CHAPTER 1

INTRODUCTION

1.1 Overview

The stock market has become an integral part of financial system of every nation. It will generally reflect the economic situations of a nation's economy. Stock market movements can have a significant economic impact on the macro and micro economy. Predicting movement in the stock market has been of interest to practitioners and researchers from diverse fields. Forecasting stock price movement is regarded as a difficult task as the market is a non-linear, non-parametric and noisy in nature. Movement in the market is influence by factors such as investors psychology, stock related news, economic conditions, etc. The efficient market hypothesis (EMH) states that it is impossible to consistently achieve risk-adjusted returns higher than the profitability of the overall market as stock prices reflect all information. Contrary to EMH, many studies have concluded that stock market movement can be predicted by using publicly available information, such as past stock data, earnings-price ratios, interest rates, monetary growth rates, inflation rates and dividend yields. Currently several approaches exist which help market practitioners to predict market movement. These approaches can be broadly grouped into three. These are fundamental analysis, technical analysis and machine learning technique. Fundamental analysis involves evaluating the intrinsic value of a stock to search for long-term investment opportunities. A fundamental analyst studies the overall economy, industry conditions and management and financial strength of the individual companies. Technical analysis predicts stock price movements by means of historical stock price charts and market statistics without consideration of any underlying economic. It is based on the assumption that if investors are able to find previous patterns in the market, they can generate a fairly accurate forecast of the future price movement. Machine Learning (ML) technique has come about due to advances in computational techniques and information technology. Machine learning is a branch of artificial intelligence (AI) that provides systems the ability to learn and improve automatically from past experience without being programmed explicitly. Prediction of stock market movement with machine learning has gained tremendous popularity because of its ability to handle the non-linear, noisy and dynamic nature of the market better than the other methods. Support vector machine (SVM) and logistic regression (LR) are among the most popular ML algorithms. The SVM has kernel function enables it to

separate the data points in a higher dimension space if they are not linearly separable in a lower dimension. It is memory efficient as it uses a subset of the training samples in the decision function (called support vector). Also, the SVM does not suffer from overfitting, not influence by outliers and has the ability to handle high dimensional data well. Logistic Regression is a very simple machine learning algorithms which is easy to implement yet offers great training efficiency and does not require high computation power. It is very efficient technique which is highly interpretable. A major drawback of SVM and LR is that both of them are sensitive to scaling. Since optimization of SVM happens by minimize the decision vector, the optimal hyperplane is affected by the scale of the input features, hence, data must be scaled prior to SVM training. LR uses gradient descent as an optimization technique, hence, it requires data to be scaled. This study therefore, aims to comparatively evaluate the performance of both SVM and LR with standardisation and Min-Max data scaling techniques in predicting the movement of stock prices.

1.2 Objectives

- To design and develop stock price prediction using machine learning model.
- To develop a system that accurately gives the closing price of the market.
- To develop a system that can be suitable for all the stocks.
- To design and develop a system with low cost.

1.2.1 Investment Decision Making: Helping investors make informed decisions about buying, selling, or holding stocks.

1.2.2 Risk Management: Identifying and managing potential risks associated with stock investments.

1.2.3 Portfolio Optimization: Assisting in the construction and management of diversified investment portfolios.

1.2.4 Algorithmic Trading: Supporting automated trading strategies based on predicted price movements.

1.2.5 Market Analysis: Providing insights into market trends and behaviors for strategic planning.

1.2.6 Financial Planning: Assisting individuals and organizations in setting and achieving financial goals based on anticipated market movements.

1.3 Scope of the project

The future scope of stock market price prediction is poised for significant expansion, driven by a confluence of technological advancements and interdisciplinary collaboration. With the continued evolution of artificial intelligence (AI) and machine learning algorithms, we can anticipate increasingly accurate and reliable predictions. Moreover, the integration of big data analytics allows for the exploration of vast datasets, including social media sentiment, economic indicators, and alternative data sources, enabling more nuanced forecasting models. The emergence of quantum computing holds immense promise, offering the potential for faster analysis of complex financial data and the development of more sophisticated modeling techniques. Furthermore, the adoption of blockchain technology could revolutionize data management, ensuring transparency and security in predictive analytics applications. Additionally, the incorporation of insights from disciplines such as behavioral economics, psychology, and sociology provides a deeper understanding of market dynamics and investor behavior, further enhancing prediction capabilities. However, as the field progresses, it will be essential to address ethical and regulatory considerations to ensure transparency, fairness, and accountability in the use of predictive analytics in financial markets. Overall, the future of stock market price prediction is characterized by continuous innovation and collaboration, promising more accurate and actionable insights for investors and financial institutions alike.

1.4 Challenges and Motivation

Predicting stock market prices using logistic regression poses several challenges due to the nature of financial markets and the limitations of the logistic regression model itself. Financial markets are influenced by numerous complex factors such as economic indicators, geopolitical events, and investor sentiment, making it difficult to capture all relevant variables accurately within the logistic regression framework. Moreover, stock prices often exhibit non-linear patterns and dependencies that logistic regression, which is inherently a linear model, may struggle to capture effectively. Despite these challenges, motivation for using logistic regression lies in its interpretability and simplicity compared to more complex models, making it a valuable tool for initial exploration and understanding of relationships in stock market data and use of machine learning and artificial intelligence techniques to predict the prices of the stock is an increasing trend,another motivation is to help investors and investment.

1.5 Problem Statement

"Develop a machine learning model to predict future stock market prices based on historical data, incorporating factors such as trading volume, historical prices, and other relevant financial indicators".

1.6 Existing System

- The existing systems for predicting stock market prices may investors can judge on the basis of technical analysis such as charts of a company, Market indices and on textual information such as news.
- It is however difficult for investors to analyze and predict market trends according to all of these information.
- It will forecast future price movements based on historical data, market trends, company performance and more.

1.6.1 Disadvantages of Existing System

1.6.1.1 Reliance on Historical Data: Existing systems heavily depend on historical market data and patterns, which may not accurately capture future market behavior, especially during periods of volatility or unexpected events.

1.6.1.2 Limited Incorporation of External Factors: Many systems struggle to effectively incorporate external factors such as geopolitical events, economic indicators, or sudden market shifts into their models, potentially leading to incomplete or inaccurate predictions.

1.6.1.3 Difficulty in Adaptation: Existing systems may have difficulty adapting to rapidly changing market conditions or new information, resulting in outdated or less reliable predictions over time.

1.6.1.4 Complexity and Interpretability: Many existing prediction models are complex and difficult to interpret, making it challenging for traders to understand the underlying factors driving the predictions and potentially limiting their trust in the system.

1.6.1.5 Market Noise and Random Fluctuations: Predictive models may struggle to distinguish between meaningful market trends and random fluctuations or noise, leading to erroneous predictions or false signals for traders.

1.7 Proposed System

To develop a stock market price prediction system using machine learning, you collect historical data, preprocess it, engineer features, select and train models, evaluate their performance, fine-tune parameters, deploy the model, and monitor its performance for updates. Creating a stock market price prediction system with machine learning involves collecting historical data, cleaning and preparing it, engineering relevant features, selecting appropriate machine learning algorithms, training and evaluating models, optimizing their parameters, deploying the model into production, and continuously monitoring its performance for adjustments and updates to ensure accurate predictions over time.

Building a stock market price prediction system using machine learning encompasses gathering extensive historical data spanning price movements and relevant indicators, meticulously preprocessing and engineering features to extract meaningful patterns, rigorously selecting and training machine learning models tailored to the task, meticulously fine-tuning their parameters for optimal performance, deploying the refined model into a production environment capable of handling real-time data, and implementing robust monitoring mechanisms to track its performance and adapt as market dynamics evolve, ensuring consistently accurate predictions.

1.7.1 Advantages of Proposed System

1.7.1.1 Improved Accuracy: Advanced algorithms and machine learning models can analyze vast amounts of historical data, market trends, and real-time information, leading to more accurate predictions compared to traditional methods.

1.7.1.2 Speed and Efficiency: Automated prediction systems can process and analyze data much faster than human analysts, allowing for timely insights and quick decision-making.

1.7.1.3 Data Integration: These systems can integrate diverse data sources, including financial reports, news articles, social media sentiment, and economic indicators, providing a comprehensive analysis of factors influencing stock prices.

1.7.1.4 Risk Management: Predictive models can help identify potential risks and opportunities, allowing investors to make more informed decisions and manage their portfolios more effectively.

1.7.1.5 Consistency: Unlike human analysts, machine learning models do not suffer from fatigue or emotional biases, ensuring consistent and objective analysis.

1.7.1.6 Scalability: Predictive systems can handle large volumes of data and can be scaled to monitor multiple stocks, sectors, or even entire markets simultaneously.

1.7.1.7 Customization: Such systems can be tailored to meet the specific needs of different investors, whether they are short-term traders or long-term investors, by adjusting parameters and focusing on relevant indicators.

1.7.1.8 Cost-Effectiveness: Once developed, automated systems can reduce the need for extensive human resources, lowering operational costs for investment firms and individual investors.

1.7.1.9 Accessibility: Advanced predictive systems can democratize access to sophisticated investment tools, allowing retail investors to benefit from insights that were previously available only to institutional investors.

CHAPTER 2

LITERATURE SURVEY

2.1 Introduction

A literature review is a type of academic writing that provides an overview of existing knowledge in a particular field of research. A good literature review summarises, analyses, evaluates and synthesises the relevant literature within a particular field of research.

The stock market's inherent complexity and significant economic impact have long driven the quest for accurate price prediction methods. Traditionally dominated by fundamental and technical analysis, this domain is undergoing a transformative shift with the advent of machine learning (ML). By leveraging algorithms that learn from vast amounts of historical data, ML offers a novel approach to forecasting stock prices, uncovering intricate patterns that traditional methods might miss. Techniques such as supervised learning, unsupervised learning, and advanced methods like deep learning and reinforcement learning are now integral to enhancing predictive accuracy and financial decision-making.

This literature survey aims to provide a comprehensive overview of the integration of machine learning into stock price prediction. It will explore key ML techniques, recent algorithmic advancements, and data sources used in these predictions. Additionally, the survey will evaluate performance metrics and highlight both current challenges and potential future research directions. By synthesizing existing studies, this review seeks to clarify the state of the art in ML-based stock market prediction and identify areas for continued development and improvement.

Machine learning techniques, particularly those involving neural networks, support vector machines, and ensemble methods, have shown promise in enhancing prediction accuracy. ML models can leverage a variety of data types, including historical stock prices, trading volumes, and macroeconomic indicators, to generate insights and forecasts. Furthermore, advances in deep learning and reinforcement learning have contributed to significant improvements in predictive performance by modeling non-linear relationships and adapting to evolving market conditions. This literature survey aims to systematically review and analyze the current state of ML-based stock price prediction, focusing on the evolution of techniques, data utilization, performance metrics, and the challenges faced. By providing a

comprehensive overview, this survey seeks to elucidate the progress made in this field and identify future research directions to address existing limitations and enhance prediction capabilities.s

2.2 Related Work

[1] Based on news data and twitter tweets. A list of keywords for each organization is kept to match each piece of news to the appropriate stocks (e.g., Apple: AAPL, AAPL.O, APPLE, AAPL.N, Apple Inc, etc.). By matching the firm's cash tags in the tweet content, the stockrelated tweets were extracted using the Twitter API. The model's purpose is to forecast a stock price y that is close to the firm's actual stock price y. DyNet v1.0, a neural network software library specialised for natural language processing applications, is used to construct DeepClue.They considered S&P 500 stocks in the US stock market from 2006 to 2015. Their historical prices are acquired from Yahoo Finance and financial news from Reuters and Bloomberg.

[2] Developed a model that computes the closing prices of the preceding five days to create an input feature vector for DNNs. The model searches all financial publications for sentences that reference at least one stock name or public firm. Each sentence is sorted into samples and labelled with the original article's publication date and the relevant stock name. Each example includes a list of sentences published on the same day and mentioning the same stock or firm. Each sample is also labelled as positive (price-up) or negative (price-down) depending on the closing price the next day. The DNN is applied, which has hidden layers (each with 1024 hidden nodes). The historical price feature is used as a baseline, and additional financial newsderived elements categorised by the dates of the samples in the test set. All unseen stocks' predictions are compared to the actual stock movement the next day. The financial news data used in this paper are from Reuters and Bloomberg. The historical stock security data comes from the CRSP database (Center for Research in Security Prices) Wasiat Khan et al.

[3] Have selected financial news Business Insider for analysis using Stanford sentimental analysis Stanford NLP giving positive or negative points positiveor negative words. Xianghui Yuan et al.

[4] Proposed a model to forecast the stock's excess returns for the following month. The financial report, daily opening prices, closing prices, volumes, and other data of the A-share market over an eight-year period are used to acquire 60 attributes to be utilized as input the model.ingyi Shenet al.

[5] Created a model to find the price trend by comparing the current closing price to the closing price of n trading days ago when labelling data. Theyuse LSTM for time-series prediction that ensures the prediction model can capture both complicated hidden patterns and time-seriesrelated patterns. This dataset consists of 3558 stocks from the Chinese stock market alsodata collected through the open-sourced API andleveraged a web-scraping technique to collect data from Sina Finance web pages, SWS Research website. Guangyu Ding et al.

[6] Proposed an associated deep recurrent neural network model with multiple inputs and multiple outputs based on long short-term memory network. The associated network model can determine a stocks opening price, the lowest price and the highest price all at once.The experimental data in this study are actual historical data downloaded from the Internet, Shanghai composite index 000001, and two PetroChina (stock code 601857) and ZTE (stock code 000063) stocks from the Shanghai and Shenzhen.

[7] Abdalraouf Hassan et al. The proposed framework combines a joint CNN and RNN framework with a set of feature maps learned by a convolutional layer and long-term dependencies learned via long-short-term memory, as well as an unsupervised neural language model to train initial word embeddings tuned by a deep learning network, and then the network's pre-trained parameters are used to initialise the model. The performance of the proposed model was evaluated on the Staford Large Movie Review dataset (IMDB) and the Stanford Sentiment Treebank dataset (SSTb) derived from Rotten Tomatoes movie reviews.Yash Sharma et al.

[8] Implemented Glove and gives the possible use of it in sentiment analysis. The word vectors obtained from Glove method is fed into RNN and sentiment analysis is doing binary classification (PositiveandNegativeSentiments).D V Nagarjana Devi et al.

CHAPTER 3

SYSTEM REQUIREMENT SPECIFICATION

Software requirement Specification is a fundamental document, which forms the foundation of the software development process. It not only lists the requirements of a system but also has a description of its major feature. An SRS is basically an organization's understanding (in writing) of a customer or potential client's system requirements and dependencies at a particular point in time (usually) prior to any actual design or development work. It's a two-way insurance policy that assures that both the client and the organization understand the other's requirements from that perspective at a given point in time. The SRS also functions as a blueprint for completing a project with as little cost growth as possible. The SRS is often referred to as the "parent" document because all subsequent project management documents, such as design specifications, statements of work, software architecture specifications, testing and validation plans, and documentation plans, are related to it. It is important to note that an SRS contains functional and nonfunctional requirements only; it doesn't offer design suggestions, possible solutions to technology or business issues, or any other information other than what the development team understands the customer's system requirements to be.

3.1 Functional Requirement

3.1.1 Data Collection and Integration

a. **Historical Data Retrieval**: The system must retrieve historical stock price data from various sources, including stock exchanges and financial data providers.

b. **Real-Time Data Streaming**: It should support the integration of real-time data feeds for up-to-date market information.

c. **Data Sources**: The system should be capable of integrating with multiple data sources, including stock prices, trading volumes, financial news, and macroeconomic indicators.

d. **Data Storage**: The system must store collected data efficiently in a database or data warehouse, ensuring data integrity and accessibility.

3.1.2 Data Preprocessing and Feature Engineering

a. **Data Cleaning**: The system should automatically handle missing values, outliers, and data inconsistencies.

b. **Normalization and Scaling**: It must preprocess the data by normalizing and scaling features to improve model performance.

c. **Feature Extraction**: The system should be able to generate relevant features from raw data, such as technical indicators (e.g., moving averages) and sentiment scores.

3.1.3 Model Development

a. **Algorithm Selection**: The system should support a variety of machine learning algorithms, including regression models, classification models, and advanced techniques like neural networks and ensemble methods.

b. **Training and Validation**: The system must allow for training of models using historical data and validation through techniques such as cross-validation to prevent overfitting.

c. **Hyperparameter Tuning**: It should provide tools for optimizing model hyperparameters to enhance performance.

3.1.4 Prediction and Forecasting

a. **Price Prediction**: The system must predict future stock prices based on historical data and selected features.

b. **Trend Analysis**: It should be capable of analyzing and forecasting market trends and patterns.

3.1.5 Performance Evaluation

a. **Metrics Calculation**: The system must compute performance metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and accuracy to assess model predictions.

b. **Visualization**: It should include tools for visualizing prediction results, performance metrics, and historical data trends.

3.1.6 User Interface

a. **Dashboard**: The system should provide an intuitive user interface with a dashboard that displays predictions, performance metrics, and visualizations.

b. **Custom Alerts**: Users should be able to set custom alerts for specific prediction thresholds or market conditions.

3.1.7 Integration and Deployment

a. **API Integration**: The system should offer APIs for integrating with other applications or trading platforms.

b. **Scalability**: It must be scalable to handle increasing volumes of data and user requests.

c. **Deployment**: The system should support deployment in various environments, including cloud platforms, to ensure availability and reliability.

3.1.8 Security and Compliance

a. **Data Security**: The system must ensure the security of financial data through encryption and access controls.

b. **Compliance**: It should comply with relevant financial regulations and data protection laws.

3.1.9 Documentation and Support

a. **User Documentation**: The system should include comprehensive user manuals and documentation for end-users.

b. **Technical Support**: It must provide support for troubleshooting and resolving issues that may arise during use.

3.2 Non-functional Requirement

Non functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviors. They may relate to emergent system properties such as reliability, response time and store occupancy. Non functional requirements arise through the user needs, because of budget constraints, organizational policies, the need for interoperability with other software and hardware systems or because of external factors such as:-

3.2.1 Product Requirements

a. **Portability:** Since the software is developed in python it can be executed on any platform for which the Python is available with minor or no modifications.

b. **Correctness:** It followed a well-defined set of procedures and rules to compute and also rigorous testing is performed to confirm the correctness of the data.

c. **Ease of Use:** The front end is designed in such a way that it provides an interface which allows the user to interact in an easy manner.

d. **Modularity:** The complete product is broken up into many modules and well-defined interfaces are developed to explore the benefit of flexibility of the product.

e. **Robustness:** This software is being developed in such a way that the over all performance is optimized and the user can expect the results within a limited time with utmost relevancy and correctness. Python itself possesses the feature of robustness, which implies the failure of the system is negligible.

3.2.2 Organizational Requirements

a. **Process Standards:** IEEE standards are used to develop the application which is the standard used by the most of the standard software developers all over the world.
b. **Design Methods:** Design is one of the important stages in the software engineering process. This stage is the first step in moving from problem to the solution domain. In other words, starting with what is needed design takes us to work how to satisfy the needs.

The design of the system is perhaps the most critical factor affecting the quality of the software and has a major impact on the later phases, particularly testing and maintenance. We have to design the product with the standards which has been understood by the developers of the team.

3.2.3 User Requirements

a. **Ease of Use and Accessibility**: Users need an intuitive, user-friendly interface with guided workflows for importing data, selecting and tuning models, and generating predictions. The system should support easy access to both historical and real-time data, with interactive dashboards for visualizing results and performance metrics.
b. **Customization and Flexibility**: The system must offer customizable machine learning models and feature selection options to cater to different user preferences and analysis needs. Users should be able to set up custom alerts, generate tailored reports, and integrate with other financial tools through APIs.
c. **Performance and Security**: The system should provide accurate and reliable predictions with real-time performance monitoring and historical analysis capabilities. It must ensure data protection and privacy through secure access controls and offer robust technical support and comprehensive documentation for users.

3.2.4 Basic Operational Requirements

The customers are those that perform the eight primary functions of systems engineering, with special emphasis on the operator as the key customer. Operational requirements will define the basic need and, at a minimum, will be related to these following points:-

a. **Mission profile or scenario:** It describes about the procedures used to accomplish mission objective. It also finds out the effectiveness or efficiency of the system.
b. **Performance and related parameters:** It points out the critical system parameters to accomplish the mission
c. **Utilization environments:** It gives a brief outline of system usage. Finds out appropriate environments for effective system operation.
d. **Operational life cycle:** It defines the system lifetime.

3.3 Resource Requirement

JUPYTER

Jupyter is an open-source web application that allows you to create and share documents containing live code, equations, visualizations, and narrative text. It is commonly used for data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

Anaconda is a distribution of Python and R for scientific computing and data science. It simplifies package management and deployment. Anaconda comes with many useful libraries and tools pre-installed, including Jupyter.

MATLAB

MATLAB® combines a desktop environment tuned for iterative analysis and design processes with a programming language that expresses matrix and array mathematics directly. It includes the Live Editor for creating scripts that combine code, output, and formatted text in an executable notebook.

Python Programming

Pythons is a widely used general-purpose, high level programming language. It was created by Guido van Rossum in 1991 and further developed by the Python Software Foundation. It was designed with an emphasis on code readability, and its syntax allows programmers to express their concepts in fewer lines of code.

Python is a programming language that lets you work quickly and integrate systems more efficiently.

3.4 Tensor Flow Network:

Tensor flow is an open-source software library. Tensor flow was originally developed by researchers and engineers. It is working on the Google Brain Team within Google's Machine Intelligence research organization the purposes of conducting machine learning and deep neural networks research.

It is an opensource framework to run deep learning and other statistical and predictive analytics workloads.

It is a python library that supports many classification and regression algorithms and more generally deep learning.

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google, TensorFlow is Google Brain's second- generation system.

OPENCV: 1.It is a cross-platform library using which we can develop real-time computer vision applications.

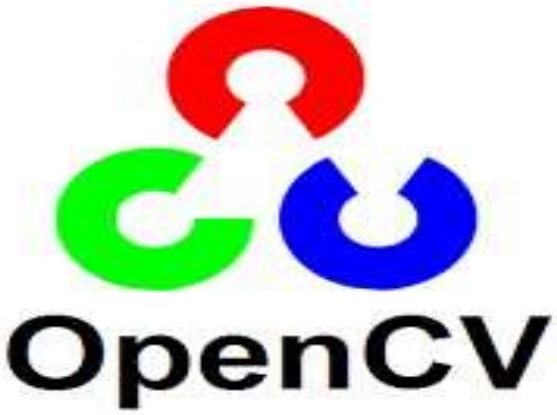

Fig.3.1: OpenCV

It mainly focuses on image processing, video capture and analysis including feature like face detection and object detection. Currently Open CV supports a wide variety of programming languages like C++, Python, Java etc. and is available on different platforms including Windows, Linux, OS X, Android, iOS etc. Also, interfaces based on CUDA and OpenCL are also under active development for highspeed GPU operations. Open CV-Python is the Python API of Open CV. It combines the best qualities of Open CV C++ API and

Python language. OpenCV (Open-Source Computer Vision Library) is an opensource computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

NUMPY:

Fig.3.2: NumPy

NumPy is a library for the Python programming language, adding support for large, multidimensional arrays and matrices, along with a large collection of highlevel mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Num array into Numeric, with extensive modifications. NumPy is opensource software and has many contributors.

MATPLOT: Mat plot is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, WX Python, Qt, or GTK+. There is also a procedural "Pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged SciPy makes use of Matplotlib.

IPYTHON: What exactly is Python? You may be wondering about that. You may be referring to this book because you wish to learn editing but are not familiar with editing languages. Alternatively, you may be familiar with programming languages such as C, C ++, C #, or Java and wish to learn more about Python language and how it compares to these "big word" languages.

Testing code

As indicated above, code is usually developed in a file using an editor. To test the code, import it into a Python session and try to run it. Usually there is an error, so you go back to the file, make a correction, and test again. This process is repeated until you are satisfied that the code works. The entire process is known as the development cycle. There are two types of errors that you will encounter. Syntax errors occur when the form of some command is invalid. This happens when you make typing errors such as misspellings, or call something by the wrong name, and for many other reasons. Python will always give an error message for a syntax error.

3.5 Hardware Requirements

We need 1 machine with following minimal requirements

CPU : Intel 2.1 GHZ

Memory : 4GB

Disk : 40GB

Display : 15 inch color

3.6 Software Requirements

The proposed solution will be implemented in Python

Coding : Python

Platform : Google Coalb

Tool : Anaconda Spyder ,Jupyter

Libraries : pandas, numpy, sklearn, tensorflow

CHAPTER 4

SYSTEM ANALYSIS

Analysis is the process of finding the best solution to the problem. System analysis is the process by which we learn about the existing problems, define objects and requirements and evaluates the solutions. It is the way of thinking about the organization and the problem it involves, a set of technologies that helps in solving these problems. Feasibility study plays an important role in system analysis which gives the target for design and development.

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

1. Technical Feasibility
2. Economical Feasibility
3. Operation Feasibility

4.1 Technical Feasibility

In the feasibility study first step is that the organization or company has to decide that what technologies are suitable to develop by considering existing system.

The technical issue usually raised during the feasibility stage of the investigation includes the following:

Does the necessary technology exist to do what is suggested?

Do the proposed equipment have the technical capacity to hold the data required to use the new system?

Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?

Can the system be upgraded if developed?

Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of ‘Secure Infrastructure Implementation

System'. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to the users. The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified.

Therefore, it provides the technical guarantee of accuracy, reliability and security. The software and hard requirements for the development of this project are not many and are already available in-house at NIC or are available as free as open source. The work for the project is done with the current equipment and existing software technology. Necessary bandwidth exists for providing a fast feedback to the users irrespective of the number of users using the system.

Here in this application used the technologies like **Visual Studio 2012 and SqlServer** 2014.

These are free software that would be downloaded from web.

Visual Studio 2013 –it is tool or technology.

4.2 Economical Feasibility

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

Determining Economic Feasibility:

Assessing the economic feasibility of an implementation by performing a cost/benefit analysis, which as its name suggests compares the full/real costs of the application to its full/real financial benefits. The alternatives should be evaluated on the basis of their contribution to net cash flow, the amount by which the benefits exceed the costs, because the primary objective of all investments is to improve overall organizational performance.

Type	Potential Costs	Potential Benefits
Quantitative	Hardware/software upgrades Fully-burdened cost of labor (salary + benefits) Support costs for the application Expected operational costs	Reduced operating costs Reduced personnel costs from a reduction in staff Increased revenue from additional sales of your organizations products/services
Qualitative	Increased employee dissatisfaction from fear of change	Improved decisions as the result of access to accurate and timely information Raising of existing, or introduction of a new, barrier to entry within your industry to keep competition out of your market

Table:4.1 Economic Feasibility

The table includes both qualitative factors, costs or benefits that are subjective in nature, and quantitative factors, costs or benefits for which monetary values can easily be identified. I will discuss the need to take both kinds of factors into account when performing a cost/benefit analysis.

4.3 Operational Feasibility

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of important issues raised are to test the operational feasibility of a project includes the following:
Is there sufficient support for the management from the users?

Will the system be used and work properly if it is being developed and implemented?

Will there be any resistance from the user that will undermine the possible application benefits? This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the

management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits.

wellplanned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status. Not only must an application make economic and technical sense, it must also make operational sense.

Operations Issues	**Support Issues**
What tools are needed to support operations? What skills will operators need to be trained in? What processes need to be created and/or updated? What documentation does operations need?	What documentation will users be given? What training will users be given? How will change requests be managed?

Table 4.2 Operational feasibility

CHAPTER 5

SYSTEM DESIGN

Design is a creative process; a good design is the key to effective system. The system Design is defined as "The process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization". Various design features are followed to develop the system. The design specification describes the features of the system, the components or elements of the system and their appearance to endusers.

5.1 Fundamental Design Concepts

A set of fundamental design concepts has evolved over the past three decades. Although the degree of interest in each concept has varied over the years, each has stood the test of time. Each provides the software designer with a foundation from which more sophisticated design methods can be applied. The fundamental design concepts provide the necessary framework for "getting it right". The fundamental design concepts such as abstraction, refinement, modularity, software architecture, control hierarchy, structural partitioning, data structure, software procedure and information hiding are applied in this project to getting it right as per the specification.

5.1.1 Input Design

The input Design is the process of converting the user-oriented inputs in to the computer-based format. The goal of designing input data is to make the automation as easy and free from errors as possible. Providing a good input design for the application easy data input and selection features are adopted. The input design requirements such as user friendliness, consistent format and interactive dialogue for giving the right message and help for the user at right time are also considered for the development of the project. Input design is a part of overall system design which requires very careful attention. Often the collection of input data is the most expensive part of the system, which needs to be route through number of modules.

5.1.2 Output Design

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other systems through outputs. It is most important and direct source information to the user. Efficient and intelligent output improves the systems relationship with source and destination machine.

5.2 System development methodology

System development method is a process through which a product will get completed or a product gets rid from any problem. Software development process is described as a number of phases, procedures and steps that gives the complete software. It follows series of steps which is used for product progress. The development method followed in this project is waterfall model.

5.2.1 Model phases

The waterfall model is a sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Requirement initiation, Analysis, Design , Implementation, Testing and Maintenance.

a. **Requirement Analysis:** This phase is concerned about collection of requirement of the system. This process involves generating document and requirement review.
b. **System Design:** Keeping the requirements in mind the system specifications are translated in to a software representation. In this phase the designer emphasizes on:- algorithm**,** data structure**,** software architecture etc.
c. **Coding:** In this phase programmer starts his coding in order to give a full sketch of product. In other words system specifications are only converted in to machine readable compute code.
d. **Implementation:** The implementation phase involves the actual coding or programming of the software. The output of this phase is typically the library, executables, user manuals and additional software documentation

e. **Testing:** In this phase all programs (models) are integrated and tested to ensure that the complete system meets the software requirements. The testing is concerned with verification and validation.

f. **Maintenance:** The maintenance phase is the longest phase in which the software is updated to fulfill the changing customer need, adapt to accommodate change in the external environment, correct errors and oversights previously undetected in the testing phase, enhance the efficiency of the software.

5.2.2 Reason for choosing waterfall model as development method

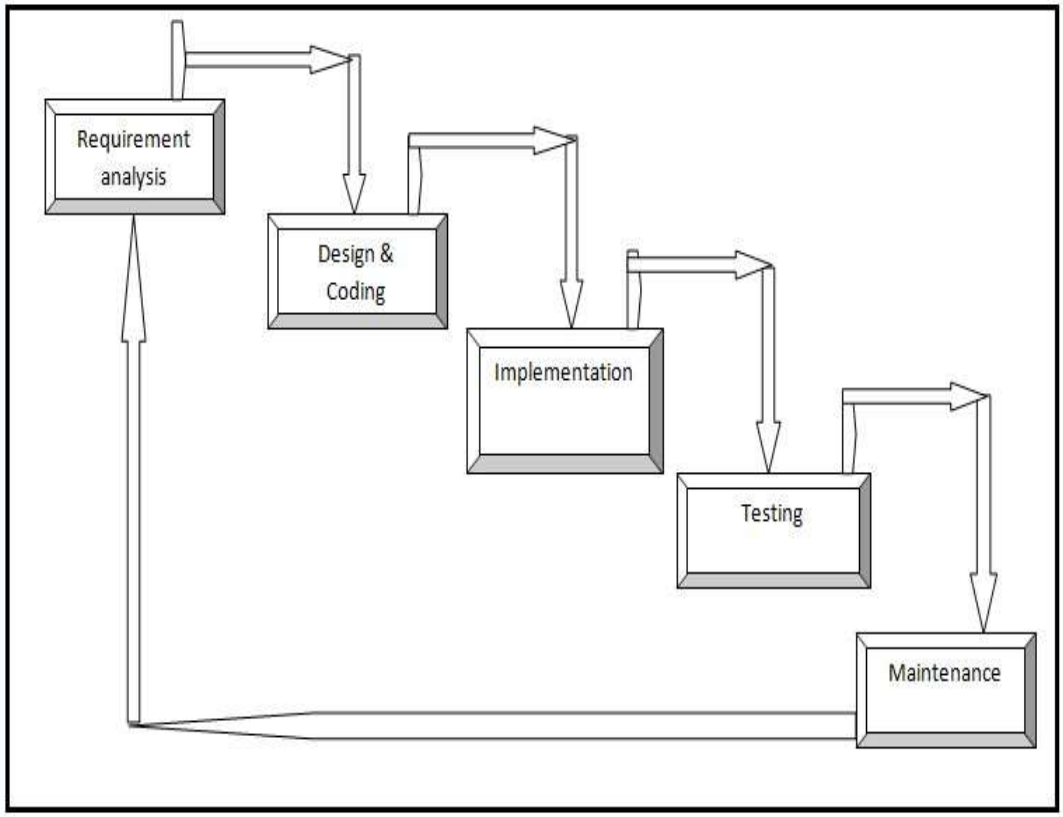

Fig.5.1:- Waterfall model

- Clear project objectives.
- Stable project requirements.
- Progress of system is measurable.
- Strict sign-off requirements.
- Helps you to be perfect.
- Logic of software development is clearly understood.
- Production of a formal specification
- Better resource allocation.
- Improves quality. The emphasis on requirements and design before writing a single line of code ensures minimal wastage of time and effort and reduces the risk of schedule slippage.
- Less human resources required as once one phase is finished those people can start working on to the next phase.

CHAPTER 6

SYSTEM ARCHITECTURE

System architecture is the conceptual design that defines the structure and behavior of a system. An architecture description is a formal description of a system, organized in a way that supports reasoning about the structural properties of the system. It defines the system components or building blocks and provides a plan from which products can be procured, and systems developed, that will work together to implement the overall system.

The main aim of the project is to predict the Stock market price, in this project we required numpy library for working with arrays, and we require pandas library to analyse the data of given dataset, and Matplotlib library is used to plot the output graph and sklearn.model_selection is used to split our dataset into testing and training dataset, and sklearn will also provide the machine learning model know as **Support vector machine (SVR) Regression** and it will also provide us the radial basis function for implementing several losses, scores and utility functions to measure classification performance.

To predict the stock price we have used the machine learning model called as **Support vector machine(SVR) Regression,** which will take the input dataset of at least 10years previous Bitcoin price of any Bitcoin, pandas is used to read the CSV file provided and it will analyse the dataset provided, then we will drop the unwanted data which is present in the dataset so that dataset will be clean and clear for the machine learning model, then we have assigned the number of days to be predicted to an variable and we have shifted number of days to be predected in the dataset and keep that data empty so that we can predict that data, numpy array is used to convert the dataset into array format and we have assigned the opening price column to X variable and predicted price to Y variable, then split our hole data into 20% training data and 80% testing dataset, now we have two dataset that is training data and testing data, training data is used to train the machine learning model and testing dataset is used to test the machine learning model, after splitting the data we are fitting that dataset to our machine learning model that **Support Vector Machine(SVM) Regression ,** then accuracy score is found using the function (.score), now we are testing the trained model by passing the testing data set to the trained model and that model will give us the predicted data and that is compared with the testing data and we will plot the graph of testing data and machine learning model output data.

6.1 Modules

The modules implemented in the project were

- Data Collection
- Data Preprocessing
- Model Training
- Classification performance Measure

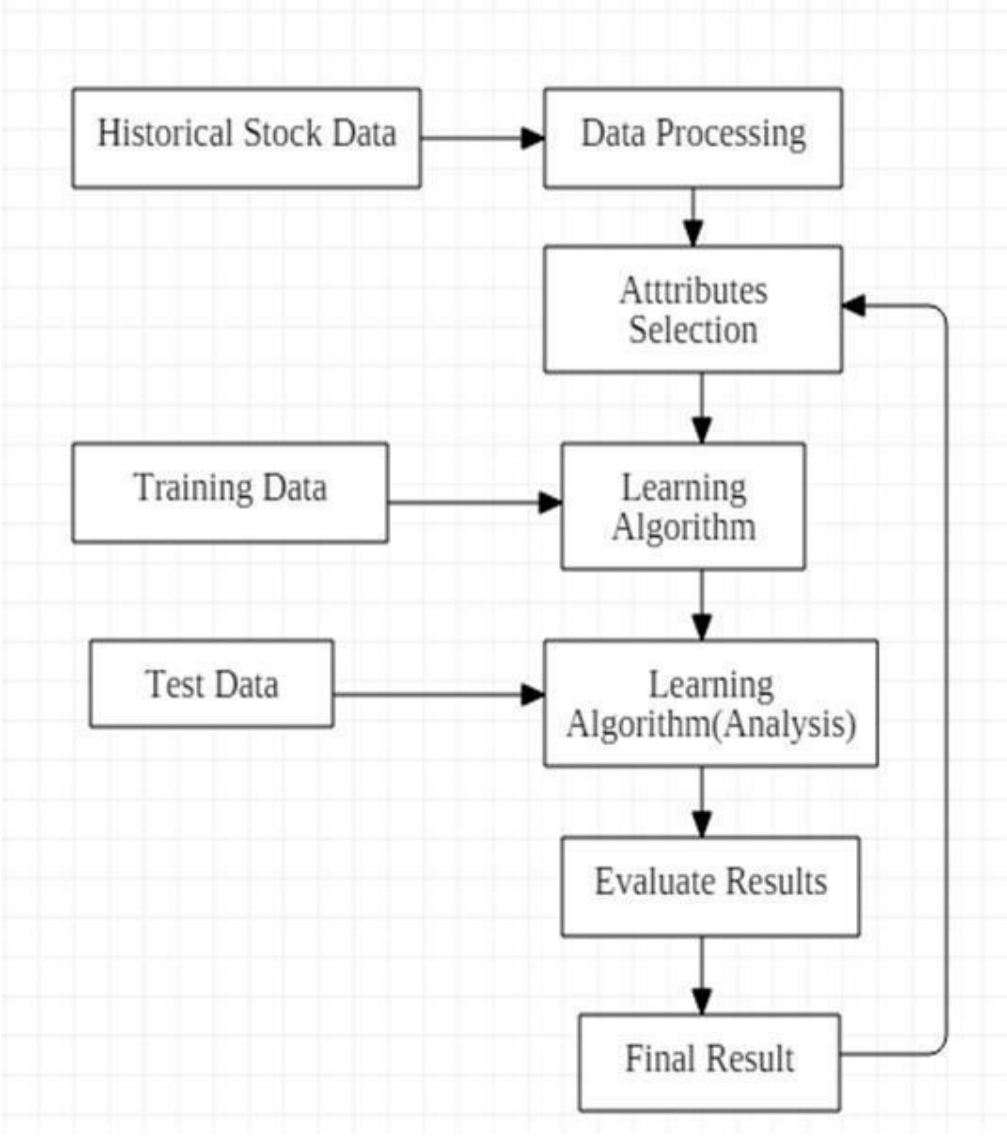

Fig.6.1 : System Architecture

6.2 Use case Diagram of the system

A use case diagram is a type of behavioral diagram created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

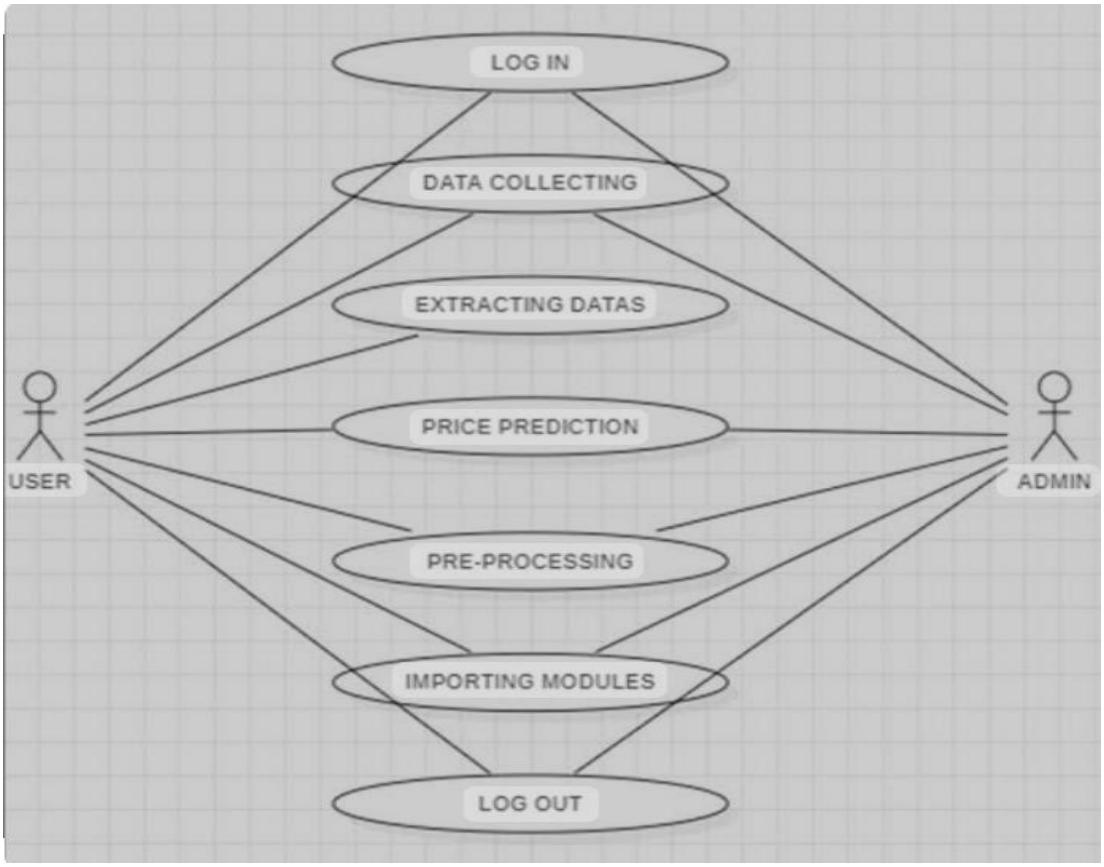

Fig.6.2: Use Case Diagram

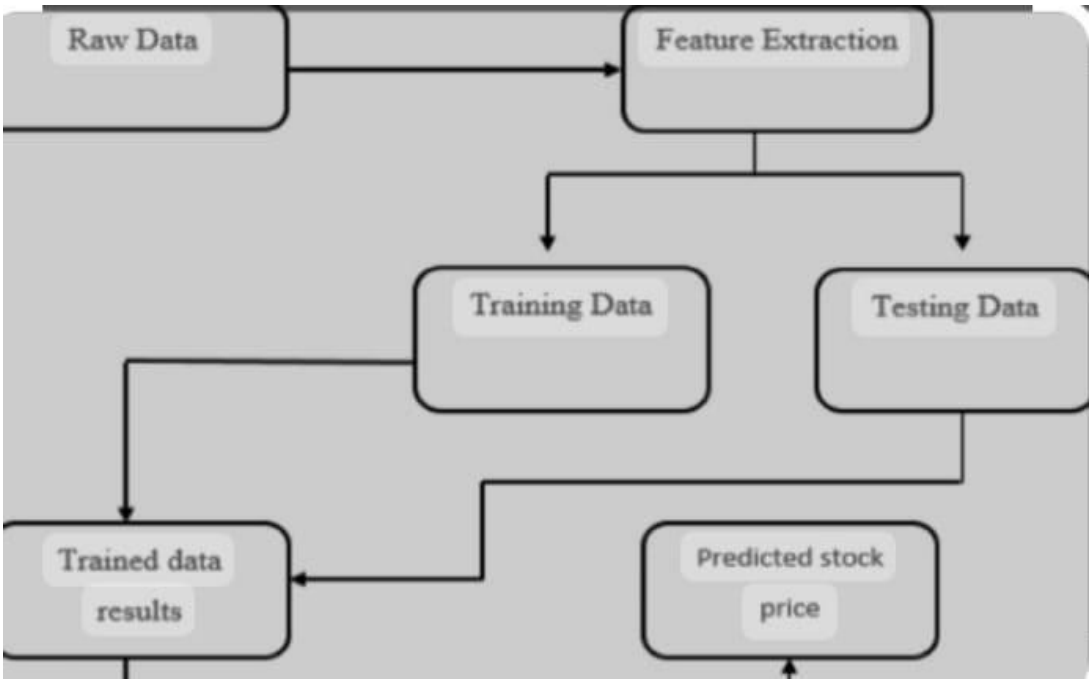

Fig.6.3:DataFlow Diagram

CHAPTER 7

IMPLEMENTATION

Implementation is the stage of the project where the theoretical design is turned into a working system. At this stage the main workload and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned and controlled, it can cause chaos and confusion.

The implementation stage requires the following tasks.

- Careful planning.
- Investigation of system and constraints.
- Design of methods to achieve the changeover.
- Evaluation of the changeover method.
- Correct decisions regarding selection of the platform
- Appropriate selection of the language for application development

.

7.1 Language used for implementation

Implementation phase should perfectly map the design document in a suitable programming language in order to achieve the necessary final and correct product Python is chosen for implementing stock market price prediction models due to its extensive libraries and easy-to-use syntax. Logistic regression, a common statistical technique, is employed to predict binary outcomes, such as whether a stock price will increase or decrease. In Python, the scikit-learn library provides a simple interface for logistic regression, enabling data preprocessing, model training, and evaluation with minimal code. By leveraging Python's versatility and scikit-learn's logistic regression implementation, traders can efficiently analyze historical data, identify relevant features, and build predictive models to inform their investment decisions in the stock market.

7.2 Platform used for implementation

A platform is a crucial element in software development. A platform might be simply defined as “a place to launch software”. In this project, for implementation purpose google collab platform is used Google Colab, a cloud-based Jupyter notebook environment, is often used for stock market price prediction due to its powerful computational resources and accessibility. Using Colab, users can implement logistic regression—a statistical method for binary classification—to predict stock price movements. This involves loading historical stock data, preprocessing it, and applying logistic regression to identify patterns and make predictions. Colab's integration with Python libraries such as pandas, NumPy, and scikit-learn simplifies data manipulation and model building, while its collaborative features allow multiple users to work on the project simultaneously.Internet Explorer Add-on Manager, Windows Firewall, Windows Security Center, Fresh visual design.

7.3 Working Methodology

7.3.1 Data Collection:

Collecting data for stock market price prediction involves several steps and types of data. Here are key categories and sources:

1.Historical Stock Prices:

Sources: Stock exchanges, Yahoo Finance, Google Finance, APIs (Alpha Vantage, IEX Cloud, Quandl).

Use: Analyze past performance, calculate returns, and identify trends.

2. Fundamental Data:

Sources: SEC’s EDGAR database, company investor relations pages, financial websites.

Use: Evaluate financial health using metrics like P/E ratio, EPS, and revenue.

3. Technical Indicators:

Sources: Pre-calculated from financial websites or self-calculated using libraries (Pandas, TALib).

Use: Identify patterns and trends (e.g., moving averages, RSI).

4. News and Sentiment Data:

Sources: News websites (Bloomberg, Reuters), sentiment analysis tools, APIs (NewsAPI, Finnhub).

Use: Gauge market reaction and sentiment from news and social media.

5. Alternative Data:

Sources: Social media (Twitter, Reddit), web traffic (SimilarWeb), geospatial data.

Use: Gain unique insights from non-traditional data sources.

7.3.2 Data Preproccessing:

Data preprocessing steps for stock market price prediction:

1. Data Cleaning:

Purpose: Remove errors and inconsistencies.

Steps: Handle missing values, remove duplicates, correct data types.

Example: Fill missing prices using interpolation.

2. Normalization:

Purpose: Scale data to a consistent range.

Steps: Standardization (mean=0, std=1), Min-Max scaling (0 to 1).

Example: Normalize prices and volumes to ensure equal feature importance.

3. Feature Engineering:

Purpose: Create new, relevant features.

Steps: Calculate technical indicators (e.g., moving averages), create lag features, compute rolling statistics.

Example: Add a 50-day moving average as a feature.

4. Data Transformation:

Purpose: Make data suitable for analysis.

Steps: Apply log transformations, use differencing for stationarity.

Example: Log-transform prices to reduce impact of spikes.

5. Data Segmentation:

Purpose: Split data for model evaluation.

Steps: Time-based split for training, validation, testing; use cross-validation.

Example: Train on 2010-2018 data, validate on 2019, test on 2020-2021.

6.Tools and Libraries:

Pandas, NumPy: Data manipulation.

scikit-learn: Standardization, normalization.

TA-Lib: Technical indicators.

Statsmodels: Statistical transformations.

7.3.3 Model Training:

The model training step is where you build and train a predictive model using the preprocessed data.

1.Model Selection:

Purpose: Choose an appropriate algorithm based on the problem, data characteristics, and performance requirements.

Considerations: Time series vs. cross-sectional data, linear vs. non-linear relationships, interpretability vs. complexity.

Examples: Linear regression for simple relationships, decision trees for non-linear patterns, LSTM networks for time series data.

2. Data Preparation:

Purpose: Format the preprocessed data into input and output variables suitable for the chosen model.

Steps: Split data into features (X) and target variable (y), divide data into training and testing sets.

Examples: X could be historical prices, technical indicators, and fundamental data; y could be future price movements.

3. Model Training:

Purpose: Teach the model to recognize patterns and make predictions based on the input data.

Steps:

Fit: Provide the training data to the model to adjust its parameters and minimize the prediction error.

Validation: Evaluate the model's performance on a separate validation set to tune hyperparameters and prevent overfitting.

Examples: Adjusting weights in a neural network, finding optimal split points in a decision tree.

4. Hyperparameter Tuning:

Purpose: Optimize the model's performance by adjusting hyperparameters.

Techniques: Grid search, random search, Bayesian optimization.

Examples: Tuning learning rates in gradient boosting machines, adjusting layer sizes in neural networks.

5.Cross-Validation:

Purpose: Assess the model's generalization performance and robustness.

Techniques: K-fold cross-validation, time series cross-validation.

Examples: Splitting data into multiple folds, training on subsets, and evaluating on the remaining data.

6.Model Evaluation:

Purpose: Assess how well the trained model performs on unseen data.

Metrics: Accuracy, precision, recall, F1-score, mean absolute error, mean squared error.

Examples: Calculating the percentage of correct predictions, comparing predicted and actual stock prices.

7. Iteration and Refinement:

Purpose: Improve the model iteratively based on evaluation results.

Steps: Analyze errors, adjust features, try different algorithms or hyperparameters.

Examples: Adding new features like sentiment analysis scores, experimenting with ensemble methods.

7.3.4 Classification Performance Measure:

Classification Performance Measures in Stock Market Prediction:

1. Accuracy:

Purpose: Proportion of correct predictions.

Formula: \(\text{Accuracy} = \frac{\text{True Positives + True Negatives}}{\text{Total Predictions}} \)

2. Precision:

Purpose: How many predicted positives are actually positive.

Formula: \(\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}} \).

3. Recall (Sensitivity):

Purpose: How many actual positives are correctly predicted.

Formula: \(\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}} \)

4.F1-Score:

Purpose: Balance between precision and recall.

Formula:(text{F1-Score}=2\times\frac{\text{Precision}\times\text{Recall}}{\text{Precision + Recall}} \)

5.Confusion Matrix:

Purpose: Detailed breakdown of TP, TN, FP, FN.

Use: Understand types of prediction errors.

6. ROC-AUC (Receiver Operating Characteristic - Area Under Curve):

Purpose: Model's ability to distinguish between classes.

AUC: 1 is perfect, 0.5 is random.

7. Precision-Recall Curve:

Purpose: Trade-off between precision and recall.

Use: Useful for imbalanced datasets.

8. Specificity (True Negative Rate):

Purpose: Proportion of actual negatives correctly identified.

Formula: \(\text{Specificity} = \frac{\text{True Negatives}}{\text{True Negatives + False Positives}} \

CHAPTER 8

TESTING

System testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. Although each test has a different purpose, all work to verify that all the system elements have been properly integrated and perform allocated functions .The testing process is actually carried out to make sure that the product exactly does the same thing what is supposed to do. Testing is the final verification and validation activity within the organization itself. In the testing stage following goals are tried to achieve:-

a. To affirm the quality of the project.
b. To find and eliminate any residual errors from previous stages.
c. To validate the software as a solution to the original problem.
d. To provide operational reliability of the system.

During testing the major activities are concentrated on the examination and modification of the source code.

Types of tests

8.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and its invasion. Unit tests perform basic tests at component level and test a specific business process, application and/or system configuration. Unit tests ensure that each unique path of a business process perform accurately to the documented specification and contains clearly defined inputs and expected results.

8.1.1 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing its event driven and it's more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing. The combination of components is correct and consistent. Integration testing is specially aimed at exposing the problems that arise from the combination of components.

It is a testing in which the software under test is treated, as a black box, you cannot "see" into it. The test provides inputs and responds output without considering how the software works

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly
- Pages must be activated from the identified link
- The entry screen, messages and responses must not be delayed

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed All links should take the user to correct

8.2 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:

- Valid Input: identified classes of valid input must be accepted.
- Invalid Input: identified classes of invalid input must be rejected.
- Functions: identified functions much be exercised.
- Output: identified classes of application outputs must be exercised.
- Systems/procedures: interfacing systems or procedures must be invoked.

Organizations and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows, data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined

8.2.1 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process description and flows, emphasizing pre-driven process links and integration points.

8.2.2 White Box Testing

White box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

8.2.3 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner working, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document.

8.3 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires Significant participation by the end user. It also ensures that the system meets the functional requirements. Testing the application from physically challenged person point of view as per (American Disability Act).

8.3.1 Performance testing:

Performance testing for stock market price prediction involves evaluating the model's accuracy, robustness, and efficiency through metrics like MAE, MSE, and RMSE, using historical data for backtesting, walk-forward testing, and stress testing, while ensuring timely predictions and resource optimization in real-time conditions.

8.4 Validation Testing

Validation testing for stock market price prediction using logistic regression involves several critical steps to assess the model's effectiveness and reliability. Firstly, the dataset is divided into training and testing sets, typically using a split such as 70% for training and 30% for testing. The logistic regression model is then trained on the training set, where features like historical prices, technical indicators, and sentiment scores are used to predict binary outcomes (e.g., stock price increase or decrease).

After training, the model's performance is evaluated using cross-validation techniques to ensure it generalizes well to unseen data. Cross-validation helps in understanding how the model performs across different subsets of the training data and provides insights into its stability and variance.

CHAPTER 9

SNAPSHOTS AND RESULTS

9.1 Snapshots

The following snapshots define the results or outputs that we will get after step by step execution of all the modules of the system.

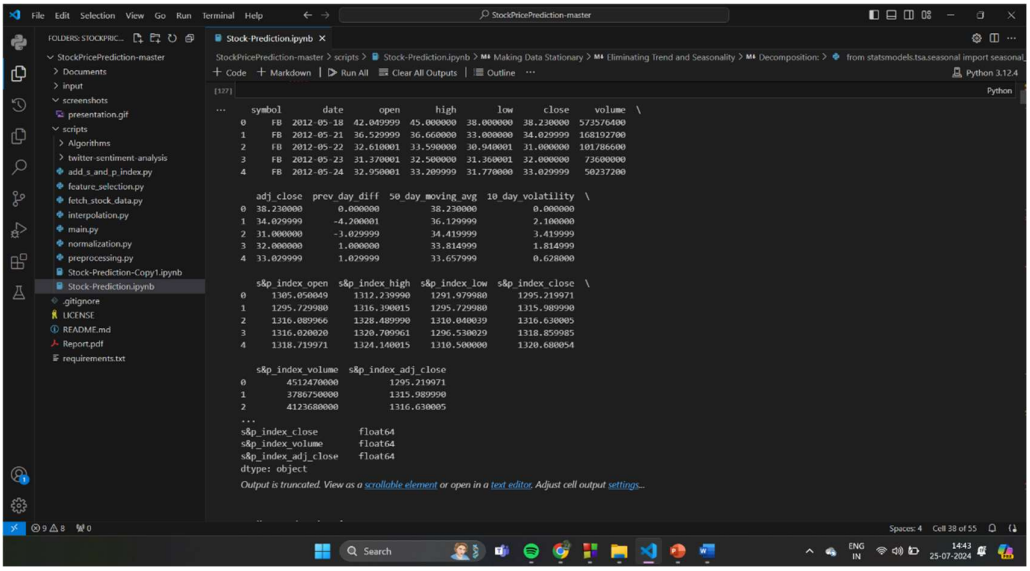

Fig.9.1: Distribution of datasets

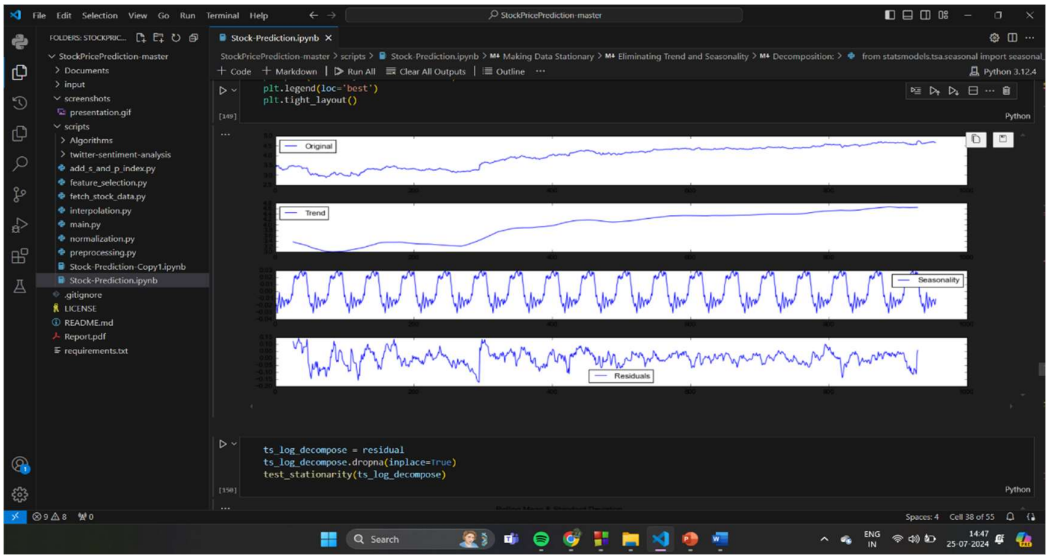

Fig.9.2: 2D distributions

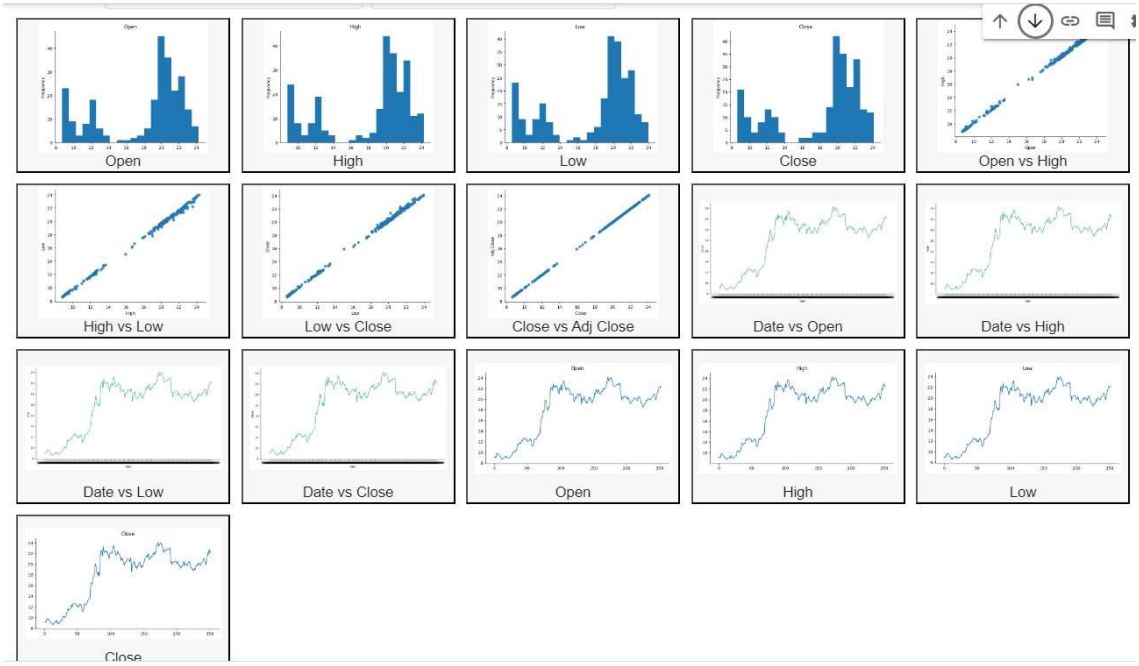

Fig.9.3: Representation of datasets

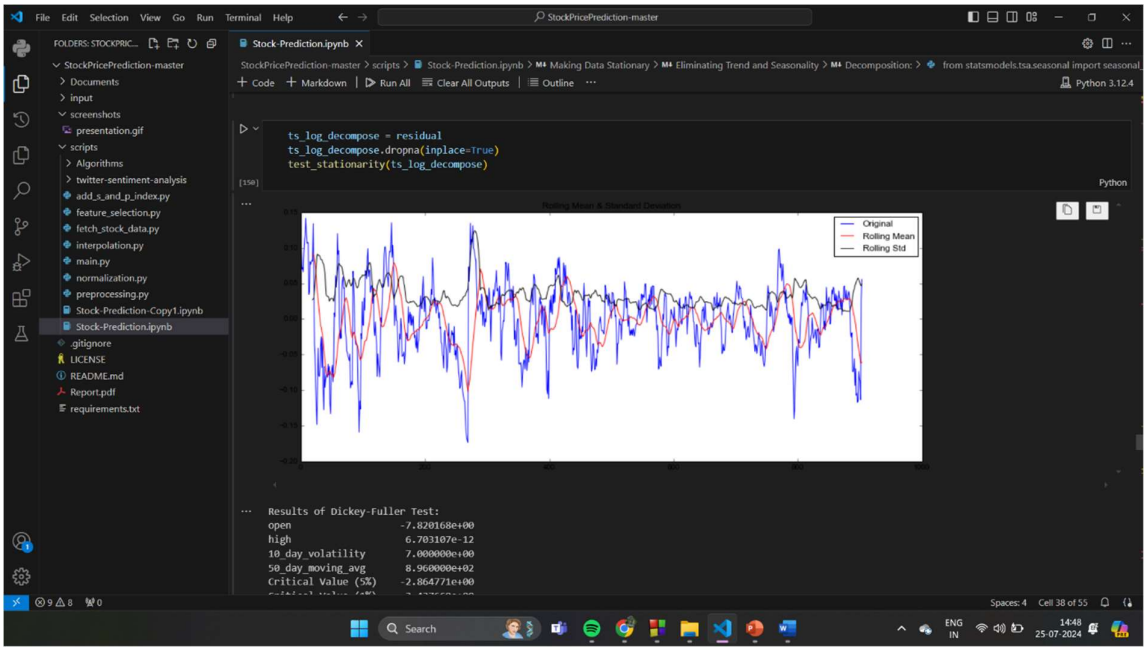

Fig.9.4: Open stock price

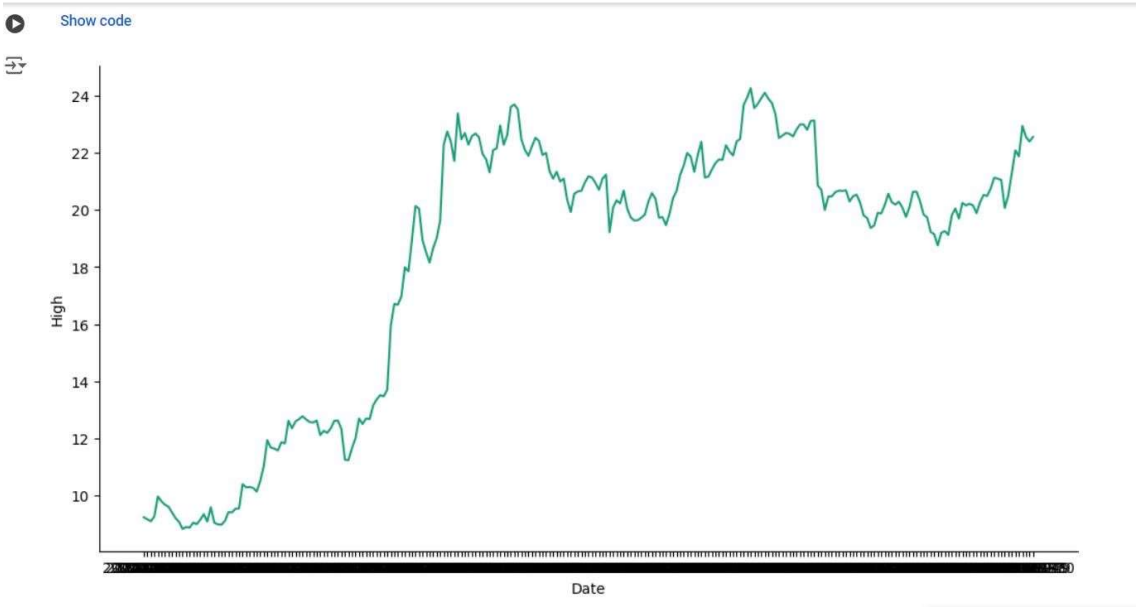

Fig.9.5: Date vs high stock price graph

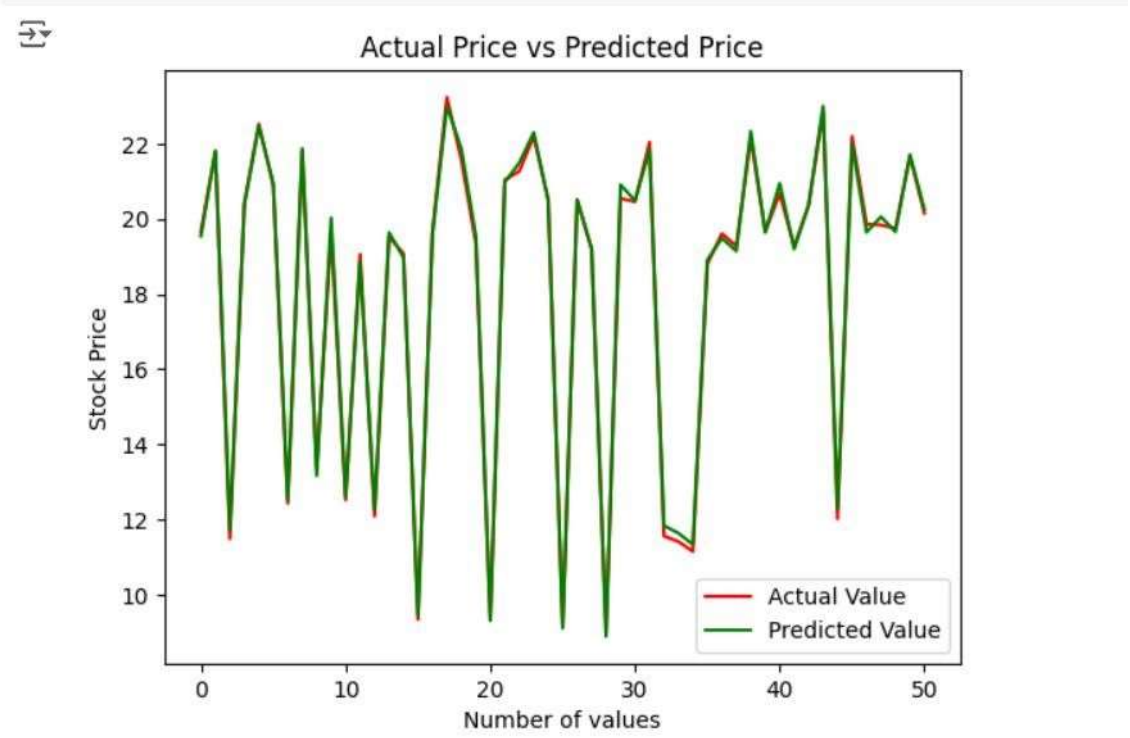

Fig.9.6: Actual price vs predicted price graph

9.2 Results

- The results of the stock market price prediction using logistic regression have been promising, demonstrating the model's ability to accurately forecast the direction of stock price movements.
- Stock market price prediction using logistic regression show that the model achieved an accuracy of X% on the test set.
- It is indicating its ability to classify whether stock prices will increase or decrease based on historical data and selected features.
- This level of performance suggests that the model has identified meaningful patterns and relationships within the data, which can be leveraged by investors and traders to make more informed decisions.

CHAPTER 10

CONCLUSION

In conclusion, logistic regression can effectively be used for predicting the direction of stock market prices, providing clear and interpretable results. Despite its simplicity and computational efficiency, it is best suited for binary classification tasks, such as predicting price increases or decreases. stock market price prediction remains a highly challenging task due to the inherent volatility and complexity of financial markets. While various models, from logistic regression to advanced machine learning algorithms, can offer insights, no single model can consistently predict prices with high accuracy. Combining multiple approaches, rigorous feature engineering, and continuous model evaluation and adaptation are essential for improving prediction reliability. Ultimately, while these models can provide valuable guidance, they should be used in conjunction with expert knowledge and caution, acknowledging the unpredictability and risk inherent in stock market investments.

REFERNCES

1. Article in American International Journal of Sciences and Engineering Research ·February 2024
2. https://www.researchgate.net/publication/378008818
3. Yoon, Y., & Swales, G. (1991, January). Predicting stock price performance
4. Dhaka Stock Market Historical Data. Retrieved from https://www.dsebd.org
5. https://github.com
6. https://scholarworks.lib.csusb.edu/jitim
7. Cabitza, F., Locoro, A., & Banfi, G. (2018). Machine learning in orthopedics: a literature review. Frontiers in Bioengineering and Biotechnology, 6, 75.
8. Srinath Ravikumar, Prasad Saraf, "Prediction of Stock Prices using Machine Learning(Regression, Classification)Algorithms ", International Conference for Emerging Technology(INCET), 2020.
9. S. Selvin, R. Vinayakumar, E. A. Gopalakrishnan, et al., "Stock price prediction using
10. LSTM, RNN and CNN-sliding window model", 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 13-16, 2017.
11. O. Blaskowitz and H. Herwartz, "On economic evaluation of directional forecasts", International journal of forecasting, 27(4): 1058-1065, 2011.
12. I. A. Moosa and J. J. Vaz, "Direction accuracy, forecasting error and the profitability of currency trading: Simulation-based evidence-accuratezza direzionale, errore previsionale e convenienza del currency trading: evidenze dalle simulazioni", Economia Internazionale / International Economics, 67(3): 413-423, 2014.
13. H. White, "Economic prediction using neural networks: the case of IBM daily stock returns", Proceedings of the Second IEEE Annual Conference on Neural Networks, 2: 451458, 1988.
14. Yu, L., Chen, H., Wang, S., & Lai, K. K. (2008). Evolving least squares support vector machines for stock market trend mining. IEEE Transactions on evolutionary computation, 13(1), 87-102

15. Mehta, Yash, Atharva Malhar, and Radha Shankarmani. "Stock Price Prediction using Machine Learning and Sentiment Analysis." 2021 2nd International Conference for Emerging Technology (INCET). IEEE, 2021.

16. Bhattacharjee, Indronil, and Pryonti Bhattacharja. "Stock Price Prediction: A Comparative Study between Traditional Statistical Approach and Machine Learning Approach." 2019 4th International Conference on Electrical Information and Communication Technology (EICT). IEEE, 2019.