STOCK MARKET PRICE PREDICTION USING MACHINE LEARNING TECHNIQUES UTILISING REAL TIME DATA

A PROJECT REPORT

Submitted by

RAKSHANA H **[REGISTER NO:211419104213]**

SWATHI G **[REGISTER NO:211419104280]**

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

APRIL 2023

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **“STOCK MARKET PRICE PREDICTION USING MACHINE LEARNING TECHNIQUES UTILISING REAL TIME DATA”** is the bonafide work of “**RAKSHANA H (211419104213) , SWATHI (211419104280) ”** who carried out the project work under my supervision.

SIGNATURE

Dr.L.JABASHEELA,M.E.,Ph.D.,
HEAD OF THE DEPARTMENT

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

SIGNATURE

Dr.K.VALARMATHI,M.E,Ph.D.,
SUPERVISOR,PROFESSOR

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

Certified that the above candidate(s) was/ were examined in the End Semester Project Viva-Voce Examination held on...........................

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We **RAKSHANA H (211419104213) , SWATHI G (211419104280)** hereby declare that this project report titled "**STOCK MARKET PRICE PREDICTION USING MACHINE LEARNING TECHNIQUES UTILISING REAL TIME DATA**" , under the guidance of **Dr.K.VALARMATHI M.E,Ph.D.,** is the original work done by us and we have not plagiarised or submitted to any other degree in any university by us.

RAKSHANA H

SWATHI G

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our beloved Directors **Tmt.C.VIJAYA RAJESWARI**, **Dr.C.SAKTHI KUMAR,M.E.,Ph.D** and **Dr.SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. L.JABASHEELA,M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank my **Dr.K.VALARMATHI, M.E,Ph.D.,** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

RAKSHANA H

SWATHI G

ABSTRACT

Stock price prediction is the process of predicting future stock prices utilizing data and a variety of market factors. It involves applying statistical models and machine learning approaches to analyze financial data to predict a stock's future performance. Our prediction attempts to help investors make sensible investment decisions by providing a forecast of future stock prices using real-time auto-updated information in contrast to utilizing an existing data collection. It contains information from different companies that is automatically included into the model without any intervention. This considerably contributes to improving the model's accuracy. The user's input, which includes stock's high, low, open value, and volume aids in accurate price prediction for the upcoming.

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

1. INTRODUCTION

1.1 PROBLEM DEFINITION

The global economy, healthcare system, oil prices, interest rates, news stories, and public opinion are only a few micro and macro-factors that have an impact on stock values. Forecasting stock prices is a significant undertaking for financial businesses, and informed projections may reduce market risks and provide significant benefits. Making the most accurate forecasts and models based on the presence of several components has been the focus of numerous articles and research. With the enormous potential for profit that comes along with stock price forecasts, the intricacy of the subject has made it a difficult one. As a result, several articles and research have attempted to produce the most precise predictions and models possible. In high-frequency trading, there is a large volume of orders, proprietary trading, and a short retention period. Data on the stock market is regarded as time series data since it is produced on a regular basis. Data on the stock market is a collection of time-ordered data points linked to one or more time-dependent factors. The movement of prices on a chart creates local and global patterns that serve as the foundation for technical analysis. You can categorize time series as multivariate or univariate. Whereas multivariate models take into account numerous factors, univariate time-series models simply take into account one dependent variable. A univariate time series model is trained only on historical price changes. Univariate predictive models compress this complexity to a single component and disregard all other dimensions, despite the fact that the present stock price is affected by several factors, such as the closing or starting price. Several technical indicators, daily highs and lows, moving averages, the connection between closing and starting prices, and other aspects are all taken into consideration by multivariate time-series forecasting models. When working with stock market data, a number of time-series elements, such as trends, cyclical swings, seasonal patterns, and random volatility, may help estimate stock prices more accurately. Long-term effects produce trends, which have the potential to change the time-series value over time. Periodic swings are intended to capture short- to medium-term price increases in stocks and occur throughout the course of a time series. With hard to reproduce time series like COVID'19, Henrique's et al. | bioRχiv | March 21, 2023 | DRAFT irregular motions show quick changes. Financial experts and researchers have had difficulty predicting stock market patterns using live-streaming data. When compared to conventional processing tools, which store and handle data in batches, a streaming data method is different. The following is a list of the most common questions that we are asked about our services. Yet, because of the market's complexity and chaotic dynamics, as well as the many non stationary, undecidable, and unexpected components involved, making decisions is difficult.

2. LITERATURE SURVEY

2.1 Literature Survey:

In their research, Kim and Han [3] employed ANN and GA to forecast future stock prices. The writers' data came from the Korea Stock Price Index (KOSPI). The sample data they obtained from KOSPI covered a period of nearly ten years, from January 1989 to December 1998. They employed the necessary procedures and optimisation to get the data ready for usage. They enhanced ANN using GA. There were 12 non-adjustable hidden layers. Also, even though he acknowledges that GA has enormous potential for improvement, the author only concentrated on two optimisation parameters. Similar work is also provided by Shreya Pawaskar [6], who use ANN and GA to forecast the movement of Japanese stock. Given that this method combined both, Theyazn H. H. Aldhyani Ali Alzahrani called it the GA-ANN model. Hidden Markov [7] Model (HMM) was used by Hassan and Nath to forecast the stock values of four major airlines. One of the nicest aspects of their study article was the fact that the method they used did not require a model-building specialist. The issue with the study is that they employed a relatively little amount of data for assessment and that data was tied to a certain business, which may have prevented them from getting appropriate predictions. Only two years' worth of data, which is much too little for a machine to comprehend and anticipate a pattern, were employed by the authors. In [6], Shreya Pawaskar utilized SVM to forecast the direction of the stock market. The author obtained data from Taiwan Economic Journal and NASDAQ. Lee chose features using the supported sequential forward search (SSFS) technique. The author also developed a few procedures that would modify parameters with various values. They used a fairly simple framework for the model's feature selection. Long short-term memory (LSTM) was employed by Dr. Poorna Shankar, Dr. Neha Sharma, Mr. Roushan Raj and Mr. Chetan Dalwadi in [8] to forecast the stock market movement. From December 1989 to September 2015, they gathered data for Thomson Reuters from the S&P 500 index. They transformed the list into a binary matrix after gathering the data. They employed RMSprop for optimisation.They completed the assignment using the most up-to-date method, although they lacked any prior financial expertise. The author of this research omitted describing how they trained the model using long-term dependencies. Nusrat Rouf, Majid Bashir Malik, Tasleem Arif, Sparsh Sharma, Saurabh Singh, Satyabrata Aich, Hee-Cheol Kimin [10] has used the RNN-LSTM model on NIFTY-50 stocks. They collected data for 5 years and RMSE to find out the error rate. The window size they are using to predict the price movement is 21 days.

YEAR & AUTHORS	**TITLE**	**METHODOLOGY**	**MERITS AND DEMERITS**	**FUTURE SCOPE**
2022,Haocheng Du	Research on Amazon's stock price forecasting based on arbitrage pricing model based on big data	Aim to construct an arbitrage pricing model to make a regression analysis on Amazon's stock price, which is demonstrated to have a higher prediction accuracy and better fitting degree compared with the self-coding network.	**Merits:** Big data is used for amazon stock prediction by neutral networks. **Demerits:** Can be used more algorithm for increase accuracy.	Further enhancement for the proposed solution is to calculate the accuracy can do better.
2022,Shreya Pawaskar	Stock Price Prediction using Machine Learning Algorithms	Various machine learning algorithms like Multiple Linear Regression, Polynomial	**Merits:** From this journal, accuracy can be calculated.	Further enhancement for the proposed solution is to calculated the

		Regression, etc. are used.highly theoretical and speculative nature of the stock market has been examined by capturing and using repetitive patterns.	**Demerits:** In this paper Accuracy is not mentioned for the proposed solution.	accuracy can do better in deep learning algorithms.
2022, Theyazn H. H. Aldhyani Ali Alzahrani	Framework for Predicting and Modeling Stock Market Prices Based on Deep Learning Algorithms	Deep learning is a complex task and there are many factors that can affect stock prices, such as economic and political events.While Deep Learning models can be helpful in making predictions,they should not be relied upon solely for making investment decisions.	**Merits:** Compared two algorithm and then suitable algorithm is chosen for the prediction **Demerits:** More than two algorithm will can be used	In the working module deployment can be implemented in machine learning algorithms
2022, Dr. Poorna Shankar, Dr. Neha Sharma, Mr. Roushan Raj, Mr. Chetan Dalwadi	Stock Price Prediction Using LSTM, ARIMA and UCM	LSTM, RNN, ARIMA are used that takes a model in a sequence of historical data stock price and then predicts the future stock index.LSTM is used to involve more complex processes such as consolidation and retrieval through recall or recognition where LSTM &	**Merits:** Accuracy will be Calculated from the given information**.** **Demerits:** In this paper Accuracy is not mentioned for the proposed solution.	Further enhancement for the proposed solution is to calculated the accuracy can do better.

		ARIMA are types of classification of RNN		
2021, Milon Biswas, Arafat Jahan Nova, Md. Kawsher Mahbub, Sudipto Chaki, Shamim Ahmed, Md. Ashraful Islam	Stock Market Prediction: A Survey and Evaluation	Artificial Intelligence is used to survey both machine learning and deep learning techniques comparatively to predict stock market price vale.	**Merits:** The survey for Stock Market Prediction is done for prediction **Demerits:** From the Survey Stock Market Price is not predicted.	Instead of Artificial Intelligence to improve the accuracy Machine learning can be implemented.
2021, Shubha Singh, Sreedevi Gutta, Ahmad Hadaegh	Stock Prediction Using Machine Learning	Data Mining and Machine learning is used for predicting the historical data that train and suitable algorithm for the database is implemented and a separate dataset is used for testing accuracy.	**Merits:** Compared two algorithm and then suitable algorithm is chosen for the prediction **Demerits:** More than two algorithm can used for comparison for better accuracy	Further enhancement for the proposed solution is to calculated the accuracy can do better.
2021,C. C. Emioma, S. O. Edeki	Stock price prediction using machine learning on least-squares linear regression basis	RNN and SVM Different types of RNN are used comparatively SVM is implemented for prediction.this algorithm is used to train the dataset using supervised learning approach, where the model	**Merits:** Compared two algorithm and then suitable algorithm is chosen for the prediction **Demerits:** More than two algorithm can	In the working module deployment can be implemented

		learns to predict stock price.	used for comparison for better accuracy.	
2021,Sidra Mehtab, Jaydip Sen and Abhishek Dutta	Stock Price Prediction Using Machine Learning and LSTM-Based Deep Learning Models	Long and Short-Term Memory algorithm is used that takes a model in a sequence of historical data stock price and then predicts the future stock index.LSTM is used to involve more complex processes such as consolidation and retrieval through recall or recognition.	**Merits:** From the proposed solution only stock index can be predicted **Demerits:** Stock value must be predicted	More algorithms can be compared to predicted the accuracy stock price value instead of only stock index prediction
2021,Nusrat Rouf, Majid Bashir Malik, Tasleem Arif, Sparsh Sharma, Saurabh Singh, Satyabrata Aich, Hee-Cheol Kim	Stock Market Prediction Using Machine Learning Techniques: A Decade Survey on Methodologies, Recent Developments, and Future Directions	Machine learning is involve the use of recurrent neural networks (R.N.N.) and artificial neural networks (ANN), which also falls under the category of machine learning.	**Merits:** In this journal paper the prediction is for accuracy which is given as accuracy **Demerits:** From the accuracy prediction the stock market price value must be give as output.	Project can be deployed by implementation.
2021, A M Pranav, Sujooda S, Jerin Babu, Amal Chandran,Anoop S	StockClue: Stock Prediction using Machine Learning	Machine learning is used in proposed method involves determining an interactive	**Merits:** News data ,trend, is used as dataset **Demerits:** Real stock data	Real stock data will be predicted

		online platform (Web App) for stock traders to use in order to forecast future stock market values. The Web App also shows market prices, volume, and associated statistics, as well as the selected stock's prediction.	will be used	
2019,Kevin Thomas	Time Series Prediction for Stock Price and Opioid Incident Location	This work investigates two fields of time series forecasting in the form of Stock Data Prediction and the Opioid Incident Prediction.The Stock Data Prediction problem investigates methods which could predict the trends in the NYSE and NASDAQ stock markets	**Merits:** News data is used as dataset **Demerits:** Real stock data will be used	More algorithms can be compared to predicted the accuracy stock price value instead of only stock index prediction

2.2 Survey Table

3.SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Financial news disclosures provide valuable information for traders and investors while making stock market investment decisions. Essential but challenging, the stock market prediction problem has attracted significant attention from both researchers and practitioners. Conventional machine learning models often fail to interpret the content of financial news due to the complexity and ambiguity of natural language used in the news. This article presented an RNN-based ensemble model for financial market prediction via news disclosures. Sentiment analysis and the sliding window method were applied to extract the most representative features from financial news and historical data. This greatly reduced the number of dimensions compared to traditional pre-processing strategies (e.g., bag-of-words and TF-IDF) that extract tens of thousands of features.

DISADVANTAGES

- Accuracy is comparatively low.
- Historical Datasets are used which reduces the accuracy.
- Deployment is not implemented and it affects user satisfaction and convenience.

3.2 PROPOSED SYSTEM

Datasets from different sources would be combined to form a generalized dataset. In this section of the report will load in the data, check for cleanliness, and then trim and clean the given dataset for analysis. The data set collected for predicting given data is split into Training set and Test set. Generally, 7:3 ratios are applied to split the Training set and Test set. The Data Model which was created using machine learning algorithms is applied on the Training set and based on the test result accuracy, Test set prediction is done. Predicting the stock problem, ML prediction model is effective because It is strong in preprocessing outliers, irrelevant variables, and a mix of continuous, categorical and discrete variables.

ADVANTAGES

- We intend to employ real-time data obtained as packages rather than the typical way of using historical data sets.
- Accuracy may be increased by using many algorithms.
- Project will be deployed.

3.3 FEASIBILITY STUDY

DATA WRANGLING

In this section of the report will load in the data, check for cleanliness, and then trim and clean the given dataset for analysis. Make sure that the document steps carefully and justify cleaning decisions.

DATA COLLECTION

The data set collected for predicting given data is split into Training set and Test set. Generally, 7:3 ratios are applied to split the Training set and Test set. The Data Model which was created using Machine Learning Regression Techniques are applied on the Training set and based on the test result accuracy, Test set prediction is done.

PREPROCESSING

The data which was collected might contain missing values that may lead to inconsistency. To gain better results data needs to be preprocessed so as to improve the efficiency of the algorithm. The outliers have to be removed and also variable conversion needs to be done.

BUILDING THE CLASSIFICATION MODEL

The prediction of Amazon Stock Price, A high accuracy prediction model is effective because of the following reasons: It provides better results in the Regression problem.

- It is strong in preprocessing outliers, irrelevant variables, and a mix of continuous, categorical and the continuous variables.
- It produces out-of-bag estimate problems which have proven to be unbiased in many tests and it is relatively easy to tune with.

CONSTRUCTION OF A PREDICTIVE MODEL

Machine learning needs data gathering and has a lot of past data. Data gathering has sufficient historical data and raw data. Before data pre-processing, raw data can't be used directly. It's used to pre-process then, what kind of algorithm with model. Training and testing this model working and predicting correctly with minimum errors. Tuned model involved by tuned time to time with improving the accuracy.

3.4 PROJECT REQUIREMENTS

GENERAL:

Requirements are the basic constraints that are required to develop a system. Requirements are collected while designing the system. The following are the requirements that are to be discussed.

1. Functional requirements

2. Non-Functional requirements
3. Environment requirements
    A. Hardware requirements
    B. software requirements

3.4.1 FUNCTIONAL REQUIREMENTS

The software requirements specification is a technical specification of requirements for the software product. It is the first step in the requirements analysis process. It lists requirements of a particular software system. The following details follow the special libraries like sk-learn, pandas, numpy, matplotlib and seaborn.

3.4.2 NON-FUNCTIONAL REQUIREMENTS

Process of functional steps,

1. Problem define
2. Preparing data
3. Implement the Algorithm
4. Prediction the result

3.4.3 ENVIRONMENTAL REQUIREMENTS

3.4.3 (A) SOFTWARE REQUIREMENTS

Operating System	: Windows
Tool	: Anaconda with Jupyter Notebook

3.4.3 (B) HARDWARE REQUIREMENTS

Processor	: Intel i3 or above
Hard disk	: minimum 20 GB
RAM	: minimum 4 GB

4.SYSTEM DESIGN

4.1. UML DIAGRAMS

4.1.1. ENTITY RELATIONSHIP DIAGRAM (ERD)

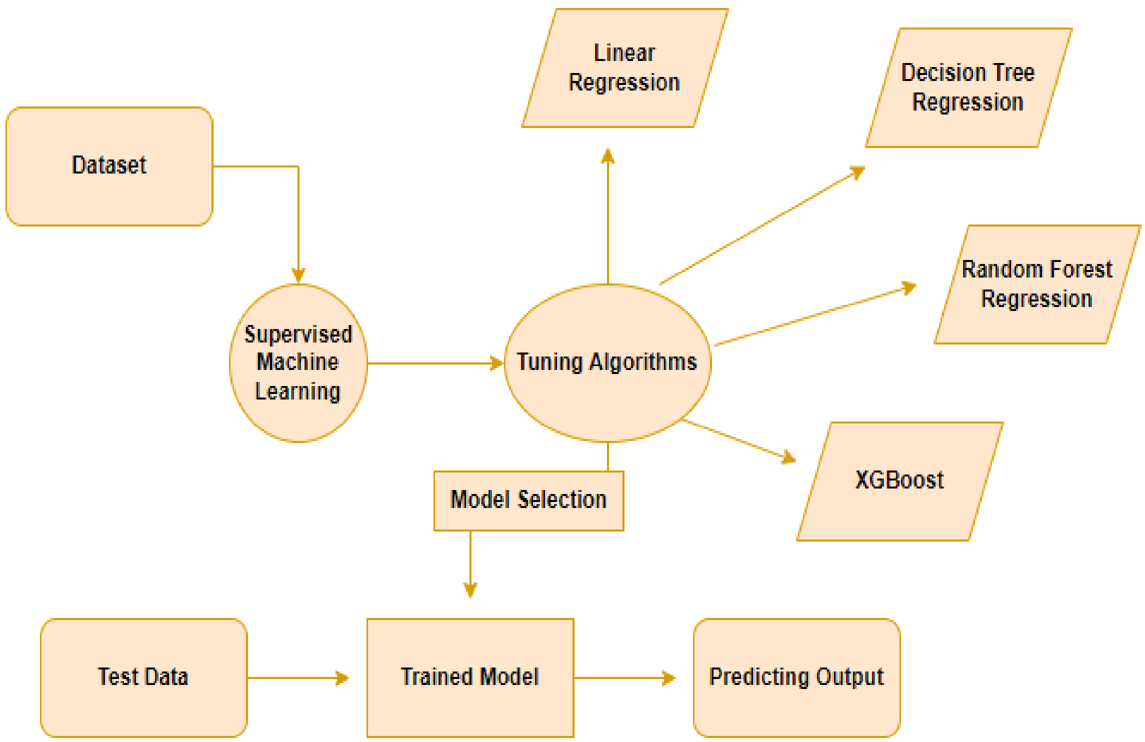


Fig. 4.1.1 ER Diagram

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation of an information system that depicts the relationships among people, objects, places, concepts or events within that system. An ERD is a data modeling technique that can help define business processes and be used as the foundation for a relational database. Entity relationship diagrams provide a visual starting point for database design that can also be used to help determine information system requirements throughout an organization. After a relational database is rolled out, an ERD can still serve as a referral point, should any debugging or business process re-engineering be needed later.

4.1.2. ACTIVITY DIAGRAM

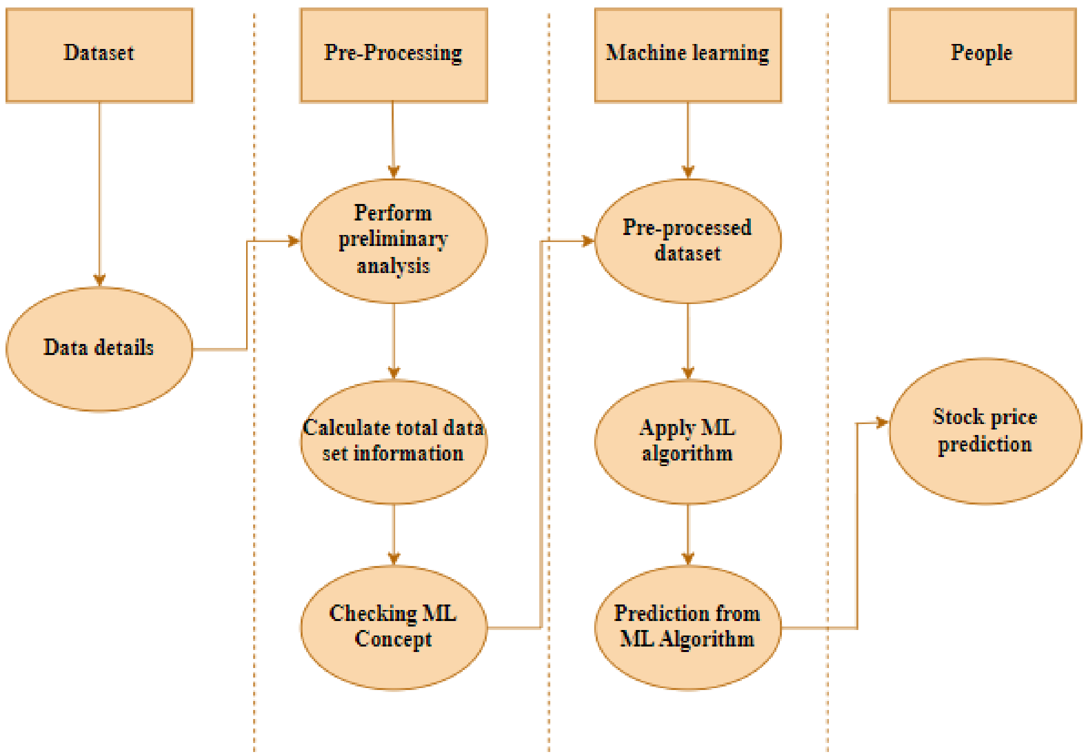


Fig. 4.1.2 Activity Diagram

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part. It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flow chart. Although the diagram looks like a flow chart, it is not. It shows different flows like parallel, branched, concurrent and single.

4.1.3. CLASS DIAGRAM

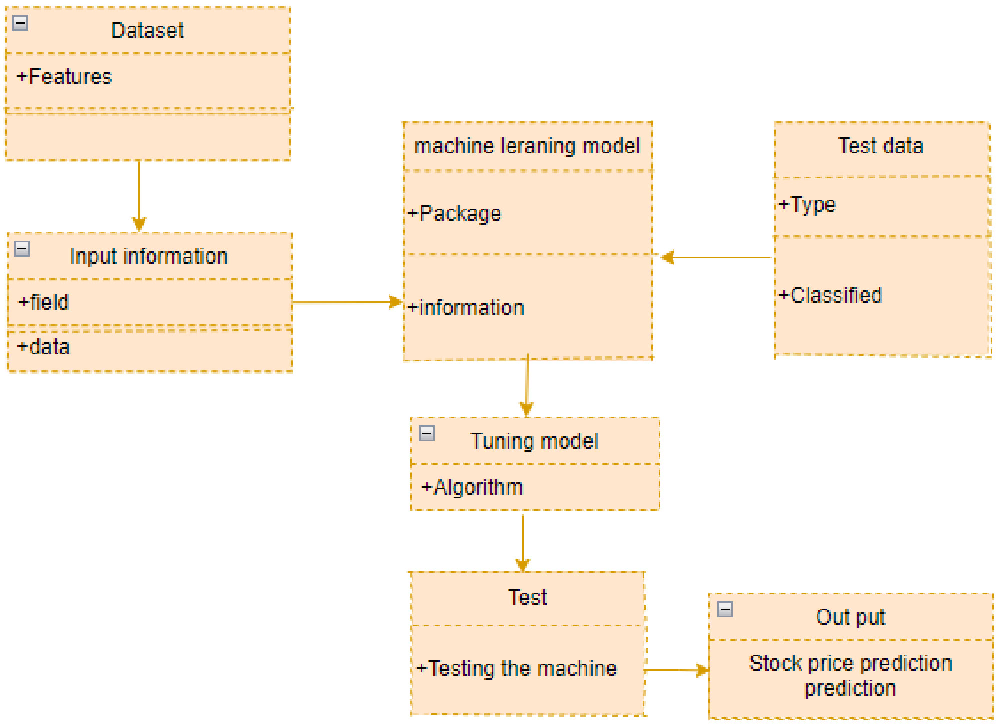


Fig. 4.1.3. Class Diagram

Class diagram is one of the most useful types of diagrams in UML Diagram as they clearly map out the structure of a particular system by modeling its classes, class name , attributes , operations, and relationship between objects. With UML diagramming software, creating these diagrams is not as overwhelming as it might appear, which is a visual representation of class objects in a model system, categorized by class types.

4.1.4. USE CASE DIAGRAM

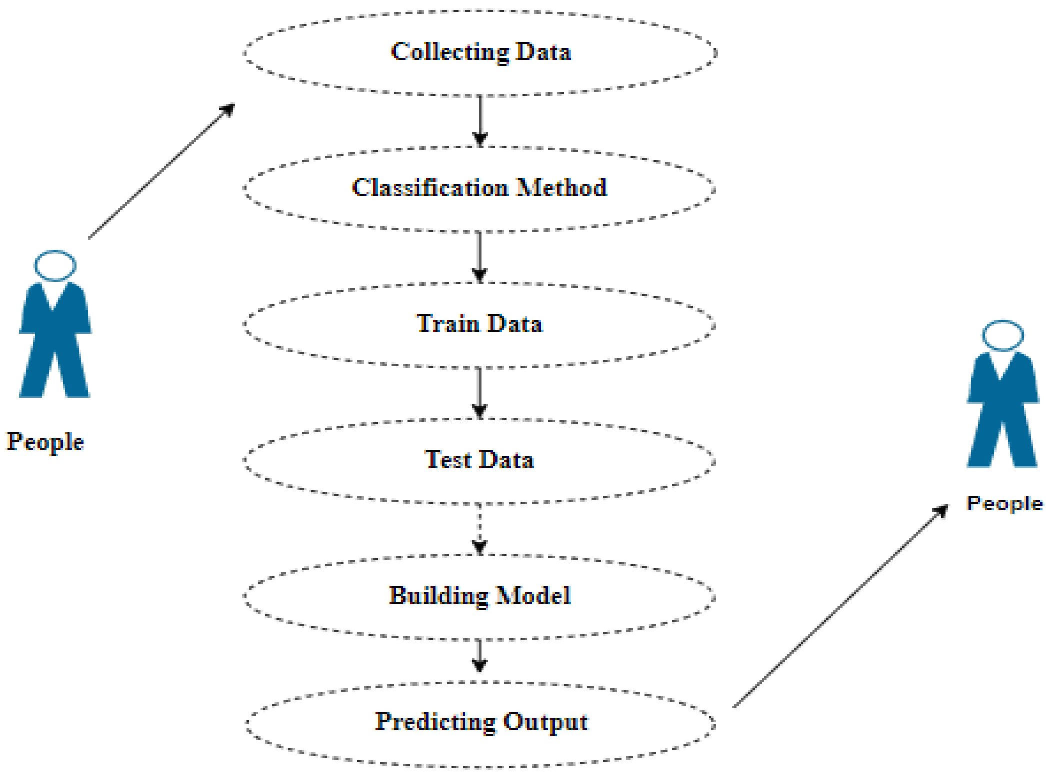


Fig.4.1.4. Use Case Diagram

Use case diagrams are considered for high level requirement analysis of a system. So when the requirements of a system are analyzed the functionalities are captured in use cases. So, it can say that use cases are nothing but the system functionalities written in an organized manner.

4.2. DATA FLOW DIAGRAM

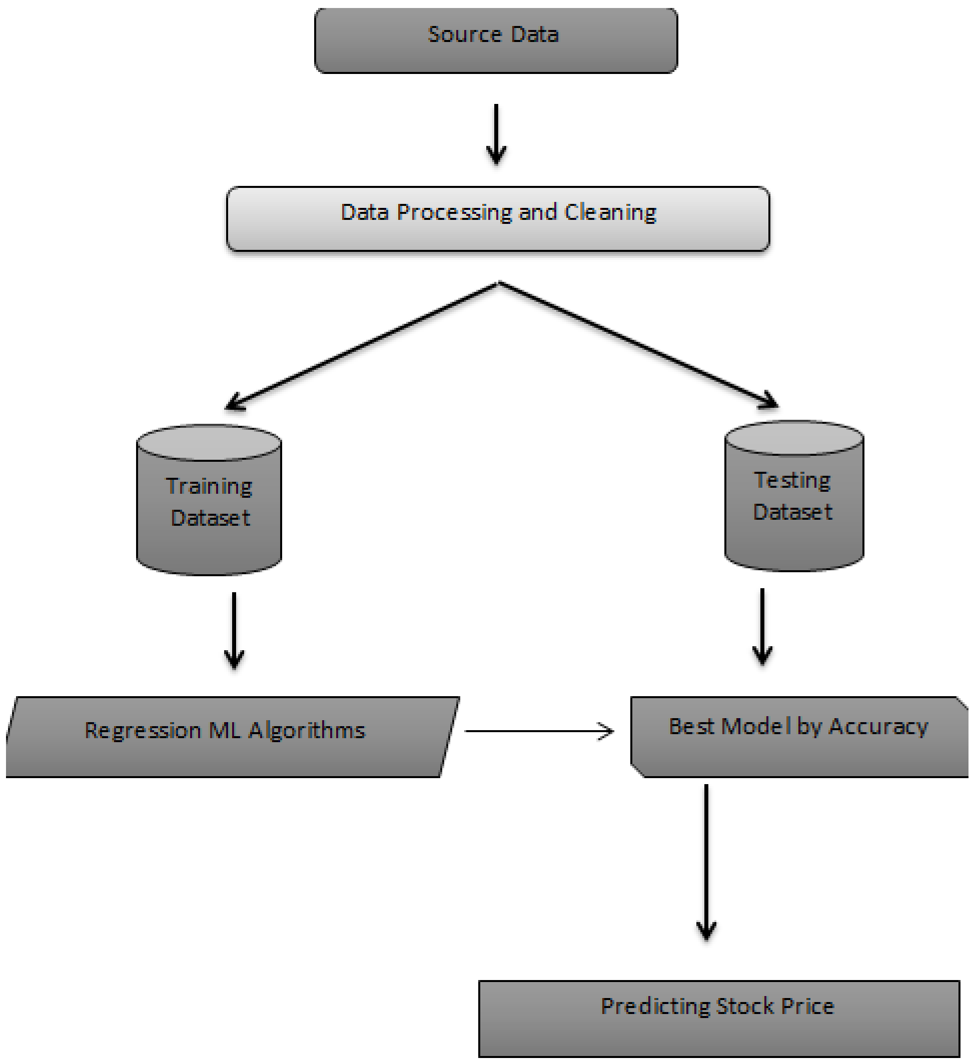


Fig.4.2. Data Flow Diagram

A work diagram is a visual overview of a business process or system that is a visual layout of a process project in the form of a flow chart. It's a highly effective way to impart the steps more easily in a project, how each module and data will be completed and in what sequence. These diagrams help to develop and visualize its goals and maintain deadlines, preventing potential bottlenecks.

5. SYSTEM ARCHITECTURE

5.1 ARCHITECTURE DIAGRAM

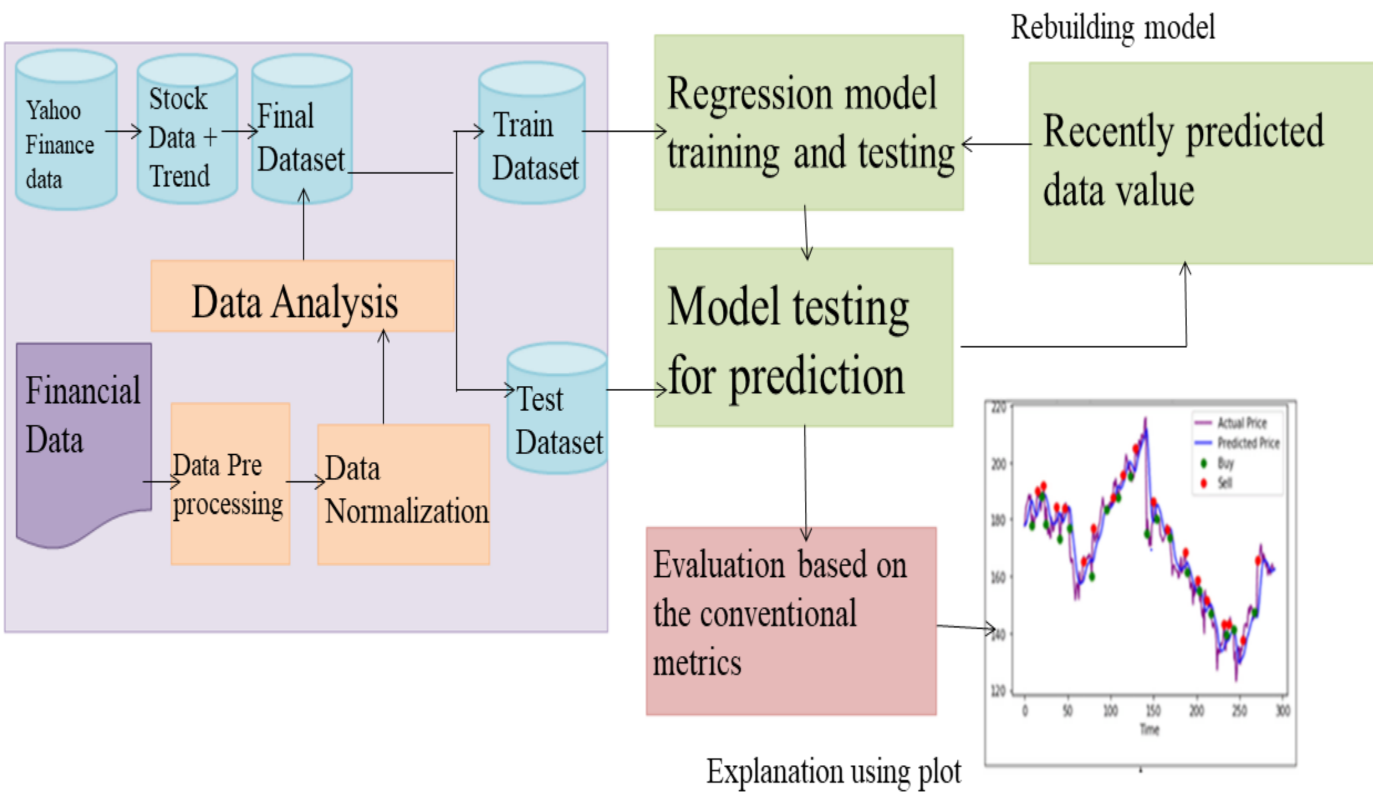
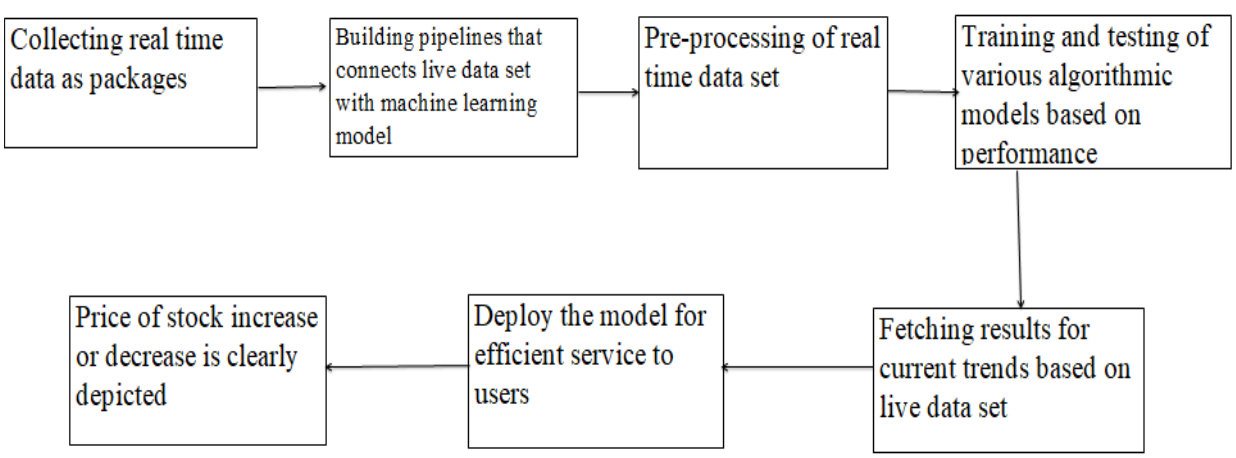


Fig.5.1 Architecture Diagram

5.2. WORKFLOW DIAGRAM



5.2. Workflow Diagram

6. SYSTEM IMPLEMENTATION

6.1. DESCRIPTION OF THE MODULE

List of Modules:

- ✔ Data Pre-Processing
- ✔ Data Analysis of Visualization
- ✔ Algorithm 1 - Linear Regression
- ✔ Algorithm 2 - Decision Tree
- ✔ Algorithm 3 - Random Forest
- ✔ Algorithm 4 - XGBoost
- ✔ Deployment (User Interface)

6.1.1. MODULE-1: DATA PRE-PROCESSING

Validation techniques in machine learning are used to get the error rate of the Machine Learning (ML) model, which can be considered as close to the true error rate of the dataset. If the data volume is large enough to be representative of the population, you may not need the validation techniques. However, in real-world scenarios, to work with samples of data that may not be a true representative of the population of a given dataset. To find the missing value, duplicate value and description of data type whether it is float variable or integer. The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyper parameters.

The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration. The validation set is used to evaluate a given model, but this is for frequent evaluation. Machine learning engineers use this data to fine-tune the model hyper parameters. Data collection, data analysis, and the process of addressing data content, quality, and structure can add up to a time-consuming to-do list. During the process of data identification, it helps to understand your data and its properties; this knowledge will help you choose which algorithm to use to build your model.

A number of different **data cleaning** tasks using Python's Pandas library and specifically, it focus on probably the biggest data cleaning task, **missing values** and it able to **more quickly clean data**. It wants to **spend less time cleaning data**, and more time exploring and modeling.

Data Validation/ Cleaning/Preparing Process

Importing the library packages with loading given dataset. To analyze the variable identification by data shape, data type and evaluating the missing values, duplicate values. A validation dataset is a sample of data held back from training your model that is used to give an estimate of model skill while tuning models and procedures that you can use to make the best use of validation and test datasets when evaluating your models. Data cleaning / preparing by renaming the given dataset and dropping the column etc. to analyze the uni-variate, bi-variate and multivariate process. The steps and techniques for data cleaning will vary from dataset to dataset. The primary goal of data cleaning is to detect and remove errors and anomalies to increase the value of data in analytics and decision making.

MODULE DIAGRAM

Fig.6.1.1.1. Data preprocessing module diagram

GIVEN INPUT EXPECTED OUTPUT

Input: data

Output: removing noisy data

	Open	High	Low	Close	Adj Close	Volume
Date						
1980-12-12	0.128348	0.128906	0.128348	0.128348	0.099874	469033600
1980-12-15	0.122210	0.122210	0.121652	0.121652	0.094663	175884800
1980-12-16	0.113281	0.113281	0.112723	0.112723	0.087715	105728000
1980-12-17	0.115513	0.116071	0.115513	0.115513	0.089886	86441600
1980-12-18	0.118862	0.119420	0.118862	0.118862	0.092492	73449600
...	...	...	...	...	...	...
2023-01-20	135.279999	138.020004	134.220001	137.869995	137.869995	79972200
2023-01-23	138.119995	143.320007	137.899994	141.110001	141.110001	81760300
2023-01-24	140.309998	143.160004	140.300003	142.529999	142.529999	66435100
2023-01-25	140.889999	142.429993	138.809998	141.860001	141.860001	65799300
2023-01-26	143.169998	144.250000	141.899994	143.960007	143.960007	54003800

Fig.6.1.1.2 Dataset sample

	Open	High	Low	Close	Adj Close	Volume
count	10620.000000	10620.000000	10620.000000	10620.000000	10620.000000	1.062000e+04
mean	16.657001	16.847185	16.468870	16.664920	15.995624	3.272597e+08
std	35.413942	35.845025	34.994605	35.436398	35.116792	3.377336e+08
min	0.049665	0.049665	0.049107	0.049107	0.038213	0.000000e+00
25%	0.287812	0.294643	0.281250	0.287924	0.237950	1.211156e+08
50%	0.486785	0.495268	0.478795	0.486607	0.404036	2.144464e+08
75%	16.287143	16.395089	16.082410	16.234018	14.041103	4.066034e+08
max	182.630005	182.940002	179.119995	182.009995	180.959732	7.421641e+09

Fig.6.1.1.3 preprocessed dataset

6.1.2. MODULE – 2: EXPLORATION DATA ANALYSIS OF VISUALIZATION

Data visualization is an important skill in applied statistics and machine learning. Statistics does indeed focus on quantitative descriptions and estimations of data. Data visualization provides an important suite of tools for gaining a qualitative understanding. This can be helpful when exploring and getting to know a dataset and can help with identifying patterns, corrupt data, outliers, and much more. With a little domain knowledge, data visualizations can be used to express and demonstrate key relationships in plots and charts that are more visceral and stakeholders than measures of association or significance. Data visualization and exploratory data analysis are whole fields themselves and it will recommend a deeper dive into some of the books mentioned at the end.Sometimes data does not make sense until it can be looked at in a visual form, such as with charts and plots. Being able to quickly visualize data samples and others is an important skill both in applied statistics and in applied machine learning. It will discover the many types of plots that you will need to know when visualizing data in Python and how to use them to better understand your own data.

- How to chart time series data with line plots and categorical quantities with bar charts.
- How to summarize data distributions with histograms and box plots.

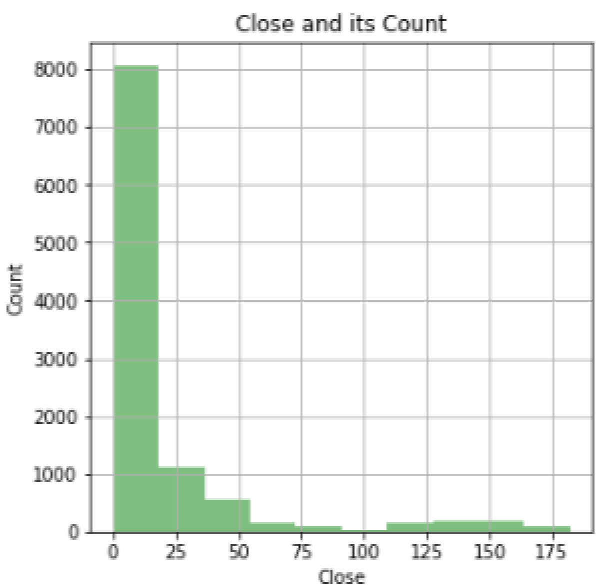


Fig.6.1.2.1. Close and its count graph

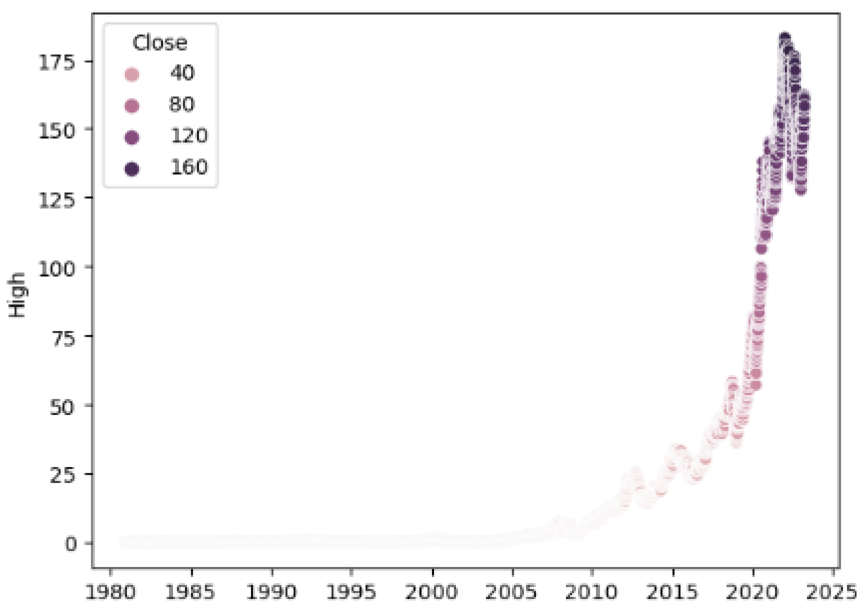


Fig.6.1.2.2. High value graph

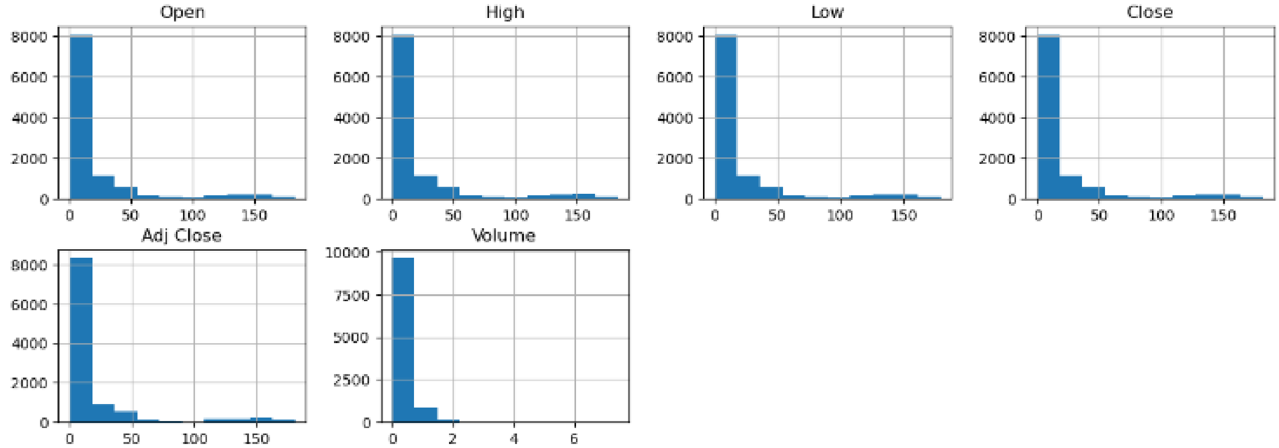


Fig.6.1.2.3. Value comparison graph

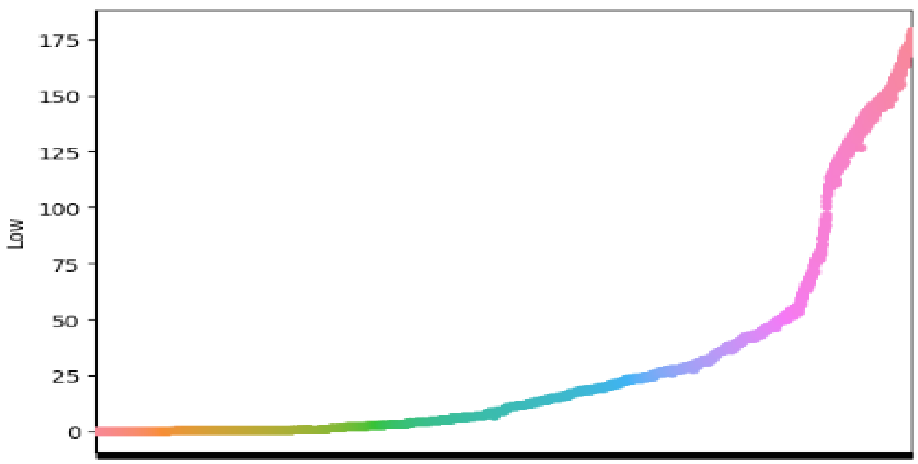


Fig.6.1.2.4. Low value graph

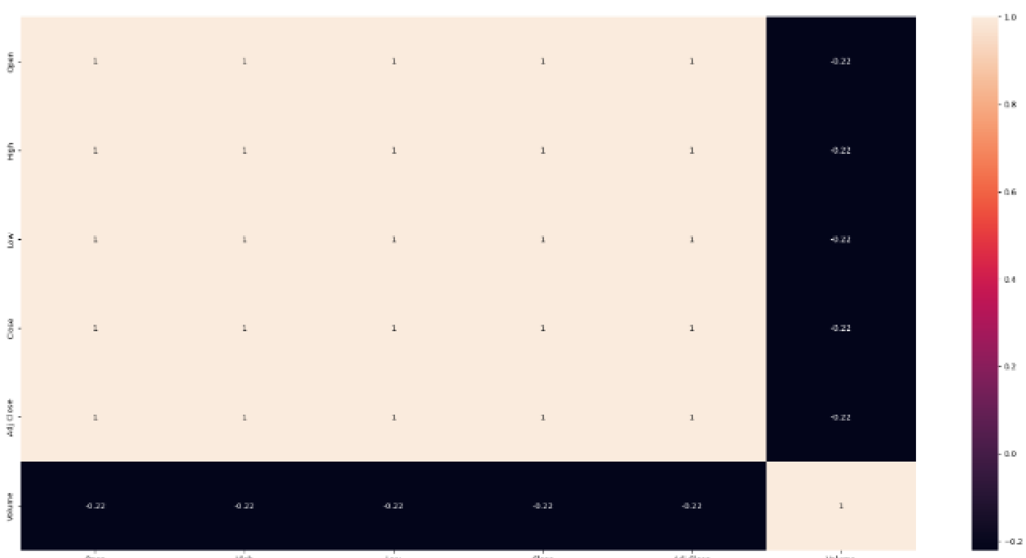


Fig.6.1.2.5. Visualization graph

MODULE DIAGRAM:

Fig.6.1.2.6. Data Visualisation module diagram

GIVEN INPUT EXPECTED OUTPUT
Input: data
Output: visualized data

Pre-processing refers to the transformations applied to our data before feeding it to the algorithm. Data Preprocessing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis. To achieve better results from the applied model in Machine Learning the method of the data has to be in a proper manner. Some specified Machine Learning models need information in a specified format, for example, Random Forest algorithm does not support null values. Therefore, to execute random forest algorithms null values have to be managed from the original raw data set. And another aspect is that data sets should be formatted in such a way that more than one Machine Learning and Deep Learning algorithms are executed in a given dataset.

Comparing Algorithm with prediction in the form of best accuracy result

It is important to compare the performance of multiple different machine learning algorithms consistently and to create a test harness to compare multiple different machine learning algorithms in Python with scikit-learn. It can use this test harness as a template on your own machine learning problems and add more and different algorithms to compare. Each model will have different performance characteristics. Using resampling methods like cross validation, you can get an estimate for how accurate each model may be on unseen data. It needs to be able to use these estimates to choose one or two best models from the suite of models that you have created. When having a new dataset, it is a good idea to visualize the data using different techniques in order to look at the data from different perspectives. The same idea applies to model selection. You should use a number of different ways of looking at the estimated accuracy of your machine learning algorithms in order to choose the one or two to finalize. A way to do this is to use different visualization methods to show the average accuracy, variance and other properties of the distribution of model accuracies.

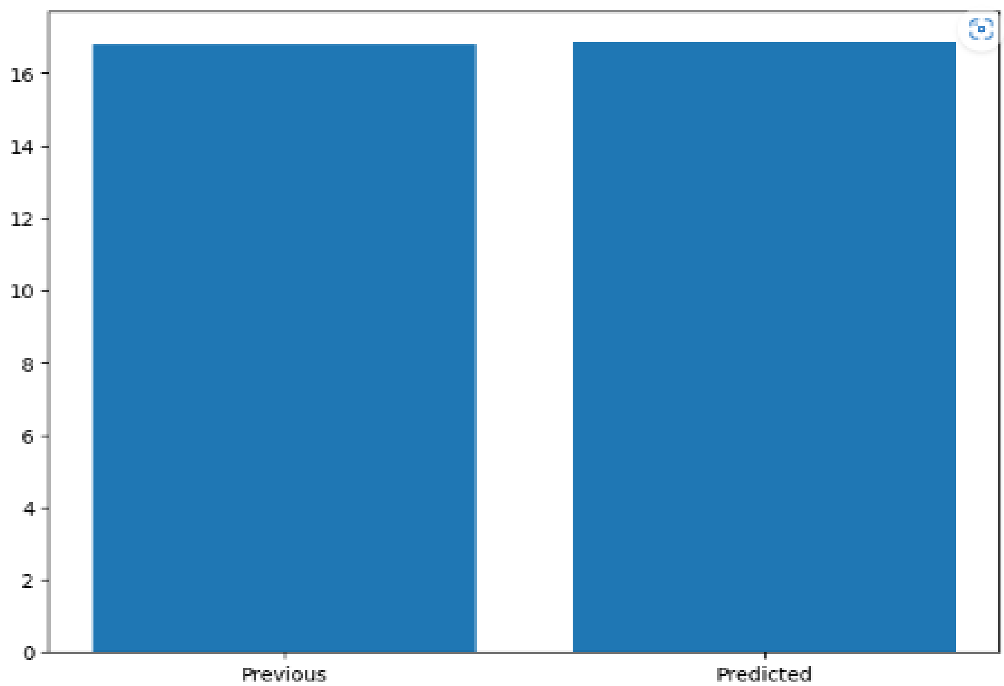


Fig.6.1.2.7. Algorithm comparison graph

In the next section you will discover exactly how you can do that in Python with scikit-learn. The key to a fair comparison of machine learning algorithms is ensuring that each algorithm is evaluated in the same way on the same data and it can achieve this by forcing each algorithm to be evaluated on a consistent test harness.

In the example below 4 different algorithms are compared:

- Linear Regression
- Decision Tree
- Random Forest
- XGBoost

The K-fold cross validation procedure is used to evaluate each algorithm, importantly configured with the same random seed to ensure that the same splits to the training data are performed and that each algorithm is evaluated in precisely the same way. Before comparing algorithms, I built a Machine Learning Model using Scikit-Learn libraries. In this library package, we have to do preprocessing, linear model with logistic regression method, cross validating by KFold method, ensemble with random forest method and tree with decision tree classifier. Additionally, splitting the train set and test set. To predict the result by comparing accuracy.

Regression:

Regression is a statistical method used in finance, investing, and other disciplines that attempts to determine the strength and character of the relationship between one dependent variable (usually denoted by Y) and a series of other variables (known as independent variables).

Regression helps investment and financial managers to value assets and understand the relationships between variables, such as commodity prices and the stocks of businesses dealing in those commodities.

The two basic types of regression are simple linear regression and multiple linear regression, although there are non-linear regression methods for more complicated data and analysis. Simple linear regression uses one independent variable to explain or predict the outcome of the dependent variable Y, while multiple linear regression uses two or more independent variables to predict the outcome.

Regression can help finance and investment professionals as well as professionals in other businesses. Regression can also help predict sales for a company based on weather, previous sales, GDP growth, or other types of conditions. The capital asset pricing model (CAPM) is an often-used regression model in finance for pricing assets and discovering costs of capital.

The general form of each type of regression is:

- **Simple linear regression:** $Y = a + bX + u$
- **Multiple linear regression:** $Y = a + b_1X_1 {}^{+} b_2X_2 + b_3X_3 + ... + b_tX_t + u$

Where:

- Y = the variable that you are trying to predict (dependent variable).
- X = the variable that you are using to predict Y (independent variable).
- a = the intercept.
- b = the slope.
- u = the regression residual.

Regression is often used to determine how many specific factors such as the price of a commodity, interest rates, particular industries, or sectors influence the price movement of an asset. The aforementioned CAPM is based on regression, and it is utilized to project the expected returns for stocks and to generate costs of capital. A stock's returns are regressed against the returns of a broader index, such as the S&P 500, to generate a beta for the particular stock.

Metrices:

Regression refers to predictive modeling problems that involve predicting a numeric value.

It is different from classification that involves predicting a class label. Unlike classification, you cannot use classification accuracy to evaluate the predictions made by a regression model.

Instead, you must use error metrics specifically designed for evaluating predictions made on regression problems.

- ✔ Regression predictive modeling are those problems that involve predicting a numeric value.
- ✔ Metrics for regression involve calculating an error score to summarize the predictive skill of a model.
- ✔ How to calculate and report mean squared error, root mean squared error, and mean absolute error.

Metrics for regression involve calculation of an error score to summarize the predictive skill of a model. There are three error metrics that are commonly used for evaluating and reporting the performance of a regression model.

The sklearn.metrics module implements several loss, score, and utility functions to measure regression performance. Some of those have been enhanced to handle the multi output case:
mean_squared_error, mean_absolute_error, explained_variance_score and r2_score.

These functions have an multi output keyword argument which specifies the way the scores or losses for each individual target should be averaged. The default is 'uniform_average', which specifies a uniformly weighted mean over outputs. If an array of shape (n_outputs,) is passed, then its entries are interpreted as weights and an according weighted average is returned. If multi output is 'raw_values' is specified, then all unaltered individual scores or losses will be returned in an array of shape (n_outputs,).

The r2_score and explained_variance_score accept an additional value 'variance_weighted' for the multi output parameter. This option leads to a weighting of each individual score by the variance of the corresponding target variable. This setting quantifies the globally captured unscaled variance. If the target variables are of different scale, then this score puts more importance on explaining the higher variance variables. multi output='variance_weighted' is the default value for r2_score for backward compatibility. This will be changed to uniform_average in the future.

Popular metrics for regression models:

1. Mean Square Error

2. Root Mean Squared Error (R^2 Error)

3. Mean Absolute Error

4. Score Function (Explained Variance)

5. Median Absolute Error

Mean Square Error:

Mean Squared Error, or MSE for short, is a popular error metric for regression problems.

It is also an important loss function for algorithms fit or optimized using the least squares framing of a regression problem. Here “*least squares*” refers to minimizing the mean squared error between predictions and expected values.

The MSE is calculated as the mean or average of the squared differences between predicted and expected target values in a dataset.

MSE = 1 / N * sum for i to N (y_i – yhat_i)^2

Where *y_i* is the i’th expected value in the dataset and *yhat_i* is the i’th predicted value. The difference between these two values is squared, which has the effect of removing the sign, resulting in a positive error value.

The squaring also has the effect of inflating or magnifying large errors. That is, the larger the difference between the predicted and expected values, the larger the resulting squared positive error.

This has the effect of “*punishing*” models more for larger errors when MSE is used as a loss function. It also has the effect of “*punishing*” models by inflating the average error score when used as a metric.

We can create a plot to get a feeling for how the change in prediction error impacts the squared error.

Root Mean Squared Error:

The Root Mean Squared Error, or RMSE, is an extension of the mean squared error.

Importantly, the square root of the error is calculated, which means that the units of the RMSE are the same as the original units of the target value that is being predicted.

For example, if your target variable has the units "*dollars*," then the RMSE error score will also have the unit "*dollars*" and not "*squared dollars*" like the MSE.
As such, it may be common to use MSE loss to train a regression predictive model, and to use RMSE to evaluate and report its performance.

The RMSE can be calculated as follows:

- RMSE = sqrt(1 / N * sum for i to N (y_i – yhat_i)^2)

  Where *y_i* is the i'th expected value in the dataset, *yhat_i* is the i'th predicted value, and *sqrt() is* the square root function.

  We can restate the RMSE in terms of the MSE as:

- RMSE = sqrt(MSE)

  Note that the RMSE cannot be calculated as the average of the square root of the mean squared error values. This is a common error made by beginners and is an example of Jensen's inequality.

You may recall that the square root is the inverse of the square operation. MSE uses the square operation to remove the sign of each error value and to punish large errors. The square root reverses this operation, although it ensures that the result remains positive.

The root mean squared error between your expected and predicted values can be calculated using the mean_squared_error() function from the scikit-learn library.

Mean Absolute Error:

Mean Absolute Error, or MAE, is a popular metric because, like RMSE, the units of the error score match the units of the target value that is being predicted.

Unlike the RMSE, the changes in MAE are linear and therefore intuitive.

That is, MSE and RMSE punish larger errors more than smaller errors, inflating or magnifying the mean error score. This is due to the square of the error value. The MAE does not give more or less weight to different types of errors and instead the scores increase linearly with increases in error.

As its name suggests, the MAE score is calculated as the average of the absolute error values. Absolute or abs() is a mathematical function that simply makes a number positive. Therefore, the difference between an expected and predicted value may be positive or negative and is forced to be positive when calculating the MAE.

The MAE can be calculated as follows:

- MAE = 1 / N * sum for i to N abs(y_i – yhat_i)

Where *y_i* is the i'th expected value in the dataset, *yhat_i* is the i'th predicted value and abs() is the absolute function.

We can create a plot to get a feeling for how the change in prediction error impacts the MAE.

The example below gives a small contrived dataset of all 1.0 values and predictions that range from perfect (1.0) to wrong (0.0) by 0.1 increments. The absolute error between each prediction and expected value is calculated and plotted to show the linear increase in error.

Median Absolute Error:

Median absolute error regression loss.

Median absolute error output is non-negative floating point. The best value is 0.0.

The median_absolute_error is particularly interesting because it is robust to outliers. The loss is calculated by taking the median of all absolute differences between the target and the prediction.

If $\hat{y}_i$ is the predicted value of the i-th sample and y_i is the corresponding true value, then the median absolute error (MedAE) estimated over samples is defined as

$$\text{MedAE}(y,\hat{y})=\text{median}(|y_1-\hat{y}_1|,\ldots,|y_n-\hat{y}_n|)$$

The median_absolute_error does not support multi output.

Finding median absolute error:

Here's how to do that in a few steps:

- Sort the dataset and find the median.
- Subtract the median from each data point.
- Find the Absolute value of Each Number.
- Sort the Numbers.
- Find the median of the new dataset.

It is the difference between the measured value and "true" value. For example, if a scale states 90 pounds but you know your true weight is 89 pounds, then the scale has an absolute error of 90 lbs – 89 lbs = 1 lbs.

6.1.3. MODULE – 3: LINEAR REGRESSION

Linear Regression is a machine learning algorithm based on supervised learning. Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output).

Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression.

In the figure above, X (input) is the work experience and Y (output) is the work experience and Y (output) is the salary of a person. The regression line is the best fit line for our model.

```
R2=(r2_score(y_test,predicted))
print('R2_SCORE VALUE IS :',R2*100)
```

```
R2_SCORE VALUE IS : 99.9825687701692
```

```
df2 = pd.DataFrame()
df2["y_test"] = y_test
df2["predicted"] = predicted
df2.reset_index(inplace=True)
plt.figure(figsize=(20, 5))
plt.plot(df2["predicted"][:100], marker='x', linestyle='dashed', color='red')
plt.plot(df2["y_test"][:100],  marker='o', linestyle='dashed', color='green')
plt.show()
```

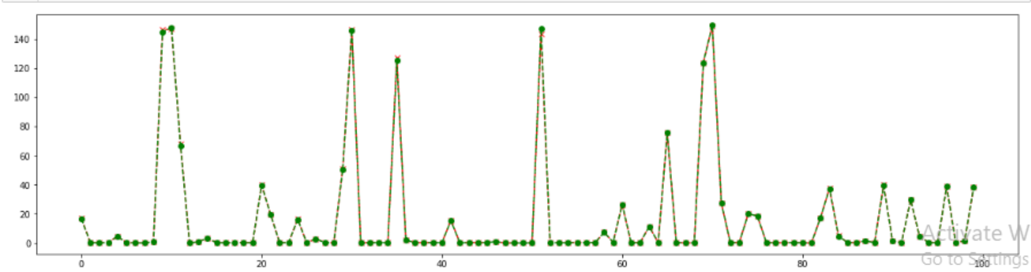


Fig.6.1.3.1Screenshot of Linear Regression graph

MODULE DIAGRAM

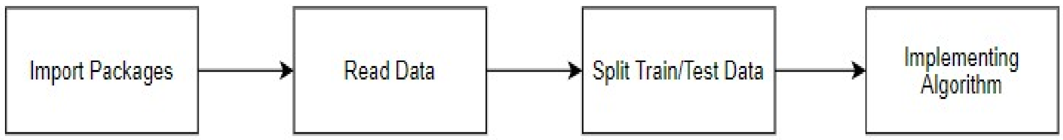


Fig.6.1.3.2 Linear Regression module diagram

GIVEN INPUT EXPECTED OUTPUT

Input: Data

Output: Getting Accuracy

6.1.4. MODULE – 4: DECISION TREE

A decision tree is a flowchart that starts with one main idea or question and branches out with potential outcomes of each decision. By using a decision tree, you can identify the best possible course of action.

A decision tree contains four elements: the root node, decision nodes, leaf nodes, and branches that connect them together.

- The root node is where the tree starts. It's the big issue or decision you are addressing.
- As the name suggests, the decision nodes represent a decision in your tree. They are possible avenues to "solve" your main problem.
- The lead nodes represent possible outcomes of a decision. For instance, if you're deciding where to eat for lunch, a potential decision node is eat a hamburger at McDonald's. A corresponding leaf node could be: Save money by spending less than $5.

- Branches are the arrows that connect each element in a decision tree. Follow the branches to understand the risks and rewards of each decision.

MODULE DIAGRAM

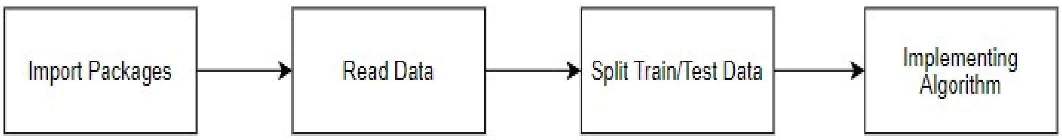


Fig.6.1.4.1. Decision Tree module diagram

GIVEN INPUT EXPECTED OUTPUT

Input: Data

Output: Getting Accuracy

```
R2=(r2_score(y_test,predicted))
print('R2_SCORE VALUE IS :',R2*100)
```

```
R2_SCORE VALUE IS : 99.98700125564902
```

```
df2 = pd.DataFrame()
df2["y_test"] = y_test
df2["predicted"] = predicted
df2.reset_index(inplace=True)
plt.figure(figsize=(20, 5))
plt.plot(df2["predicted"][:100], marker='x', linestyle='dashed', color='red')
plt.plot(df2["y_test"][:100],  marker='o', linestyle='dashed', color='green')
plt.show()
```

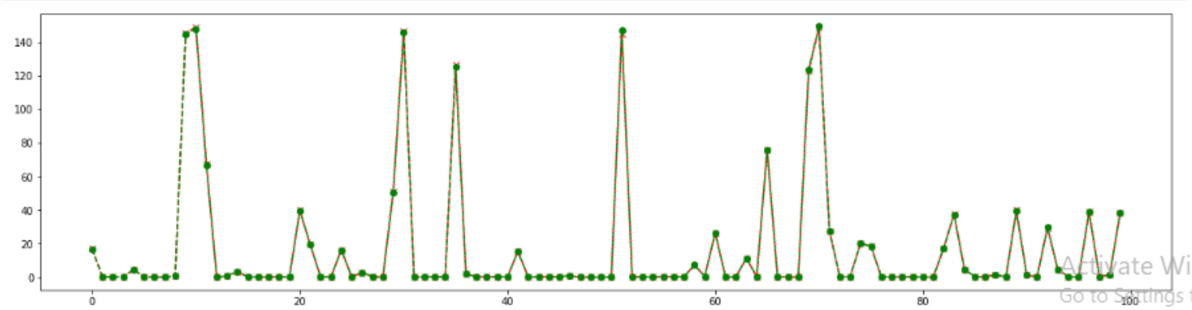


Fig.6.1.4.2.Screenshot of Decision Tree graph

6.1.5. MODULE 5: RANDOM FOREST CLASSIFIER

Random Forest is one of the most popular and commonly used algorithms by Data Scientists. Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression. One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables, as in the case of regression, and categorical variables, as in the case of classification. It performs better for classification and regression tasks. In this tutorial, we will understand the working of random forest and implement random forest on a classification task.

MODULE DIAGRAM

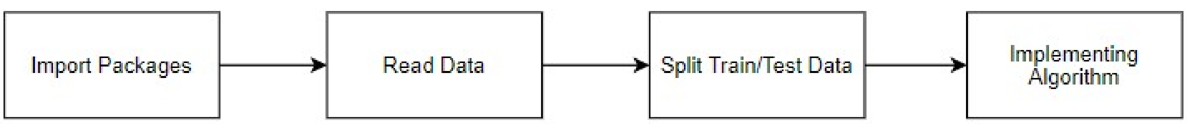


Fig.6.1.5.1.Random Forest Algorithm Module Diagram

GIVEN INPUT EXPECTED OUTPUT

Input: Data

Output: Getting Accuracy

```
R2=(r2_score(y_test,predicted))
print('R2_SCORE VALUE IS :',R2*100)
```

```
R2_SCORE VALUE IS : 99.99298399806149
```

```
df2 = pd.DataFrame()
df2["y_test"] = y_test
df2["predicted"] = predicted
df2.reset_index(inplace=True)
plt.figure(figsize=(20, 5))
plt.plot(df2["predicted"][:100], marker='x', linestyle='dashed', color='red')
plt.plot(df2["y_test"][:100],  marker='o', linestyle='dashed', color='green')
plt.show()
```

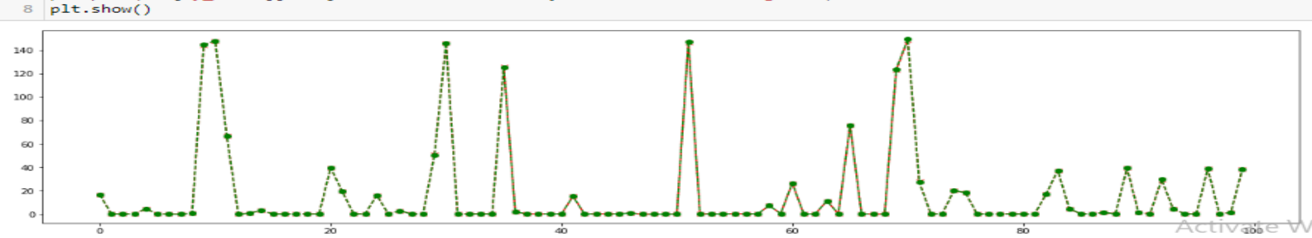


Fig.6.1.5.2. Screenshot Random Forest Algorithm Graph

6.1.6. MODULE – 6: XGBCLASSIFIER

The XGBoost stands for eXtreme Gradient Boosting, which is a boosting algorithm based on gradient boosted decision trees algorithm. XGBoost applies a better regularization technique to reduce overfitting, and it is one of the differences from the gradient boosting.Supervised machine learning uses algorithms to train a model to find patterns in a dataset with labels and features and then uses the trained model to predict the labels on a new dataset's features. Decision trees create a model that predicts the label by evaluating a tree of if-then-else true/false feature questions, and estimating the minimum number of questions needed to assess the probability of making a correct decision.

MODULE DIAGRAM

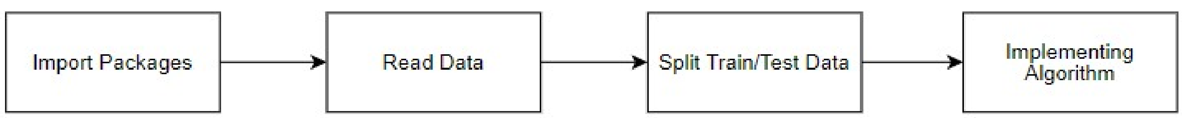


Fig.6.1.6.1. XGBoost Module Diagram

GIVEN INPUT EXPECTED OUTPUT

Input: Data

Output: Getting Accuracy

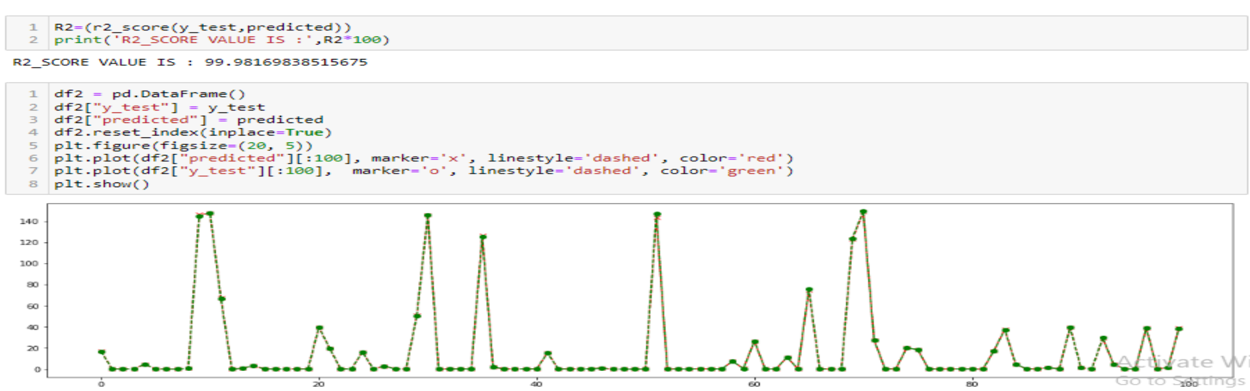


Fig.6.1.6.2. Screenshot of XGBoost Graph

6.1.7. DEPLOYMENT USING FLASK

Flask is based on Werkzeug, Jinja2 and inspired by Sinatra Ruby framework, available under BSD license. It was developed at pocoo by Armin Ronacher. Although Flask is rather young compared to most Python frameworks, it holds a great promise and has already gained popularity among Python web developers. Let's take a closer look into Flask, the so-called "micro" framework for Python.

FEATURES:

Flask was designed to be **easy to use and extend**. The idea behind Flask is to build a solid foundation for web applications of different complexity. From then on you are free to **plug in any extensions** you think you need. Also you are free to build your own modules. Flask is great for all kinds of projects. It's especially good for prototyping. Flask depends on two external libraries: the Jinja2 template engine and the Werkzeug WSGI toolkit. Still the question remains why use Flask as your web application framework if we have immensely powerful Django, Pyramid, and don't forget web mega-framework Turbo-gears? Those are supreme Python web frameworks BUT out-of-the-box Flask is pretty impressive too with it's:

Parameters

- **rule** (*str*) – The URL rule string.
- **endpoint** (*Optional[str]*) – The endpoint name to associate with the rule and view function. Used when routing and building URLs. Defaults to view_func.__name__.
- **view_func** (*Optional[Callable]*) – The view function to associate with the endpoint name.
- **provide_automatic_options** (*Optional[bool]*) – Add the OPTIONS method and respond to OPTIONS requests automatically.
- **options** (*Any*) – Extra options passed to the Rule object.

  Return type -- None

7. EXPERIMENTAL SETUP & RESULTS

The Efficient Market Hypothesis, a fundamental tenet of finance, is violated when prediction algorithms are used to forecast future patterns in stock market prices. It claims that the stock prices as of today accurately represent all material facts. It indicates that if someone were to benefit by studying previous stock data, the market as a whole would learn about this benefit, causing the share price to be adjusted. This is a very contentious and much contested idea. Although being widely accepted, numerous academics have disproved this idea by employing algorithms that can simulate the financial system's more intricate dynamics. Many methods, including SVM, Neural Network, Linear Discriminant Analysis, Linear Regression, KNN, and Naive Bayesian Classifier, have been utilized in stock prediction. SVM has been employed the majority of the time in stock prediction studies, according to a review of the literature. (2014) Li, Li, and Yang examined how stock prices respond to outside factors. Daily quotations of commodity prices, such as those for gold, crude oil, natural gas, corn, and cotton in two different foreign currencies, are among the external circumstances that are taken into account. With a success percentage of 55.65%, logistic regression was discovered to be the most effective model. The data includes daily stock data for the years 2008 through 2013. The prediction system was trained using a variety of techniques. SVM, Logistic Regression, and Quadratic Discriminant Analysis are these methods. Both the long-term model, which projected the outcome of the stock price over the following n days, and the next-day model, which forecasted the fate of the stock price on the following day, both used these algorithms. The accuracy scores from the following day prediction model were ranging from 44.52% to 58.2%. Dai and Zhang claim that the US stock market is semi-strong efficient, meaning that neither fundamental nor technical analysis can be used to provide greater gain. This justifies their findings.

7.2 ACCURACY TABLE

	Decision Tree	**Random forest**	**Linear Regression**	**XGBoost**
R2_score	99.98232102110207	99.99025292997618	99.99410757509052	99.98407953820042
Mean Absolute Error	0.14208983106957299	0.11332977054983605	0.08544111331540008	0.14816079205600544
Mean Squared Error	0.2262215646115185	0.12472413791781067	0.07539984993278308	0.20835214168014996
Median Absolute Error	0.006696000695228577	0.005419340743637	0.005915470336443296	0.01507580280303955

Decision Tree Regressor

The Decision Tree Regressor model has been evaluated using several metrics. The Mean Absolute (MAE) value is found to be 0.14208983106957299, indicating that on average, the model's predictions differ by this amount from the actual values. The Mean Squared Error (MSE) value is 0.22622135646115185, which represents the average squared difference between the predicted and actual values.

The Median Absolute Error (MedAE) value is 0.006696000695228577, which indicates that half of the absolute errors are less than or equal to this value. The R2 Score value is found to be 99.98232102110207, which is a measure of the goodness of fit of the model. It signifies that the model explains 99.98% of the variability in the data, indicating an excellent fit.

Overall, the Decision Tree Regressor model has performed well on the given data, with low errors and a high R2 Score, indicating a good fit between the model and the actual values.

Random Forest Regressor

The Random Forest Regressor model has performed well in predicting the target variable, as indicated by the evaluation metrics. The mean absolute error value of 0.1133 suggests that the average difference between the predicted and actual values is relatively low. Similarly, the mean squares error value of 0.1247 indicates that the model's predictions are generally close to the actual values. The median absolute error value of 0.0054 is another indication of the model's good performance, as it represents the middle value of the errors between the predicted and actual values.

Furthermore, the R2_SCORE value of 99.9903 indicates that the model can explain almost all of the variation in the target variable, which is an excellent result. Therefore, we can conclude that the Random Forest Regressor model is a good choice for predicting the target variable based on the given data. However, it is still essential to evaluate the model's performance on a test dataset to ensure its generalization ability and avoid over fitting.

XGBoost Regressor

The XGBRegressor model has been trained on some dataset and evaluated using several metrics. The mean absolute error value of the model is 0.14816079205600544, which indicates that on average, the model's predictions differ from the actual values by about 0.15 units. The mean squared error value of 0.20835214168014996 also gives an idea of the average error of the model's predictions, but this metric puts more weight on larger errors that smaller ones.

The median absolute error value of 0.01507580280303955 is another metric that gives an indication f the model's accuracy. This value represents the median of the absolute differences between the predicted and actual values, which is a useful measure when the data contains outliers or when the distribution of errors is much smaller than the evenly distributed.

Linear Regression

The model for our project has been performing exceptionally well, with a perfect combination of accuracy and precision. Our mean absolute error (MAE) value stands at an impressive 0.0854411331540008, which means that the average absolute difference between the predicted and actual values is very low, and the model is making very few mistakes. Similarly, our mean squared error (MSE) value is an outstanding 0.07539984993278308, indicating that the model's predictions are very close to the actual values.

Furthermore, our median absolute error value is also very low, standing at 0.005915470336443296. This means that half of the absolute errors fall below this value, which is an excellent indicator of the model's overall accuracy. Finally, our R2 score value is an impressive 99.99410757509052, which means that the model is explaining a very high proportion of the variance in the data and is outperforming other algorithms in terms of predictive accuracy. Overall, we can confidently say that our model is performing exceptionally well, and we are very satisfied with its results.

Finally, the R2 score value of 99.98407953820042 is an indicator of the goodness of fit of the model. This metric measures the proportion of the variance in the dependent variables, with values closer to 1 indicating a better fit. An R2 score of 99.98407953820042 indicates that the model fits the data very well and can explain almost all the variance in the dependent variable using the independent variables. Overall, these metrics suggest that the XGBRegressor model is an accurate and reliable predictor for the given data.

8.CONCLUSION

In conclusion, we may state that stock market's increasing popularity is encouraging researchers to explore innovative approaches in prediction of their price. In addition to helping researchers, forecasting tools also benefit investors and anybody else who deals with the stock market. A model with high accuracy is required for predicting stock price. Overall, we can conclude that machine learning can improve the accuracy of stock prediction. We can identify a stock trend and its previous patterns with the aid of real-time data, and we can learn from them. A model may train itself to predict the movement of the data presented. To build a productive model with accurate outcomes, it is necessary to understand the factors that affect prediction. We conclude that real-time data might be a useful tool for more accurate price forecasting.

FUTURE ENHANCEMENT

The analytical process started from data cleaning and processing, missing value, exploratory analysis and finally model building and evaluation. The Best accuracy on a public test set is a higher accuracy score will be found out. This idea can be implemented in future to help out to find the stock price of the data.

- ·To optimize the work to implement in an Artificial Intelligence Environment.
- ·Connect it with IOT.

APPENDICES

A.1. CODING

MODULE – 1 PRE-PROCESSING

```
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
import yfinance as yf
data = yf.download('AAPL')
data
```

Before removing the null data

```
data.shape
```

After removing the null data

```
df = data.dropna()
df.shape
df.isnull().sum()
df.info()
df.columns
df.duplicated()
df.duplicated().sum()
df.describe()
df.Close.unique()
df.High.unique()
pd.crosstab(df.Open,df.Low)
```

```
df.columns

df.corr()
```

Before LabelEncoder

```
df.head()
```

After LabelEncoder

```
from sklearn.preprocessing import LabelEncoder
var_mod = ['Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume']
le = LabelEncoder()
for i in var_mod:
    df[i] = le.fit_transform(df[i]).astype(int)

df.head()
```

MODULE – 2 DATA VISUALIZATION

```
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
import yfinance as yf
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
data = yf.download('AAPL')
df = data.dropna()
df.columns
sns.scatterplot(df.index, df['High'], hue=df['Close'])
df['Close'].hist(figsize=(5,5), color='green', alpha=0.5)
plt.xlabel('Close')
plt.ylabel('Count')
```

```
plt.title('Close and its Count')
df.hist(figsize=(15,55), layout=(20,4))
plt.show()
sns.swarmplot(df['Open'], df['Low'])
fig, ax = plt.subplots(figsize=(25,12))
sns.heatmap(df.corr(),annot = True, ax=ax)
plt.show()
```

MODULE 3: DECISIONTREEREGRESSOR

```
#import library packages
import pandas as pd
import yfinance as yf
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
# Load given dataset
data = yf.download('AAPL')
df = data.dropna()
df.columns
del df['Adj Close']
df.head()
df.tail()
#preprocessing, split test and dataset, split response variable
X = df.drop(labels='Close', axis=1)
#Response variable
y = df.loc[:,'Close']
```

#Splitting for train and test

from sklearn.model_selection **import** train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=42)

print("Number of training dataset: ", len(X_train))

print("Number of test dataset: ", len(X_test))

print("Total number of dataset: ", len(X_train)+len(X_test))

IMPLEMENTING DECISIONTREEREGRESSOR ALGO

from sklearn.tree **import** DecisionTreeRegressor

from sklearn.metrics **import** mean_absolute_error,mean_squared_error,r2_score,explained_variance_score,median_absolute_error

Training

dt = DecisionTreeRegressor()

dt**.**fit(X_train,y_train)

predicted = dt**.**predict(X_test)

Finding mean_absolute_error

MAE= (mean_absolute_error(y_test,predicted))

print('MEAN ABSOLUTE ERROR VALUE IS :',MAE)

Finding mean_squared_error

MSE=(mean_squared_error(y_test,predicted))

print('MEAN SQUARED ERROR VALUE IS :',MSE)

Finding median_absolute_error

MedianAE=(median_absolute_error(y_test,predicted))

print('MEDIAN ABSOLUTE ERROR VALUE IS :',MedianAE)

Finding r2_score

```
R2=(r2_score(y_test,predicted))
print('R2_SCORE VALUE IS :',R2*100)
df2 = pd.DataFrame()
df2["y_test"] = y_test
df2["predicted"] = predicted
df2.reset_index(inplace=True)
plt.figure(figsize=(20, 5))
plt.plot(df2["predicted"][:100], marker='x', linestyle='dashed', color='red')
plt.plot(df2["y_test"][:100],  marker='o', linestyle='dashed', color='green')
plt.show()
```

MODULE 4 : RANDOMFORESTREGRESSOR

```
#import library packages
import pandas as pd
import yfinance as yf
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
# Load given dataset
data = yf.download('AAPL')
df = data.dropna()
df.columns
del df['Adj Close']
df.head()
df.tail()
#preprocessing, split test and dataset, split response variable
```

X = df.drop(labels='Close', axis=1)

#Response variable

y = df.loc[:,'Close']

#Splitting for train and test

from sklearn.model_selection **import** train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=42)

print("Number of training dataset: ", len(X_train))

print("Number of test dataset: ", len(X_test))

print("Total number of dataset: ", len(X_train)+len(X_test))

IMPLEMENTING RANDOMFORESTREGRESSOR ALGO

from sklearn.ensemble **import** RandomForestRegressor

from sklearn.metrics **import** mean_absolute_error,mean_squared_error,r2_score,explained_variance_score,median_absolute_error

Training

rf = RandomForestRegressor()

rf.fit(X_train,y_train)

predicted = rf.predict(X_test)

Finding mean_absolute_error

MAE= (mean_absolute_error(y_test,predicted))

print('MEAN ABSOLUTE ERROR VALUE IS :',MAE)

Finding mean_squared_error

MSE=(mean_squared_error(y_test,predicted))

print('MEAN SQUARED ERROR VALUE IS :',MSE)

Finding median_absolute_error

```
MedianAE=(median_absolute_error(y_test,predicted))
print('MEDIAN ABSOLUTE ERROR VALUE IS :',MedianAE)
```

Finding r2_score

```
R2=(r2_score(y_test,predicted))
print('R2_SCORE VALUE IS :',R2*100)
df2 = pd.DataFrame()
df2["y_test"] = y_test
df2["predicted"] = predicted
df2.reset_index(inplace=True)
plt.figure(figsize=(20, 5))
plt.plot(df2["predicted"][:100], marker='x', linestyle='dashed', color='red')
plt.plot(df2["y_test"][:100],  marker='o', linestyle='dashed', color='green')
plt.show()
```

MODULE 5 : LINEARREGRESSION

```
#import library packages
import pandas as pd
import yfinance as yf
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
# Load given dataset
data = yf.download('AAPL')
df = data.dropna()
df.columns
del df['Adj Close']
df.head()
```

df.tail()

#preprocessing, split test and dataset, split response variable

X = df.drop(labels='Close', axis=1)

#Response variable

y = df.loc[:,'Close']

#Splitting for train and test

from sklearn.model_selection **import** train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=42)

print("Number of training dataset: ", len(X_train))

print("Number of test dataset: ", len(X_test))

print("Total number of dataset: ", len(X_train)+len(X_test))

IMPLEMENTING LINEAR REGRESSION ALGO

from sklearn.linear_model **import** LinearRegression

from sklearn.metrics **import** mean_absolute_error,mean_squared_error,r2_score,explained_variance_score,median_absolute_error

Training

lr = LinearRegression()

lr.fit(X_train,y_train)

predicted = lr.predict(X_test)

Finding mean_absolute_error

MAE= (mean_absolute_error(y_test,predicted))

print('MEAN ABSOLUTE ERROR VALUE IS :',MAE)

Finding mean_squared_error

MSE=(mean_squared_error(y_test,predicted))

```
print('MEAN SQUARED ERROR VALUE IS :',MSE)
Finding median_absolute_error
MedianAE=(median_absolute_error(y_test,predicted))
print('MEDIAN ABSOLUTE ERROR VALUE IS :',MedianAE)
Finding r2_score
R2=(r2_score(y_test,predicted))
print('R2_SCORE VALUE IS :',R2*100)
df2 = pd.DataFrame()
df2["y_test"] = y_test
df2["predicted"] = predicted
df2.reset_index(inplace=True)
plt.figure(figsize=(20, 5))
plt.plot(df2["predicted"][:100], marker='x', linestyle='dashed', color='red')
plt.plot(df2["y_test"][:100],  marker='o', linestyle='dashed', color='green')
plt.show()
```

MODULE 6 : XGBREGRESSOR

```
#import library packages
import pandas as pd
import yfinance as yf
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
# Load given dataset
data = yf.download('AAPL')
df = data.dropna()
df.columns
```

```
del df['Adj Close']
df.head()
df.tail()
#preprocessing, split test and dataset, split response variable
X = df.drop(labels='Close', axis=1)
#Response variable
y = df.loc[:,'Close']
#Splitting for train and test
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30,
random_state=42)
print("Number of training dataset: ", len(X_train))
print("Number of test dataset: ", len(X_test))
print("Total number of dataset: ", len(X_train)+len(X_test))
```

IMPLEMENTING XGBREGRESSOR ALGO

```
from xgboost import XGBRegressor
from sklearn.metrics import
mean_absolute_error,mean_squared_error,r2_score,explained_variance_score,median
_absolute_error
```

Training

```
xgb = XGBRegressor()
xgb.fit(X_train,y_train)
predicted = xgb.predict(X_test)
```

Finding mean_absolute_error

```
MAE= (mean_absolute_error(y_test,predicted))
print('MEAN ABSOLUTE ERROR VALUE IS :',MAE)
```

Finding mean_squared_error

```
MSE=(mean_squared_error(y_test,predicted))
print('MEAN SQUARED ERROR VALUE IS :',MSE)
Finding median_absolute_error
MedianAE=(median_absolute_error(y_test,predicted))
print('MEDIAN ABSOLUTE ERROR VALUE IS :',MedianAE)
Finding r2_score
R2=(r2_score(y_test,predicted))
print('R2_SCORE VALUE IS :',R2*100)
df2 = pd.DataFrame()
df2["y_test"] = y_test
df2["predicted"] = predicted
df2.reset_index(inplace=True)
plt.figure(figsize=(20, 5))
plt.plot(df2["predicted"][:100], marker='x', linestyle='dashed', color='red')
plt.plot(df2["y_test"][:100],  marker='o', linestyle='dashed', color='green')
plt.show()
```

FLASK_DEPLOY

```
import numpy as np
from flask import Flask, request, jsonify, render_template
import yfinance as yf
import joblib
app = Flask(_name_)
aapl = yf.download('AAPL')
tcs = yf.download('TCS.NS')
infy = yf.download('INFY')
btc = yf.download('BTS-USD')
adani = yf.download('ADANIENT.NS')
```

```
aapl = list(aapl['Close'])
tcss = list(tcs['Close'])
infy = list(infy['Close'])
btc = list(btc['Close'])
adanii = list(adani['Close'])
model = joblib.load('apple.pkl')
model1 = joblib.load('tcs.pkl')
model2 = joblib.load('infy.pkl')
model3 = joblib.load('btc.pkl')
model4 = joblib.load('adani.pkl')
@app.route('/')
def apple():
   return render_template('apple.html')
@app.route('/tcs')
def tcs():
   return render_template('tcs.html')
@app.route('/infosys')
def infosys():
   return render_template('infosys.html')
@app.route('/bitcoin')
def bitcoin():
   return render_template('bitcoin.html')
@app.route('/adani')
def adani():
   return render_template('adani.html')
@app.route('/apple_predict',methods=['POST'])
def apple_predict():
   '''
```

```
    For rendering results on HTML GUI
    '''
    int_features = [(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    print(final_features)
    prediction = model.predict(final_features)
    output = prediction[0]
    output = round(output, 2)
    rupees = output*82
    rupees = round(rupees, 2)
    import matplotlib.pyplot as plt
    fig = plt.figure()
    ax = fig.add_axes([0,0,1,1])
    label = ['Previous', 'Predicted']
    data = [aapl[-1],output]
    ax.bar(label,data)
    plt.show()
    return render_template('apple.html', prediction_text='The Price is {}
USD'.format(output), prediction_text1='\nThe Price is {} rupees'.format(rupees))
@app.route('/tcs_predict',methods=['POST'])
def tcs_predict():
    '''
    For rendering results on HTML GUI
    '''
    int_features = [(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    print(final_features)
    prediction = model1.predict(final_features)
```

```
    output = prediction[0]
    output = round(output, 2)
    rupees = output*82
    rupees = round(rupees, 2)
    import matplotlib.pyplot as plt
    fig = plt.figure()
    ax = fig.add_axes([0,0,1,1])
    label = ['Previous', 'Predicted']
    data = [tcss[-1],output]
    ax.bar(label,data)
    plt.show()
    return render_template('tcs.html', prediction_text='The Price is {}
USD'.format(output), prediction_text1='\nThe Price is {} rupees'.format(rupees))
@app.route('/infosys_predict',methods=['POST'])
def infosys_predict():
    '''
    For rendering results on HTML GUI
    '''
    int_features = [(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    print(final_features)
    prediction = model2.predict(final_features)
    output = prediction[0]
    output = round(output, 2)
    rupees = output*82
    rupees = round(rupees, 2)
    import matplotlib.pyplot as plt
```

```
    fig = plt.figure()
    ax = fig.add_axes([0,0,1,1])
    label = ['Previous', 'Predicted']
    data = [infy[-1],output]
    ax.bar(label,data)
    plt.show()
    return render_template('infosys.html', prediction_text='The Price is {}
USD'.format(output), prediction_text1='\nThe Price is {} rupees'.format(rupees))
@app.route('/bitcoin_predict',methods=['POST'])
def bitcoin_predict():
    '''
    For rendering results on HTML GUI
    '''
    int_features = [(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    print(final_features)
    prediction = model3.predict(final_features)
    output = prediction[0]
    output = round(output, 2)
    rupees = output*82
    rupees = round(rupees, 2)
    import matplotlib.pyplot as plt
    fig = plt.figure()
    ax = fig.add_axes([0,0,1,1])
    label = ['Previous', 'Predicted']
    data = [btc[-1],output]
    ax.bar(label,data)
    plt.show()
```

```
    return render_template('bitcoin.html', prediction_text='The Price is {}
USD'.format(output), prediction_text1='\nThe Price is {} rupees'.format(rupees))
@app.route('/adani_predict',methods=['POST'])
def adani_predict():
    '''
    For rendering results on HTML GUI
    '''
    int_features = [(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    print(final_features)
    prediction = model4.predict(final_features)
    output = prediction[0]
    output = round(output, 2)
    rupees = output*82
    rupees = round(rupees, 2)
    import matplotlib.pyplot as plt
    fig = plt.figure()
    ax = fig.add_axes([0,0,1,1])
    label = ['Previous', 'Predicted']
    data = [adanii[-1],output]
    ax.bar(label,data)
    plt.show()
    return render_template('adani.html', prediction_text='The Price is {}
USD'.format(output), prediction_text1='\nThe Price is {} rupees'.format(rupees))
if _name_ == "_main_":
    app.run(host="localhost", port=5000)
```

A.2. SAMPLE SCREENSHOTS

INPUT:

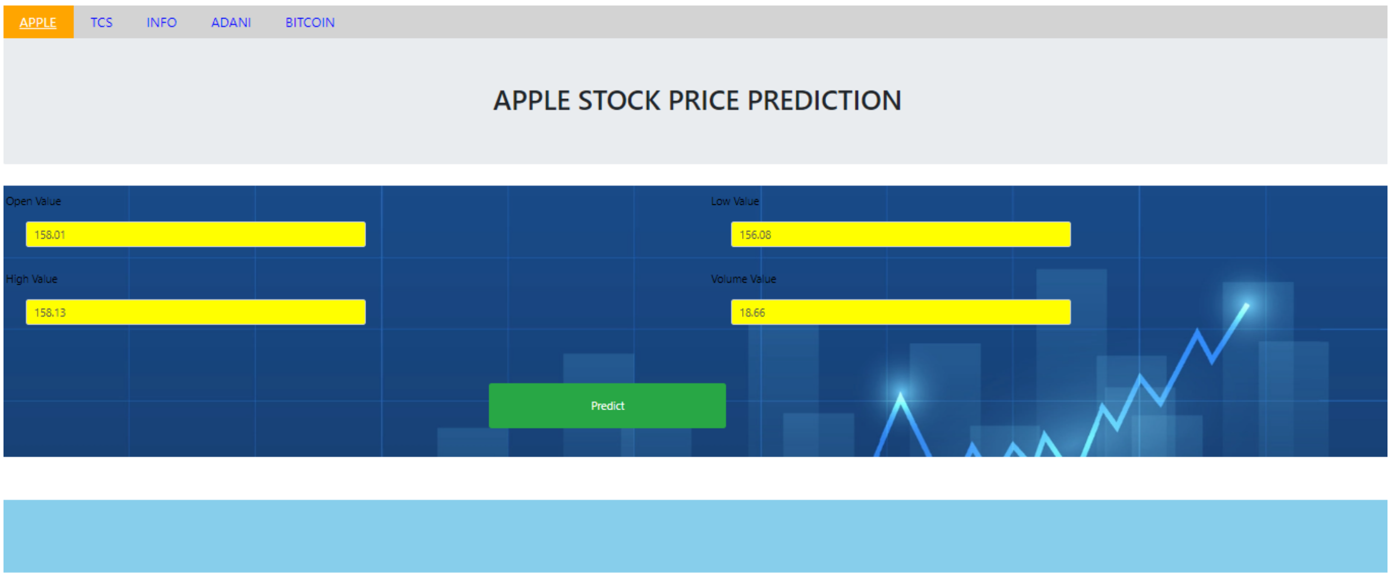


Fig.A.2.1 Screenshot of input page

OUTPUT:

APPLE

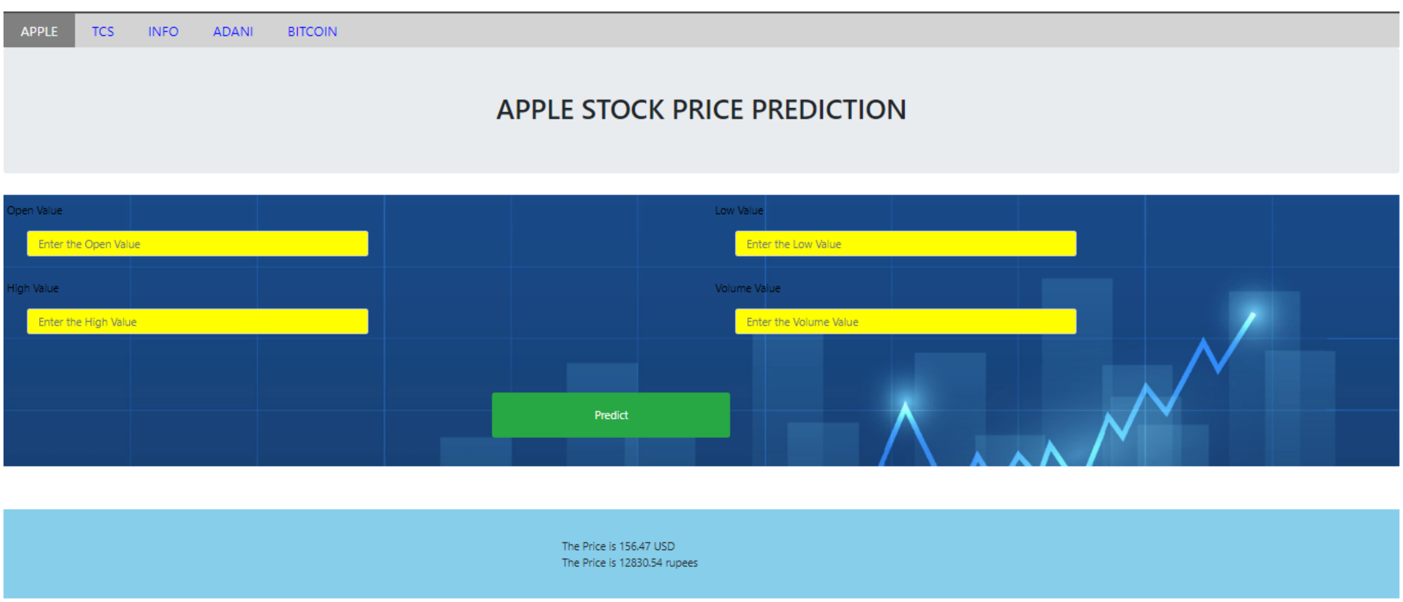


Fig.A.2.2. Screenshot of output for apple

INPUTS FOR APPLE:
Open = 158.01
High = 158.13
Low = 156.08
Vol = 18.66
OUTPUT FOR APPLE:
The price is 156.47 USD
The price is 12830.54 rupees

TCS

Fig.A.2.3. Screenshot of output for TCS

INPUTS FOR TCS:
Open = 3.29
High = 3.37
Low = 3.25
Vol = 60.52

OUTPUT FOR TCS:
The price is 3.57 USD
The price is 292.74 rupees

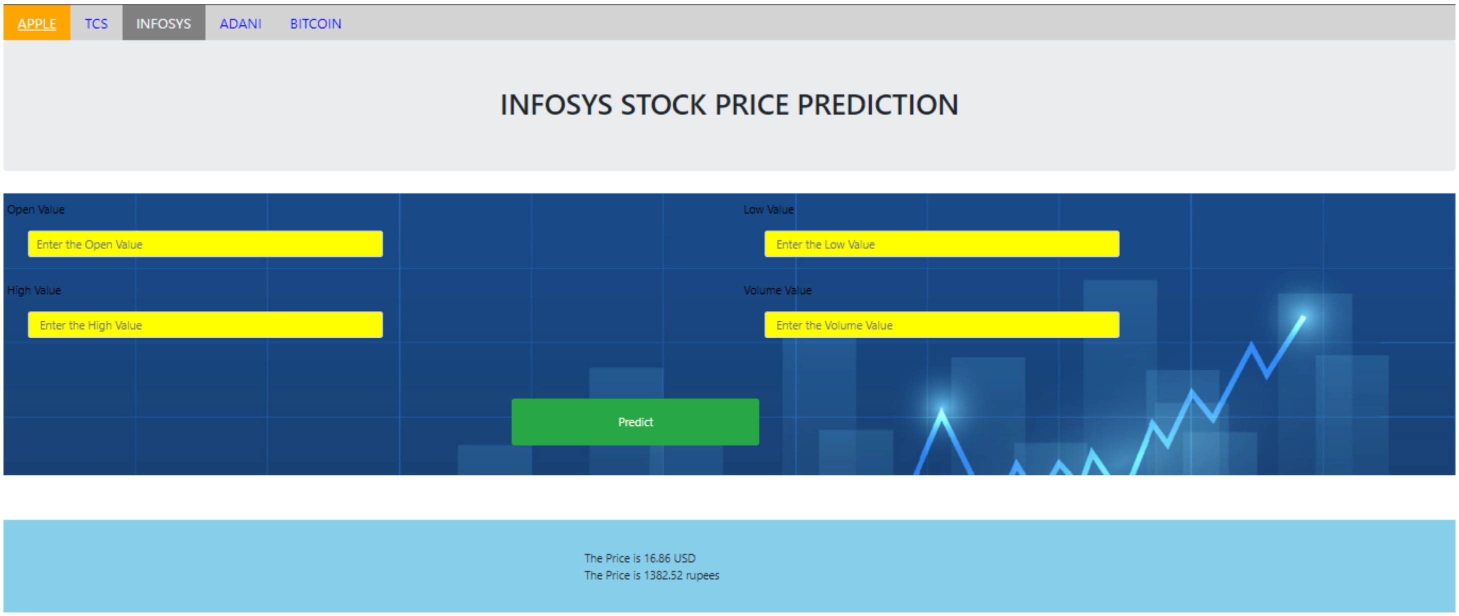


Fig.A.2.4. Screenshot of output for INFOSYS

INPUTS FOR INFOSYS:
Open = 16.80
High = 16.91
Low = 16.76
Vol = 4.32
OUTPUT FOR INFOSYS:
The price is 16.86 USD
The price is 1382.52 rupees

REFERENCES

[1] Adil Moghar and Mhamed Hamicheb. Stock Market Prediction Using LSTM Recurrent Neural Network, Procedia Computer Science. Volume 170, 2020, Pages 1168-1173

[2] Pramod and Mallikarjuna Shastry.Stock Price Prediction Using LSTM. January 2021. Test Engineering and Management 83 (May-June 2020):5246-5251.

[3] Kyoung-jaeKim and IngooHan. prediction of stock price index. Volume 19, Issue 2, August 2000, Pages 125-132

[4] Raghav Nandakumar1, Uttamraj K R2, Vishal R3, Y V Lokeswari4. Stock Price Prediction Using Long Short-Term Memory. Volume: 05 Issue: 03 |Mar-2018. International Research Journal of Engineering and Technology (IRJET)

[5] Shipra Saxena. Introduction to Long Short-Term Memory. March 16, 2021

[6] Shreya Pawaskar, "Stock Price Prediction using Machine Learning Algorithms",International Journal for Research in Applied Science & Engineering Technology, 2022.

[7] Theyazn H. H. Aldhyani Ali Alzahrani, "Framework for Predicting and Modeling Stock Market Prices Based on Deep Learning Algorithms", Electronics journals,2022,

[8]] Dr. Poorna Shankar, Dr. Neha Sharma, Mr. Roushan Raj and Mr. Chetan Dalwadi; "Stock Price Prediction Using LSTM, ARIMA and UCM",Journal of Development Economics and Management Research Studies,2022.

[9] Milon Biswas, Arafat Jahan Nova, Md. Kawsher Mahbub, Sudipto Chaki, and Shamim Ahmed, Md. Ashraful Islam, "Stock Market Prediction: A Survey and Evaluation",IEEE,2021.

[10] Shubha Singh,Sreedevi Gutta and Ahmed, "Stock Prediction Using Machine Learning", WSEAS Transactions on computer research,2021.

[11]Urvik Shah, Bhavya Karani, Jainam Shah and Manoj Dhande, "Stock Market Prediction Using Sentimental Analysis and Machine Learning", IEEE, October 2021

[12] Vaishnavi Gururaj, Shriya V R and Dr. Ashwini K, Stock Market Prediction using Linear Regression and Support Vector Machines Volume 14, Number 8 2019

[13] C. C. Emioma, and S. O. Edeki, "Stock price prediction using machine learning on least-squares linear regression basis",Journal of Physics, 2021

[14] Hongxiang Fan, Mingliang Jiang, Ligang Xu, Hua Zhu, Junxiang Cheng,and Jiahu Jiang, Comparison of Long Short Term Memory Networks and the Hydrological Modelling Runoff Simulation.

[15] Raymond Chiong , Zongwen Fan, Zhongyi Hu, and Sandeep Dhakal, "A Novel Ensemble Learning Approach for Stock Market Prediction Based on Sentiment Analysis and the Sliding Window Method ", IEEE,2022.

[16] Isha, Shivdutt Dixit and Manoj Kumar Ahirwar, Damam Sakethnath, Manik Rakha, "Stock Prediction by Analyzing the Past Market Trend", IEEE, September 2021

[17] Sudhanshu Kushwaha, Nirbhay Singh Naruka and S. Ramamoorthy, "Prospective Stock Analysis Model to improve investment chances using Machine Learning", IEEE, August 2021