] H. L. Siew, "Regression Techniques for the Prediction of Stock Price Trend," doi: 10.1109/ICSSBE.2012.6396535.

[12] M. K. A. Shatnawi, " Stock Price Prediction Using K -Nearest Neighbor
(k NN) Algorithm," vol. 3, no. 3, pp. 32-44, 2009.

[13] C. Hargreaves, "Prediction of Stock Performance Using Analytical

Techniques," vol. 5, no. 2, pp. 136-142, doi: 10.4304/jetwi.5.2.136-142.
[14] Magaji, A.S,"An intense Nigerian stock exchange market prediction using logistic with back-propagation ANN model", Science World Journal, 9(2), pp.8-13, 2014.
[15] Shashaank, D. S., V. Sruthi, M. L. Vijayalakshimi, and Jacob Shomona
Garcia. "Turnover prediction of shares using data mining techniques: a case study." arXiv preprint arXiv:1508.00088 ,2015.
[16] Khan, W., M. A. Ghazanfar, M. Asam, A. Iqbal, S. Ahmad, and Javed Ali Khan. "PREDICTING TREND IN STOCK MARKET EXCHANGE USING MACHINE LEARNING CLASSIFIERS." Science International 28, no. 2 ,2016.
[17] Zaidi, M., and A. Amirat. "Forecasting stock market trends by logistic regression and neural networks: Evidence from KSA stock market." Int. J. Econ. Commer. Manag 4,vol. IV, no. 6, pp. 220-234, 2016
[18] A. Nayak, M. M. M. Pai, and R. M. Pai, "Prediction Models for Indian Stock Market," Procedia - Procedia Comput. Sci., vol. 89, pp. 441-449, 1877, doi: 10.1016/j.procs.2016.06.096.
[19] H. Karchalkar, A. Jain, A. Singh, and V. Kumar, "Open Price Prediction of Stock Market using Regression Analysis," vol. 6, no. 5, pp. 418-421, 2007, doi: 10.17148/IJARCCE.2017.6578.
[20] Y. Yaslan, " Stock daily return prediction using expanded features and feature selection," no. 2017, pp. 4829-4840, doi: 10.3906/elk-1704-256.
[21] A. You, M. A. Y. Be, and I. In, "Application of SVM-KNN using SVR
as feature selection on stock analysis for Indonesia stock exchange Application of SVM-KNN Using SVR as Feature Selection on Stock Analysis for Indonesia Stock Exchange," vol. 020207, no. 2018

11. APPENDIX 1

RELEVANCE OF PROJECT TO POs / PSOs

Title of the project	Stock market prediction
Implementation Details	Python flask
Cost	-N.A-
Type	Website

Mapping with POs and PSOs with Justification														
Relevance	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
	2	2	3	-	3	-	-	2	3	2	-	3	2	-
Program Outcomes Justification	PO1: Engineering Knowledge: SDLC phases are followed in the execution of the project. PO2: Problem Analysis: The different steps involved in Problem Analysis for formulation of the solution i.e. literature survey and use of fundamental subject knowledge has been followed. PO3: Design/Development of solutions – Existing strategy has been enhanced using the design principles. PO5: Modern Tool Usage: K-MEANS and RANDOM FOREST Algorithm is used. PO8: Ethics: Students have followed professional ethics during the various stages of Project completion. PO9: Individual and Team Work: Students have worked both in individual as well as team capacity during the various stages of project work. PO10: Communication: Effective communication with team members and during project reviews, project seminar and viva-voce has been exhibited. PO12: Lifelong Learning. The project carried out gives the students scope to continue the work in the domain of Machine learning in future.													

11. APPENDIX 2

STOCK MARKET PREDICTION

GANTT CHART