ENVISION AND PROGNOSIS FOR COMMODITIES

A Report Submitted
In Partial Fulfilment of the Requirements
for the Degree of

BACHELOR OF TECHNOLOGY

in
Computer Science and Engineering

By

Shatakshi Singh (2001640100236)

Sneha Gupta(2001640100259)

Mohd Saif (2001640100162)

Ayush Rathore (2001640100086)

Under the Supervision of

Ms. Shashi Mishra

(Assistant Professor)

Pranveer Singh Institute of Technology Kanpur

Dr. APJ ABDUL KALAM TECHNICAL UNIVERSITY

LUCKNOW

May 2024

DECLARATION

We hereby declare that the work presented in this report entitled **"ENVISION AND PROGNOSIS FOR COMMODITIES"**, was carried out by us. We have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute. We have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution. We have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

We affirm that no portion of my work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, we shall be fully responsible and answerable.

Name: Shatakshi Singh
Roll no: 2001640100236
Signature:

Name: Sneha Gupta
Roll no: 2001640100259
Signature:

Name: Mohd Saif
Roll no: 2001640100086
Signature:

Name: Ayush Rathore
Roll no: 2001640100162
Signature:

CERTIFICATE

This is to certify that Report entitled **"ENVISION AND PROGNOSIS FOR COMMODITIES"** which is submitted by Shatakshi Singh, Sneha Gupta, Mohd Saif and Ayush Rathore in partial fulfillment of the requirement for the award of degree B. Tech. in Department of Computer Science and Engineering of Pranveer Singh Institute of Technology, affiliated to Dr. A.P.J. Abdul Kalam Technical University, Lucknow is a record of the candidates own work carried out by them under my supervision. The project embodies result of original work and studies carried out by the students themselves and the contents of the project do not form the basis for the award of any other degree to the candidate or to anybody else.

Signature:

Dr. Vishal Nagar
Dean-CSE
PSIT, Kanpur

Signature:

Ms. Shashi Mishra
CSE Department,
PSIT, Kanpur

ABSTRACT

Because the securities exchange is inherently unpredictable, predicting stock prices continues to be one of the biggest guesses in the financial industry. Determining stock costs precisely is a major test because of its dynamic nature. This difficulty is compounded by the fact that traditional approaches often struggle to identify the complex examples and patterns seen in financial exchange data, which is by its very nature time series data.

The Calculated Relapse Model, Backing Vector Machines (SVM), Autoregressive Contingent Heteroskedasticity (Curve) model, Repetitive Brain Organizations (RNN), Convolutional Brain Organizations (CNN), Backpropagation, Guileless Bayes, and Autoregressive Coordinated Moving Normal (ARIMA) model are just a few of the systems that have been studied in the quest to predict stock costs. These methodologies have been applied to stock cost expectation with varying degrees of success, and each one has unique advantages.

Nevertheless, among these approaches, Long Short-Term Memory (LSTM) repeating neural networks have emerged as a particularly promising computation for handling time series problems, such as stock cost expectation. LSTM networks are a type of repeated neural network engineering that is specifically designed to capture long-term conditions and instances within subsequent data, which makes them suitable for analyzing financial exchange data that is characterized by ephemeral correlations and patterns.

This study's main objective is to accurately predict stock costs and identify current market patterns using LSTM recurrent brain organizations. We want to achieve a higher level of expectation accuracy compared to traditional methods by addressing the power of deep learning and the memory capacity of LSTM organizations.

In order to evaluate the suitability of LSTM networks for stock cost prediction, we designed experiments with vetted securities exchange data. Our results show that LSTM-based models achieve an expectation exactness of more than 93%, consistently outperforming other tactics.

This high level of accuracy suggests that LSTM companies are able to provide important insights on financial exchange components and support investors in making well-informed decisions. Additionally, the flexibility of LSTM networks allows for the integration of other components and data sources, such as expert advisors, sentiment analysis of the market, and macroeconomic variables, all of which can enhance the precision and power of forecast.

Overall, our analysis demonstrates the suitability of LSTM repeating neural networks for accurately predicting stock prices, enabling investors and financial institutions to better

investigate the nuances of the securities market with greater assurance. Through the application of deep learning and sophisticated neural network models, we can unlock new avenues for the analysis and comprehension of financial business domains, ultimately leading to the development of risk management strategies and more sophisticated venture tactics.

AKNOWLEDGEMENT

It gives us a great sense of pleasure to present the project report undertaken during **B.Tech. Final Year**. We owe special debt of gratitude to **Ms. Shashi Mishra**, Assistant Professor, Department of **Computer Science and Engineering**, PSIT, Kanpur, for his constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavours have seen light of the day. We also take the opportunity to acknowledge the contribution of **Mr. Ashutosh Pandey, Head of Department of Computer Science and Engineering, PSIT, Kanpur**, for his full support and assistance during the development of the project.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind assistance and cooperation during the development of our project. Last but not the least. we acknowledge our family & friends for their contribution in the completion of the project.

Name: Shatakshi Singh
Rollno:2001640100236
Signature:

Name: Sneha Gupta
Roll No: 2001640100259
Signature:

Name: Mohd Saif
Roll No: 2001640100162
Signature:

Name: Ayush Rathore
Roll no: 2001640100086
Signature:

TABLE OF CONTENTS

LIST OF TABLES

LIST OF EQUATIONS

LIST OF FIGURES

LIST OF ABBEREVIATIONS

Sr. No.	Abbreviation	Full Form
1.	ML	Machine Learning
2.	AI	Artificial Intelligence
3.	SVM	Support Vector Machine
4.	RNN	Recurrent Neural Network
5.	ANN	Artificial Neural Network
6.	CNN	Convolutional Neural Network
7.	CRNN	Convolutional Recurrent Neural Network
8.	LSTM	Long Short-Term Memory
9.	ARIMA	Autoregressive Coordinated Moving Normal
10.	ROI	Return On Investment
11.	GDPR	General Data Protection Regulation
12.	CCPA	Central Consumer Protection Authority
13.	IEEE	Institute Of Electrical and Electronics Engineers
14.	FEX	Foreign Exchange
15.	SLP	Single-Layer Perceptron
16.	MLP	Multi-Layer Perceptron
17.	RBF	Radial Basis Function
18.	EMA	Exponential Moving Average
19.	MLR	Multiple Linear Regression
20.	MAE	Mean Absolute Error
21.	MSE	Mean Squared Error
22.	RMSE	Root Mean Squared Error
23.	GPU	Graphics Processing Unit
24.	CPU	Control Processing Unit
25.	API	Application Programming Interface
26.	NSE	National Stock Exchange
27.	ATR	Average True Range
28.	RSI	Relative Strength Index
29.	MACD	Moving Average Convergence Divergence
30.	SMA	Simple Moving Averages
31.	EMA	Exponential Moving Average
32.	VPT	Volume Price Trend
33.	NLP	Natural Language Processing
34.	SVR	Support Vector Regression
35.	BPTT	Backpropagation Through Time
36.	LRP	Layer-Wise Relevance Propagation

CHAPTER- 1

INTRODUCTION

1.1 OVERVIEW

Due to the high profit of the stock market, it is one of the most popular investments. People investigated for methods and tools that would increase their gains while minimizing the risk, as the level of trading and investing grew. Two stock exchanges namely- the National Stock Exchange (NSE) and the Bombay Stock Exchange (BSE), which are the most of the trading in Indian Stock Market takes place. Sensex and Nifty are the two prominent Indian Market Indexes. Since the prices in the stock market are dynamic, the stock market prediction is complicated.

1.2 CHALLENGES

- **Market Volatility**: Stock markets are inherently volatile, with prices subject to rapid and sometimes unpredictable fluctuations. Factors such as changes in economic indicators, geopolitical events, and investor sentiment can all contribute to market volatility. Implementing prediction models that can accurately capture and adapt to these fluctuations poses a significant challenge, as models must be robust enough to handle sudden shifts in market conditions without sacrificing predictive accuracy.

- **Complex Relationships**: The relationship between various factors influencing stock prices is often intricate and nonlinear. Company-specific factors such as financial performance and industry trends, as well as broader market dynamics and macroeconomic indicators, all interact in complex ways to determine stock price movements. Implementing prediction models that can effectively capture and model these relationships requires sophisticated analytical techniques and data-driven approaches.

- **Data Quality and Availability**: The quality and availability of historical stock data are paramount for building reliable prediction models. However, stock market data can often be noisy, incomplete, or subject to errors. Ensuring data quality and addressing issues such as missing or inaccurate data points, data inconsistencies, and data biases are critical challenges in model implementation.Additionally, accessing and obtaining relevant data sources, especially for niche or specialized markets, can also pose challenges.

- **Avoiding Overfitting:** Overfitting occurs when a prediction model captures noise or random fluctuations in the training data, leading to poor generalization performance on unseen data. Avoiding overfitting is a significant challenge in model implementation, as models must strike a balance between capturing meaningful patterns in the data and avoiding spurious correlations. Techniques such as regularization, cross-validation, and ensemble methods are commonly employed to mitigate the risk of overfitting.

- **Interpretability:** While advanced machine learning techniques may offer high predictive accuracy, they often lack interpretability, making it challenging for users to understand the rationale behind model predictions. Interpretable models are essential for gaining insights into the factors driving stock price movements and building trust among investors and stakeholders. Balancing predictive accuracy with model interpretability is a key challenge in model implementation.

- **Adaptability to Changing Conditions:** The stock market is dynamic, with new information and market dynamics constantly emerging. Prediction models must be adaptable to changes in market conditions and able to incorporate new data in real-time to maintain relevance and accuracy. Developing models that can quickly adapt to changing market conditions, such as shifts in investor sentiment or unexpected events, is a significant challenge in model implementation.

- **Compliance and Ethics:** Implementing stock price prediction models requires adherence to regulatory standards and ethical guidelines. Models should not engage in unethical trading practices or exploit market inefficiencies, and they must comply with relevant financial regulations. Ensuring that prediction models are transparent, fair, and accountable is essential for maintaining trust and integrity in the financial markets.

  Addressing these challenges requires a multidisciplinary approach, combining expertise in finance, statistics, machine learning, and domain-specific knowledge of the stock market. By overcoming these challenges, stakeholders can harness the power of prediction models to make more informed investment decisions and navigate the complexities of the financial markets effectively.

1.3 MOTIVATION:

Stock price prediction using machine learning (ML) is motivated by several factors:

- **Profit Potential:** One of the primary motivations behind stock price prediction is the potential for profit. Investors, traders, and financial institutions aim to forecast

future price movements to make informed decisions about buying, selling, or holding stocks. ML models offer a data-driven approach to analyze historical stock data and identify patterns that may indicate potential future price movements.

- **Risk Management:** Predicting stock prices can also help in managing investment risk. By understanding potential price movements, investors can adjust their portfolios to mitigate risks or take advantage of opportunities. ML models can provide insights into market volatility, correlations between different assets, and potential risks associated with specific investments.

- **Information Synthesis:** Financial markets generate vast amounts of data from various sources, including company financial reports, market news, economic indicators, and social media sentiment. ML algorithms excel at processing and analyzing large datasets, allowing investors to synthesize diverse information sources and identify relevant signals that may impact stock prices.

- **Market Efficiency:** Efficient markets theory suggests that asset prices reflect all available information, making it difficult for investors to consistently outperform the market. However, proponents of ML-based stock prediction argue that algorithms can uncover patterns and signals that may not be immediately apparent to human analysts, providing opportunities to gain a competitive edge in the market.

- **Algorithmic Trading:** The rise of algorithmic trading has further triggered interest in stock price prediction using ML. Automated trading systems can execute trades based on predefined rules or predictive signals generated by ML models. These systems can operate at high speeds and make split-second decisions, potentially capitalizing on fleeting market opportunities that human traders may miss.

- **Research and Innovation:** Research in ML-based stock prediction contributes to advancements in both finance and machine learning fields. Developing accurate prediction models requires innovation in data preprocessing, feature engineering, model selection, and evaluation techniques. Furthermore, successful applications of ML in finance can inspire further interdisciplinary collaboration and drive the development of new methodologies and algorithms.

  Overall, stock price prediction using ML is motivated by the desire to make informed investment decisions, manage risk, leverage information efficiently, potentially outperform the market, automate trading processes, and contribute to research and innovation in finance and machine learning domains.

1.4 OBJECTIVE:

The objective of stock price prediction using machine learning is to forecast future stock prices based on historical data and various input features. This aims to create an accurate model to predict price movements, which can inform trading strategies, manage risk, and potentially maximize profits. Accurate predictions are crucial as they minimize errors measured by metrics like Mean Square Error (MSE), Mean Absolute Error (MAE), or Root Mean Square Error (RMSE). Timeliness is also essential, as predictions must be made in real-time or near-real-time to be useful for trading decisions. Additionally, the model must generalize well to unseen data, meaning it should perform well not only on historical data but also on future data, ensuring its reliability.

1.5 PROBLEM STATEMENT:

A stock market prediction aims to forecast the future value of a company's stock or other financial investments traded on the stock exchange. Successfully predicting the forthcoming price of a stock can leads to significant profit, enabling investors to make informed decisions and allocate resources effectively. By leveraging historical data, market trends, and predictive modelling techniques, stock market prediction endeavours to identify patterns and trends that may influence future price movements. This proactive approach empowers investors to strategically allocate their capital, manage risk, and capitalize on emerging opportunities in the financial markets. Additionally, accurate stock market prediction can enhance market efficiency by providing valuable insights into investor sentiment, market dynamics, and macroeconomic trends. Ultimately, the goal of stock market prediction is to optimize investment returns, mitigate downside risk, and contribute to the overall success and stability of the financial ecosystem.

1.6 FEASIBILITY STUDY

- **Executive Summary:**
  The feasibility study assesses the viability and potential of developing a stock market prediction system. This system aims to forecast future stock prices using historical data and predictive modelling techniques. The study evaluates the technical, financial, and operational aspects of the project to determine its feasibility and potential benefits.

- **Objective:**
  The objective of the feasibility study is to determine whether investing resources into the development of a stock market prediction system is justified. This involves evaluating the technical capabilities, market demand, financial implications, and operational requirements of the project.

1.6.1 Technical Feasibility:

- **Data Availability:** Assessing the availability and quality of historical stock market data required for training prediction models. Identify relevant data sources, data preprocessing requirements, and potential challenges.

- **Model Development:** Evaluating the feasibility of developing accurate prediction models using machine learning algorithms. Assess the complexity of model development, required expertise, and availability of suitable tools and technologies.

- **Infrastructure Requirements:** Determining the infrastructure needed to support model training, testing, and deployment. Evaluate hardware, software, and computational resources required for efficient model development and execution.

1.6.2 Market Feasibility:

- **Demand Analysis:** Analyze the demand for stock market prediction services among investors, traders, and financial institutions. Identify potential target markets, customer segments, and their specific needs and preferences.

- **Competitive Landscape:** Assess the competitive landscape and existing solutions in the stock market prediction market. Identify key competitors, their offerings, strengths, and weaknesses.

- **Market Trends:** Evaluate current market trends, regulatory developments, and emerging technologies that may impact the demand for stock market prediction services.

1.6.3 Financial Feasibility:

- **Cost Analysis:** Estimate the initial investment required to develop the stock market prediction system. Consider costs related to data acquisition, model development, infrastructure setup, and personnel.

- **Revenue Projection:** Forecast potential revenue streams from offering stock market prediction services. Consider pricing models, subscription fees, and potential market penetration.

- **Return on Investment (ROI):** Evaluate the expected ROI based on the projected costs and revenues. Assess the payback period and profitability of the project over time.

1.6.4 Operational Feasibility:

- **Resource Availability:** Assess the availability of skilled personnel, expertise, and other resources required for project execution. Identify any potential gaps or constraints.

- **Project Timeline:** Develop a realistic timeline for project implementation, including key milestones and deliverables. Consider factors such as data collection, model development, testing, and deployment.

- **Risk Analysis:** Identify potential risks and challenges associated with project implementation, such as data privacy concerns, regulatory compliance, and technical limitations. Develop mitigation strategies to address these risks.

1.6.5 Legal and Administrative Considerations:

- **Consistence Requirements:** Survey the lawful and administrative structures overseeing monetary business sectors and information utilization. Guarantee consistence with applicable regulations, guidelines, and industry norms, including information assurance guidelines (e.g., GDPR, CCPA) and protections guidelines.

- **Information Protection and Security:** Assess measures to guarantee the protection and security of delicate monetary information utilized in the forecast models. Execute strong information encryption, access controls, and information anonymization strategies to protect against unapproved access and information breaks.

1.6.6 Scalability and Flexibility:

- **Versatility Assessment:** Assess the adaptability of the forecast framework to deal with expanding volumes of information and client demands. Think about future development projections, framework engineering, and versatility systems, for example, cloud-based arrangements and dispersed registering.

- **Adaptability:** Survey the framework's adaptability to adjust to advancing economic situations, new information sources, and changing client

prerequisites. Execute secluded plan standards and lithe advancement procedures to work with future improvements and cycles.

1.6.7 Operational Productivity and Integration:

- **Coordination with Existing Systems:** Assess the similarity of the forecast framework with existing foundation, programming, and business processes. Distinguish joining focuses and likely conditions to smooth out execution and limit disturbance.

- **Functional Efficiency:** Evaluate the functional productivity acquires accomplished through the execution of the expectation framework. Distinguish potential open doors for process improvement, robotization, and cost reserve funds in information examination, direction, and exchanging exercises.

1.6.8 Ethical and Social Implications:

- **Moral Considerations:** Consider the moral ramifications of involving prescient calculations in monetary direction. Guarantee reasonableness, straightforwardness, and responsibility in model turn of events and sending to relieve predispositions and advance dependable artificial intelligence rehearses.

- **Social Effect Assessment:** Assess the likely friendly effect of the forecast framework on market solidness, financial backer certainty, and abundance dispersion. Consider measures to address differences, advance monetary education, and alleviate foundational takes a chance in the monetary biological system.

- By tending to these extra places, the practicality study gives a far-reaching evaluation of the securities exchange forecast project, covering specialized, market, monetary, functional, lawful, administrative, and moral aspects. This all-encompassing methodology empowers partners to come to very much educated conclusions about the undertaking's practicality, dangers, and possible advantages.

CHAPTER-2

LITERATURE SURVEY

2.1 Stock Price forecasting using Data from Yahoo Finance and analysing seasonal and non-seasonal trend

In [2], to identify the relationship between different existing time series algorithms namely ARIMA and Holt Winter and the stock prices is the main objective of the proposed work, for the investments a good risk-free range of stock prices are analysed and therefore better accuracy of the model can be seen. To find distinguished results for shares in the stock market, the combination of two different time series analysis models is opted by producing a range of prices to the consumer of the stocks. Not complex in nature and estimation of values which are purely based on the past stock prices for non-seasonal or seasonal is the main advantage of these models. In this experiment, some limitations are, the work that never takes into consideration and other circumstances like news about any new market strategy or media release relevant to any company which may get affected by the prices of stocks.

2.2 Stock Market Prediction Using Machine Learning

In this paper studies [3], the use of Regression and LSTM based Machine learning to forecast stock prices. Factors measured are open, close, low, high and volume. This paper was an attempt to determine the future prices of the stocks of a company with improved accuracy and reliability using machine learning techniques. LSTM algorithm resulted in a positive outcome with more accuracy in predicting stock prices.

2.3 Multi-Category Events Driven Stock Price Trends Prediction

In this paper [4], multi-category news events are used as features to develop stock price trend prediction, model. The multi-category events are based on already defined feature word dictionary. And we have employed both neural networks and SVM models to analyse the relationship between stock price movements and specific multi-category news. Experimental results showed that the predefined multi-category news events are more improved than the baseline bag-of-words feature to predict stock price trend. As compared to long term prediction, short term prediction is better based on this study.

2.4 Share Price Prediction using Machine Learning Technique:

This paper [5] is mostly based on the approach of predicting the share price using Long-Short-Term-Memory (LSTM) and Recurrent Neural Networks (RNN) to forecast the stock value on NSE data using various factors such as current market price, price earnings ratio, base value and other anonymous events. The efficiency of the model is analysed by comparing the true data and the predicted data using an RNN graph. Machine learning to predict stock price as see the model is able to predict the stock price very close to the actual price where this model captures the detailed feature and uses different strategies to make a prediction. The model train for all the NSE data from the internet and recognize the input and group them and provide input according to the user configuration this RNN based architecture proved very efficient in forecasting the stock price by changing the configuration accordingly which also use backpropagation mechanism while gathering and grouping data to avoid mixing of data.

2.5 Stock Market Prediction Using Machine Learning Techniques:

The prominent aim of this study [6] is to forecast the market performance of the Karachi Stock Exchange (KSE) on day closing using machine learning algorithms. A variety of attributes as an input and forecasts market as Positive & Negative is predicted by using the predictions model. The features employed in the model are contains Oil rates, Gold & Silver rates, Interest rate, Foreign Exchange (FEX) rate, NEWS and social media feed. The machine learning algorithms including Single Layer Perceptron (SLP), Multi-Layer Perceptron (MLP), Radial Basis Function (RBF) and Support Vector Machine (SVM) are compared. The algorithm MLP that is multi-layer perceptron performed best as compared to different methods. The foremost helpful feature in predicting the market was the oil rate attribute. The end results of this research confirm that machine learning techniques have the ability to predict the stock market performance. The Multi-Layer Perceptron algorithm of machine learning predicted 70% correct market performance.

2.6 Forecasting stock price in two ways based on LSTM neural network:

In paper [7], the LSTM neural network is used to predict Apple stocks by consuming single feature input variables and multi-feature input variables to verify the forecast effect of the model on stock time series. The experimental results show that the model has a high accuracy of 0.033 for the multivariate input and is accurate, that is in line with the actual demand. For the univariate feature input, the predicted squared absolute error is 0.155, which is inferior to the multi-feature variable input.

2.7 Stock price trend prediction using CRNN and LSTM structure

The entire financial market majorly runs by the stock market and one of the most attractive research issues is predicting stock price volatility [8]. The information of historical stocks for assuming the future stock price as well deep learning method is applied to find approximate trend value of stock prices which are mentioned in this paper. This paper not only stores the data of historical stock with the time scale but also estimates prices of the future stock by a designed neural network, this is due to the fact that the trend of stocks is usually connected to the previous information of stock price. In this paper, the design of the neural network proposed then with the memory performance the convolutional recurrent neural network (CRNN) and for improving the long-term dependency of traditional RNN the Long Short-term memory (LSTM) are the major components. Also to enhance the accuracy as well as stability of prediction of the RNN LSTM architecture is put. This paper accumulates a total of ten stock historic data to test and accomplish an average error rate of 3.449 RMSE [3].

2.8 Applying Long Short Term Memory Neural Networks for predicting stock closing Price

To assess the scheme that merges RNNs with informative input variables which can give an improved and effective method to forecast the next-day market is the main objective of this paper [9]. The stock prediction model analyses using long-short memory (LSTM) and stock basic trading data. On Standard & Poor's (S&P500) and NASDAQ, the case study relies. The stock closing price is more precisely predicted using their forecasting system for the next day, which outperforms the comparison models. This is the main discovery of the case study. Five various models namely – moving average (MA), exponential moving average (EMA), support vector machine (SVM) and LSTM are tested by them to demonstrate the utility of the system. The closing value of the next day is the predicting target.

2.9 Developing a Prediction Model for Stock Analysis

A relative study of the three algorithms namely - Multiple Linear Regression (MLR), Support Vector Machine (SVM) and Artificial Neural Network (ANN) is the main aim of this study [10]. To predict the coming day market price, the prediction will be determined by monthly prediction and daily prediction. Sentiment analysis with the best prediction algorithm forecast the stock price. The less-developed algorithm is the Multiple Linear Regression algorithm which calculates the correlation between volume and the stock price. The result of the study shows that deep learning algorithms are more developed than MLR algorithms and SVM algorithm.

2.10 Stock price prediction based on information entropy and artificial neural network

One of the most important components of the financial system is the stock market [11]. For supporting the activity and evolvement, money is directed by the investors of the associated from. Along with information theory and Artificial Neural Network (ANN) the combination of machine learning framework is formed. Information entropy for non-linear causality and stock relevance also to facilitate ANN time series modelling are creatively used by this method. The feasibility of this machine learning framework is analysed with Amazon, Apple, Google and Facebook prices. A time series analysis method based on information theory as well as LSTM to model the stock price dynamics are outlined in this paper. The transfer entropy between relevant variables to help LSTM time series prediction is merged in this modelling infrastructure, thus the accuracy of the assumption outcome is broadly granted. Modelled and real stock price is highly correlated while differ slightly in terms of Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) which are investigated by the outcomes.

Summary Of Literature Survey:

Here, I have reviewed various approaches for Stock price prediction. All approaches have their own advantages and disadvantages. CNN & LSTM is a most popular algorithm to prediction the stock price but there are some challenges in this method like use to need a lot of training data, High computational cost, without GPU data quite slow to train, depend on any previous information for prediction. A hybrid approach can be used to overcome these issues. While machine learning is able to provide highly accurate prediction result using standards tools and also outperforms all standard prediction methods.

CHAPTER-3

METHODOLOGY

3.1 TECHSTACKS

These are the following tech-stacks used in building the model to make it efficient and enhance optimization.

3.1.1 PYTHON:

The language of select for this project was Python. This was a straightforward call for many reasons.

3.1.1.1. Python as a language has a vast community behind it. Any problems which may be faced is simply resolved with visit to Stack Overflow. Python is the foremost standard language on the positioning that makes it is very straight answer to any question [19].

3.1.1.2. Python [19] is an abundance of powerful tools ready for scientific computing Packages. The packages like NumPy, Pandas and SciPy area unit freely available and well documented. These Packages will intensely scale back, and variation the code necessary to write a given program. This makes repetition fast.

3.1.1.3. Python is a language as [19] forgiving and permits for the program that appear as if pseudo code. This can be helpful once pseudo code give in tutorial papers should be required and verified. Using python this step is sometimes fairly trivial.

3.1.1.4. However, Python is [19] not without its errors. The python is dynamically written language and packages are area unit infamous for Duck writing. This may be frustrating once a package technique returns one thing that, for instance, looks like an array instead of being an actual array. Plus, the standard Python documentation did not clearly state the return type of a method, this can't lead without a lot of trials and error testing otherwise happen in a powerfully written language. This is a problem that produces learning to use a replacement Python package or library more difficult than it otherwise may be.

3.1.2 NUMPY

NumPy is python package which provide scientific and higher-level mathematical abstractions wrapped in python. It is [20] the core library for scientific computing. It contains a provide tools for integrating C, strong n-dimensional array object, C++ etc. It is also useful in random number capability, linear algebra etc.

NumPy's array type augments the Python language with an efficient data structure used for numerical work, e.g., manipulating matrices. NumPy additionally provides basic numerical routines, like tools for locating Eigenvector.

3.1.3 SCIKIT LEARN

Scikit-learn [21] could be a free machine learning library for Python. It features numerous classification, clustering and regression algorithms like random forests, k-neighbours, support vector machine, and it furthermore supports Python scientific and numerical libraries like SciPy and NumPy.

In Python Scikit-learn is specifically written, with the core algorithms written in Cython to get the performance. Support vector machines are enforced by a Cython wrapper around LIBSVM .i.e., linear support vector machines and logistic regression by a similar wrapper around LIBLINEAR.

3.1.4 TENSORFLOW:

In the TensorFlow has an open-source software library for numerical computation using data flow graphs. Inside the graph nodes represent mathematical formulae, the edges of graph represent the multidimensional knowledge arrays (tensors) communicated between them. The versatile architecture permits to deploy the computation to at least one or many GPUs or CPUs in a desktop, mobile device, servers with a single API. TensorFlow was firstly developing by engineers and researchers acting on the Google Brain Team at intervals Google's Machine Intelligence analysis organization for the needs of conducting deep neural networks research and machine learning, but, the system is generally enough to be appropriate in a wide range of alternate domains as well [23].

Google Brain's second-generation system is TensorFlow. Whereas the reference implementation runs on single devices, TensorFlow can run on multiple GPUs and CPUs. TensorFlow is offered on Windows, macOS, 64-bit Linux and mobile computing platforms together with iOS and Android.

3.1.5 KERAS

Keras is a high-level neural networks API, it is written in Python and also capable of running on top of the Theano, CNTK, or. TensorFlow. It was developed with attention on enabling quick experimentation. having the ability to travel from plan to result with the smallest amount doable delay is key to doing great research.
Keras permits for straightforward and quick prototyping (through user-friendliness, modularity, and extensibility). Supports each recurrent networks and convolutional networks, also as combinations of the two [23].
It runs seamlessly on GPU and CPU. The library contains numerous implementations of generally used neural network building blocks like optimizers, activation functions, layers, objectives and a number of tools to create operating with text and image data easier. The code is hosted on GitHub, and community support forums embody the GitHub issues page, a Gitter channel and a Slack channel.

3.1.6 COMPILER OPTION:

Anaconda is free premium open-source distribution of the R and Python programming languages for scientific computing, predictive analytics, and large-scale process that aim is to modify package managing and deployment. Package versions unit managed by the package management system Conda [19].

3.1.7 JUPYTER NOTEBOOK:

The Jupyter Notebook is an open-source web application that enables to making and sharing documents that contain visualizations, narrative text, live code and equations. Uses include: Data, data visualization, data transformation, statistical modelling, machine learning, numerical simulation, data cleaning and much more [24].

3.2 DATA COLLECTION AND PREPROCESSING:

Pandas: A powerful library for data manipulation and analysis in Python. Pandas provides data structures like Data-Frame and tools for cleaning, filtering, and transforming data efficiently.

Yahoo Finance API, Alpha Vantage, or Google Finance API: APIs that allow fetching historical stock price data from respective sources. These APIs provide convenient access to financial data for analysis and model training.

Dataset: The dataset consists of the stock historical data from the National stock exchange (NSE) and captures the daily information of each stock from the National Stock Exchange. It collects different sectors of stock data, including Banking, Pharma, Software, Personal and skincare and it includes the opening price, the highest price, the lowest price, the closing price, the adjusted closing price and the volume of stock [18].

Here are the few datasets comprised of stock prices, trading volumes and economic indicators.

SECTOR	STOCK
Software	Netflix, Microsoft
Pharma	Sun Pharma
Banking and Finance	CITI and ICICI Bank

Table: 3.1

3.3 LIMITATIONS AND CONDITIONS

While regression models and LSTM networks are powerful tools for stock price prediction, they also have limitations and conditions under which they perform optimally. Here are some considerations:

3.3.1 Regression Models:

3.3.1.1 Assumption of Linearity: Linear regression assumes a linear relationship between input features and the target variable (stock price). If the relationship is highly nonlinear, linear regression may not capture it effectively.

3.3.1.2 Overfitting: Without proper regularization techniques, regression models can overfit to the training data, especially when dealing with a large number of features or multicollinearity among the features.

3.3.1.3 Limited Representation: Regression models may not be able to capture complex temporal dependencies or patterns in time series data. They are also less capable of handling irregularly spaced time series data.

3.3.1.4 Sensitive to Outliers: Linear regression models can be sensitive to outliers in the data, which may skew the learned parameters and affect the model's predictive performance.

3.3.2 LSTM Models:

3.3.2.1. Data Requirements: LSTM models require a significant amount of historical time series data to learn meaningful patterns and dependencies. Insufficient data can lead to poor generalization and overfitting.

3.3.2.2. Complexity and Interpretability: LSTM models are more complex than regression models and may be harder to interpret. Understanding the internal workings of LSTM networks and diagnosing issues during training can be challenging.

3.3.2.3. Hyperparameter Sensitivity: LSTM models have several hyperparameters that need to be tuned, such as the number of layers, number of units per layer, learning rate, and batch size. Finding the optimal set of hyperparameters can require extensive experimentation.

3.3.2.4. Computational Resources: Training LSTM models can be computationally expensive, especially for large datasets and complex architectures. GPU acceleration is often necessary to train LSTM networks efficiently.

3.3.2.5. Risk of Overfitting: Like other deep learning models, LSTMs are susceptible to overfitting, especially when dealing with noisy or limited data. Regularization techniques such as dropout and early stopping can help mitigate this risk.

3.3.2.6. Interpretability: While LSTMs can capture complex temporal dependencies, interpreting the learned representations and understanding why certain predictions are made can be challenging compared to traditional regression models.

3.3.3 General Considerations:

3.3.3.1. Data Quality: Both regression models and LSTM networks are sensitive to the quality of the input data. It's essential to preprocess the data carefully, handle missing values, outliers, and ensure that the data is representative of the underlying process being modelled.

3.3.3.2. Feature Engineering: The choice and engineering of features can significantly impact the performance of both regression models and LSTMs. Domain knowledge and careful feature selection are critical for building effective predictive models.

3.3.3.3. Model Evaluation: Proper evaluation of model performance is crucial. Techniques such as cross-validation, out-of-sample testing, and using appropriate evaluation metrics help assess the predictive accuracy and generalization ability of the models.

In summary, while regression models and LSTM networks offer powerful tools for stock price prediction, it's essential to consider their limitations and conditions under which they perform best. Careful experimentation, data preprocessing, feature engineering, and model evaluation are key to building accurate and robust predictive models.

3.4 DATAFLOW AND TRAINING PROCESS:

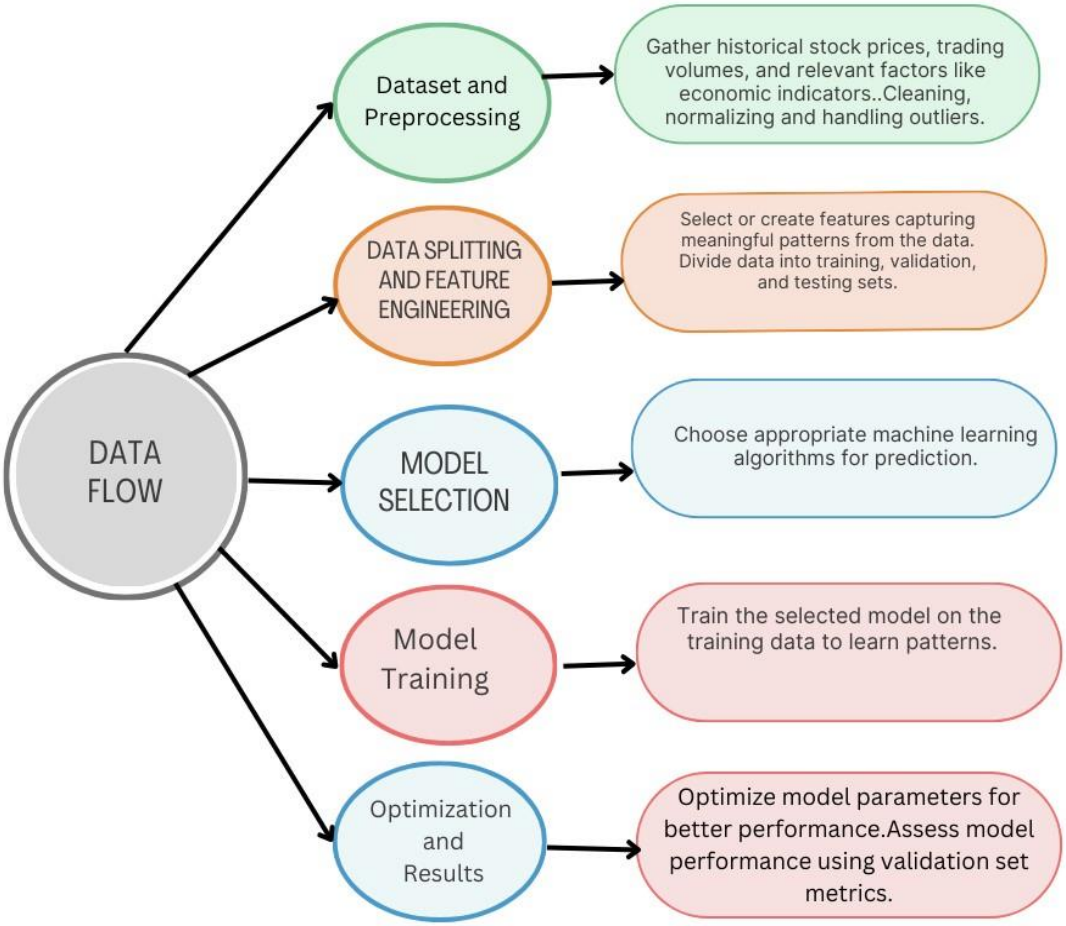


Figure: 3.1 Data Flow

The system presented here composes of five modules: -

3.4.1. Dataset And Preprocessing
3.4.2. Data Splitting and Feature Engineering
3.4.3. Model Selection
3.4.4. Model Training
3.4.5. Optimization and Result

Attribute such as: price of open, high, low, close, adjusted close price taken from huge dataset are fed as input to the models for training to pre-process the data techniques like normalization & one hot encoding in applied on dataset. After this data is divided in two sets namely training & testing which are ratio of 80:20 respectively. Then, this set are used to train a model using 3 different approaches: LSTM, CNN and Hybrid approach of LSTM+CNNS. Finally, all these modules are evaluated using Root mean square error

3.5 FEATURE EXTRACTION

let's delve deeper into feature extraction in the context of LSTM and Regression models for stock price prediction:

3.5.1. Input Data Preparation:

- Before feature extraction, raw data must be prepared for analysis. This involves cleaning the data to handle missing values, outliers, and inconsistencies. Data may also be aggregated or resampled to a consistent time frequency, such as daily, weekly, or monthly.

3.5.2. Time-Series Features:

- For LSTM models, time-series features are particularly important as they capture temporal dependencies and patterns in the data. Common time-series features include:
- **Lagged values:** Including lagged versions of the target variable or other features to capture autocorrelation and temporal dependencies.
- **Moving averages:** Calculating moving averages over different time windows to smooth out noise and capture trends.
- **Volatility measures:** Computing volatility indicators such as standard deviation, average true range (ATR), or historical volatility to quantify price fluctuations.
- **Momentum indicators:** Deriving momentum-based features such as relative strength index (RSI), moving average convergence divergence (MACD), or stochastic oscillator to capture market momentum.

3.5.3 Technical Indicators:

- Both LSTM and Regression models can benefit from incorporating technical indicators derived from price and volume data. These indicators provide insights into market dynamics and investor behaviour. Examples include:
- **Moving averages:** Simple moving averages (SMA), exponential moving averages (EMA), or weighted moving averages (WMA) to identify trends and support/resistance levels.
- **Oscillators:** Indicators such as RSI, MACD, stochastic oscillator, or Williams %R to identify overbought or oversold conditions and potential trend reversals.
- **Volume-based indicators:** Metrics like volume moving averages, on-balance volume (OBV), or volume-price trend (VPT) to assess the strength of price movements and confirm trends.

3.5.4. Fundamental Features:

- For Regression models, fundamental features derived from company financials and economic indicators can provide valuable information about the underlying value and performance of the stock. Examples include:
- **Financial ratios:** Calculating ratios such as price-to-earnings (P/E), price-to-book (P/B), or debt-to-equity (D/E) ratios to assess valuation and financial health.
- **Growth metrics:** Deriving growth rates for revenue, earnings, or book value to evaluate the company's growth prospects.
- **Economic indicators:** Incorporating macroeconomic indicators such as GDP growth, inflation rates, or interest rates to assess broader market trends and economic conditions.

3.5.5. Sentiment Analysis:

- Both LSTM and Regression models can benefit from sentiment analysis of news articles, social media posts, and other textual data sources. Sentiment-based features can capture market sentiment and investor emotions, which can influence stock prices. Techniques include:
- **Textual analysis:** Using natural language processing (NLP) techniques to analyze sentiment from textual data sources and derive sentiment scores or sentiment-based features.
- **Sentiment aggregation:** Aggregating sentiment scores from multiple sources (e.g., news articles, social media posts) to create composite sentiment indicators for each stock.

3.5.6. Feature Engineering:

- Feature engineering involves creating new features or transforming existing features to improve model performance. Techniques include:
- **Polynomial features:** Generating polynomial features or interaction terms to capture nonlinear relationships between variables.
- **Feature scaling:** Scaling features to a common range (e.g., [0, 1] or [-1, 1]) to ensure numerical stability and improve convergence.
- **Feature selection:** Selecting a subset of the most relevant features using techniques such as correlation analysis, feature importance scores, or model-based selection methods.

  By incorporating a diverse set of features derived from various data sources and employing advanced feature engineering techniques, LSTM and

Regression models can effectively capture the complex relationships and patterns in stock price data, leading to more accurate and reliable predictions.

3.6. ERROR HANDLING

Error handling is crucial in any project, including stock price prediction, to ensure robustness, reliability, and resilience against unforeseen issues. Here are some details on error handling strategies for the project:

3.6.1. Data Acquisition Errors:

- Handle errors related to data acquisition from external sources such as financial APIs or data providers. This includes:
- Implementing retry mechanisms with exponential backoff to handle temporary network failures or API rate limits.
- Logging errors and notifying administrators or developers for manual intervention in case of persistent data acquisition failures.
- Implementing caching mechanisms to store previously retrieved data and minimize dependencies on external data sources.

3.6.2. Data Preprocessing Errors:

- Address errors that may occur during data preprocessing, including missing values, outliers, or data inconsistencies. Strategies include:
- Implementing robust data cleaning routines to handle missing or invalid data values, such as imputation techniques or outlier detection algorithms.
- Performing data validation checks to ensure data integrity and consistency before feeding it into the prediction models.
- Logging preprocessing errors and implementing error handling logic to gracefully handle edge cases or unexpected data conditions.

3.6.3. Model Training Errors:

- Handle errors encountered during model training, including convergence issues, numerical instability, or resource constraints. Strategies include:
- Monitoring model training progress and performance metrics to detect anomalies or deviations from expected behaviour.
- Implementing model validation checks to ensure model stability and prevent overfitting or underfitting.
- Using techniques such as early stopping, regularization, or hyperparameter tuning to improve model robustness and generalization performance.

3.6.4. Model Deployment Errors:

- Address errors that may arise during model deployment and inference, such as runtime errors, resource exhaustion, or unexpected input data. Strategies include:
- Implementing health checks and monitoring systems to continuously monitor model performance and availability.
- Implementing fallback mechanisms or failover strategies to switch to alternative models or backup systems in case of model failures.
- Implementing input validation checks to sanitize and validate input data before passing it to the prediction models, reducing the risk of runtime errors or security vulnerabilities.

3.6.5. Logging and Error Reporting:

- Implement comprehensive logging and error reporting mechanisms throughout the project to track errors, debug issues, and facilitate troubleshooting. Strategies include:
- Logging errors at various stages of the project, including data acquisition, preprocessing, model training, and inference.
- Using structured logging formats and severity levels to categorize and prioritize errors.
- Integrating with centralized logging platforms or monitoring tools to aggregate and analyze error logs, enabling proactive error detection and resolution.

3.6.6. Documentation:

- Documenting error handling procedures, including common error scenarios, mitigation strategies, and escalation paths, to ensure consistency and clarity among project stakeholders.
- Providing detailed documentation for developers, operators, and end-users on how to interpret error messages, troubleshoot issues, and escalate unresolved problems to the appropriate parties.

By implementing robust error handling mechanisms and proactive monitoring strategies, the stock price prediction project can maintain reliability and resilience, even in the face of unexpected errors or disruptions.

- Some Important Considerations:

3.6.7 MEAN SQUARE ERROR (MSE)

Mean Square Error (MSE) is a fundamental metric used to evaluate the performance of a regression model. It measures the average squared differences between the actual values and the predicted values produced by the model. Here's a more detailed explanation of MSE in the context of regression:

$$MAE = n1\sum i=1n|yi-y\hat{}i|$$

Eq: 3.1 MSE

3.6.8 ROOT MEAN SQUARE ERROR(RMSE)

Root Mean Square Error (RMSE) is a commonly used metric to measure the accuracy of a model's predictions compared to the actual values. It is particularly useful in evaluating regression models, including those used in stock price prediction, such as LSTM (Long Short-Term Memory) and regression models.

The formula for RMSE is:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Eq:3.2 RMSE

Where:

- n is the number of observations or data points.
- y_i represents the actual values or ground truth for the i-th observation.
- $\hat{y}_i$ represents the predicted values for the i-th observation.

In the context of stock price prediction using LSTM and regression:

- In LSTM-based stock price prediction, the model learns patterns and dependencies in the historical stock price data and predicts future prices. After training the model, you evaluate its performance using RMSE by comparing the predicted prices with the actual prices from a validation dataset.

- In regression-based stock price prediction, you typically use features like historical prices, trading volume, and other relevant factors to predict future prices. After training the regression model, you evaluate its performance using RMSE by comparing the predicted prices with the actual prices from a validation dataset.

- RMSE provides a measure of the typical error in the model's predictions in the same units as the predicted values (e.g., dollars for stock prices). Lower RMSE values indicate better model performance, as they indicate smaller prediction errors.

3.6.9 ABSOLUTE ERROR:

Absolute error, also known as absolute deviation or simply error, is a measure of the difference between the predicted value and the actual value without considering the direction of the difference. It provides a straightforward indication of how much the prediction deviates from the true value, regardless of whether the prediction is too high or too low.

When using LSTM (Long Short-Term Memory) or regression models for stock price prediction, you can calculate the absolute error for each prediction by taking the absolute difference between the *predicted value* (denoted as *y***^) and the *actual value* (denoted as *y*) for each data point.

Here's the formula for absolute error:

$$\text{Absolute Error} = |\hat{y} - y|$$

Eq:3.3 Absolute Error

Where:

- $\hat{y}$ represents the predicted value.
- y represents the actual value.

3.7. MODEL SELECTION

Model selection is a crucial step in building a stock price prediction model. It involves choosing the most appropriate algorithm or model architecture for your specific prediction task. Let's delve into model selection, focusing on regression and LSTM models:

3.7.1 Regression Models:

3.7.1.1. Linear Regression: This is a simple and interpretable model that assumes a linear relationship between the input features and the target variable (stock price). It's a good starting point for regression tasks, but it may not capture complex nonlinear relationships.

3.7.1.2. Ridge Regression and Lasso Regression: These are variants of linear regression that introduce regularization to prevent overfitting. Ridge regression adds a penalty term to the loss function based on the L2 norm of the coefficients, while Lasso regression adds a penalty based on the L1 norm. These models are useful when dealing with multicollinearity and high-dimensional feature spaces.

3.7.1.3. Random Forest Regression and Gradient Boosting Regression: These are ensemble learning methods that combine multiple decision trees to improve predictive performance. They are capable of capturing nonlinear relationships and interactions between features and can handle large datasets effectively.

3.7.1.4.Support Vector Regression (SVR): SVR is a regression algorithm based on support vector machines. It works by finding the hyperplane that best fits the data while maximizing the margin. SVR is effective in high-dimensional spaces and is robust to outliers.

3.7.2 Long Short-Term Memory (LSTM):

3.7.2.1. Overview: LSTM is a type of recurrent neural network (RNN) architecture designed to capture long-term dependencies in sequential data. It's well-suited for time series data like stock prices because it can remember information over long periods.

3.7.2.2. Architecture: LSTM networks consist of memory cells that maintain a cell state and various gates (input gate, forget gate, output gate) that control the flow of information into and out of the cells. This architecture allows LSTMs to learn which information to retain or discard over time, making them effective for modelling sequences with long-term dependencies.

3.7.2.3. Training: LSTMs are trained using backpropagation through time (BPTT), a variant of backpropagation suitable for sequential data. During training, the network learns to update its internal parameters (weights and biases) to minimize the difference between the predicted and actual stock prices.

3.7.2.4. Hyperparameter Tuning: Key hyperparameters to tune when training LSTM models include the number of LSTM layers, the number of units in each layer, the learning rate, the batch size, and the sequence length (time steps) used for training.

3.7.2.5. Applications of LSTM includes:

a) Language Modelling
b) Machine Translation
c) Image Captioning
d) Handwriting generation
e) Question Answering Chatbot.

While selecting between regression models and LSTM for stock price prediction, we considered the complexity of the data, the presence of nonlinear relationships, the availability of historical time series data, and computational resources. As a result, regression models were more interpretable and computationally efficient for simpler datasets, while LSTM models excel at capturing complex temporal dependencies in sequential data but require more data and computational resources for training.

CHAPTER-4

DESIGNING

4.1 Data Flow Diagram (Level0)

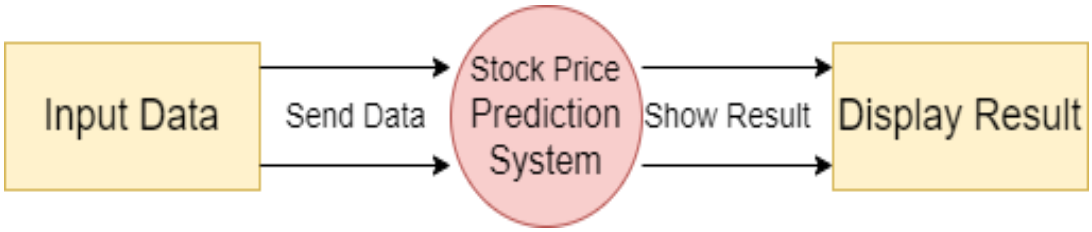


Figure:4.1

A data flow diagram (DFD0) is a graphical representation of how data flows through a system or process. It helps us understand how information moves from input sources to various processes and finally to output destinations. Let's delve into the details based on the image you provided:

4.1.1. Input Data:

- The leftmost rectangle represents the input data.
- This data could come from various sources, such as user input, sensors, or databases.

4.1.2 Stock Price Prediction System:

- The large oval in the middle represents the Stock Price Prediction System.
- It processes the input data to predict stock prices.
- Inside this system, data is analyzed, transformed, and used for prediction.

4.1.3 Send Data:

- The arrow indicates that the input data is sent to the prediction system.

- This step represents the flow of information from the input to the prediction process.

4.1.4 Show Result:

- Once the prediction is made, the system generates a result.
- The result (predicted stock price) is then *shown* or made available for further use.

4.1.5 Display Result:

- The rightmost rectangle represents the display result stage.
- Here, the predicted stock price is presented to the user or any relevant stake-holders.

4.2 Data Flow Diagram (Level1)

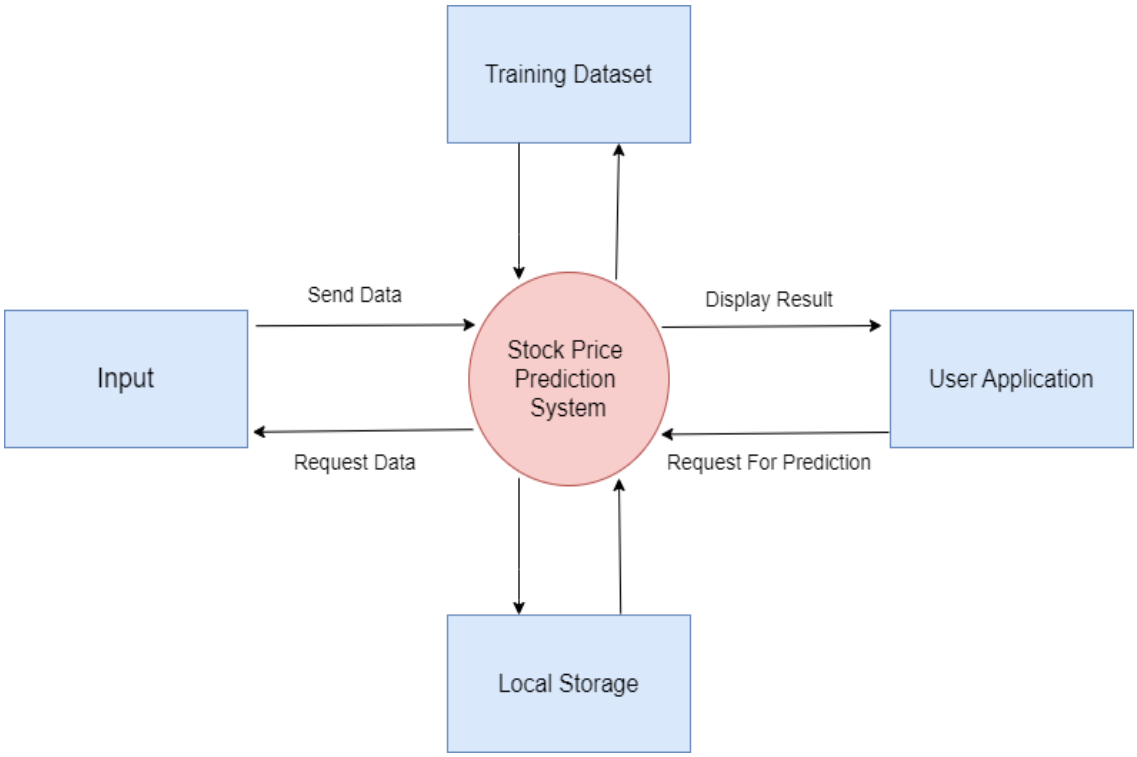


Figure:4.2

Let's break down the key points of the level 1 Data Flow Diagram (DFD) for the Stock Price Prediction System using regression and LSTM:

4.2.1 Input (Raw Data Entry):

- Represents where data enters the system.
- Could include historical stock prices, trading volumes, or other financial data.
- Sent to both the Training Dataset and Local Storage.

4.2.2 Training Dataset:

- Used for model training.
- Involves preprocessing data and training machine learning models (like regression or LSTM).
- Receives data directly from the Input component.

4.2.3 Stock Price Prediction System (Central Processing):

- Core component that processes input data.
- Makes predictions based on trained models.
- Sends prediction results to the User Application.

4.2.4 Local Storage:

- Stores input data for future use.
- Ensures data doesn't need to be fetched repeatedly.
- Can also provide requested data back to the Stock Price Prediction System.

4.2.5 User Application:

- Interface for users to interact with the system.
- Users can request stock price predictions.
- Predictions are displayed in this interface after processing by the Stock Price Prediction System.

4.3 Data Flow Diagram (Level2)

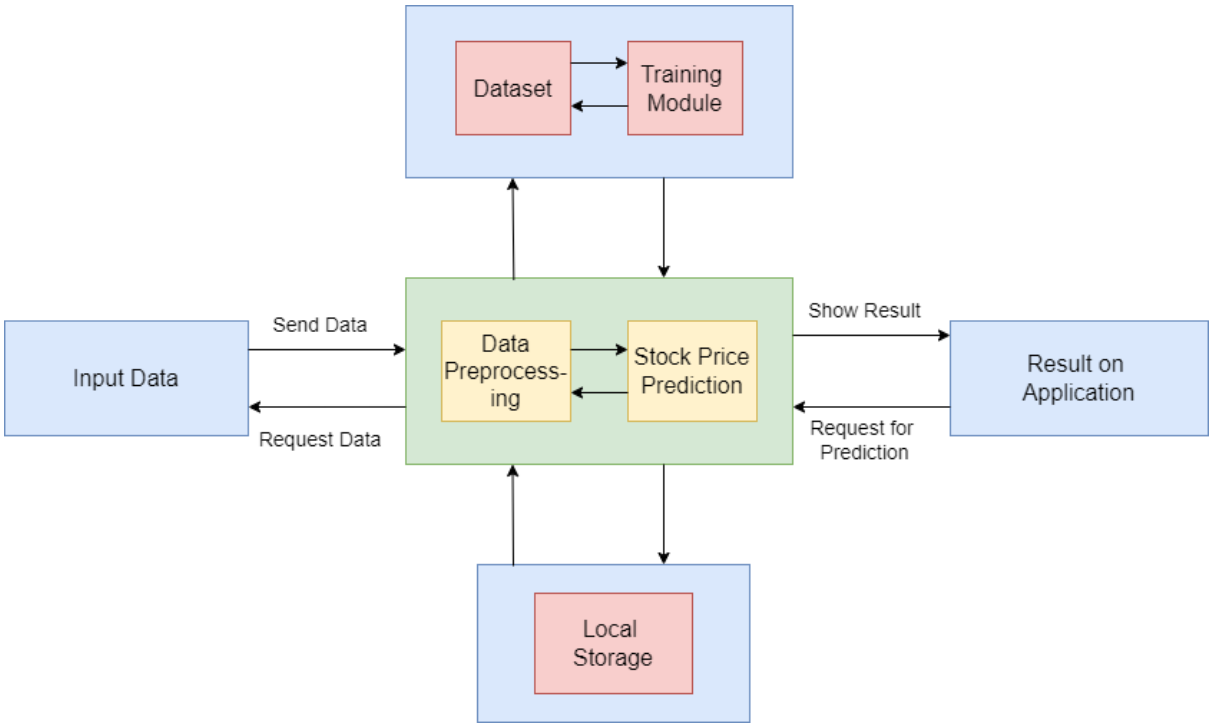


Figure: 4.3

Let's break down the key points of the Data Flow Diagram (DFD) level 2 for a stock prediction system using regression and LSTM:

4..3.1 Input Data:

- Represents where the data enters the system.
- Could include historical stock prices, trading volumes, or other relevant financial data.
- Sent to both the "Data Preprocessing" module and the "Request Data" process.

4.3.2 Data Preprocessing:

- Cleans and prepares the input data for analysis.
- Involves tasks like normalization, handling missing values, and transforming variables.
- Outputs processed data to both the "Dataset" and the "Training Module."

4.3.3 Dataset:

- Contains historical stock price data used for training the model.
- The "Training Module" learns from this dataset to make predictions.
- The dataset likely includes features like opening and closing stock prices.

4.3.4 Training Module:

- Utilizes regression and LSTM models.
- Trains on historical stock price data from the dataset.
- Learns patterns to predict future stock prices.

4.3.5 Stock Price Prediction:

- Takes pre-processed data and uses trained models to predict future stock prices.
- Considers past trends and patterns identified in historical data.
- Outputs predictions based on the input data.

4.3.6 Local Storage:

- Stores predicted results for further analysis or reference.
- Ensures data doesn't need to be fetched repeatedly.
- May be used by other components within the system.

4.3.7 Result on Application/Show Result/Request for Prediction:

- These processes manage how predicted results are displayed on an application interface.
- Users can request stock price predictions, and the system responds with the predicted values.

4.4 USE CASE:

Figure: 4.4

Description:

4.4.1. Actors:

- **User:** Represents the individual or entity interacting with the system for stock price prediction.
- **ML Model:** Represents the machine learning model used for predicting stock prices.

4.4.2. Use Cases:

- **Collect Data:** The user or system collects historical stock price data from various sources.

- **Preprocess Data:** The user or system preprocesses the collected data to make it suitable for training the machine learning model.

- **Train Model:** The user or system trains the machine learning model using the pre-processed data.

- **Evaluate Model:** The user or system evaluates the trained model to assess its performance.

- **Tune Hyperparameters:** The user or system fine-tunes the hyperparameters of the machine learning model to improve its performance.

- **Deploy Model:** The user or system deploys the trained model into a production environment.

- **Make Predictions:** The user or system utilizes the deployed model to make predictions on future stock prices.

- **Monitor Performance:** The user or system monitors the performance of the deployed model and may update it periodically as needed.

4.4.3. Associations:

- **User to Use Cases:** The user interacts with various use cases such as collecting data, training the model, making predictions, etc.
- **ML Model to Use Cases:** The machine learning model is involved in use cases such as training, evaluation, deployment, and making predictions.

4.4.4. Include/Extend Relationships:

- **Include:** Certain use cases may include other use cases. For example, the "Train Model" use case may include "Preprocess Data" and "Collect Data" as sub-steps.
- **Extend:** Some use cases may extend others based on certain conditions or scenarios. For example, the "Tune Hyperparameters" use case may extend the "Train Model" use case if the model's performance needs improvement.

4.4.5. System Boundary:

- The system boundary encapsulates all the use cases and actors involved in the stock price prediction system.

4.5 CLASS DIAGRAM:

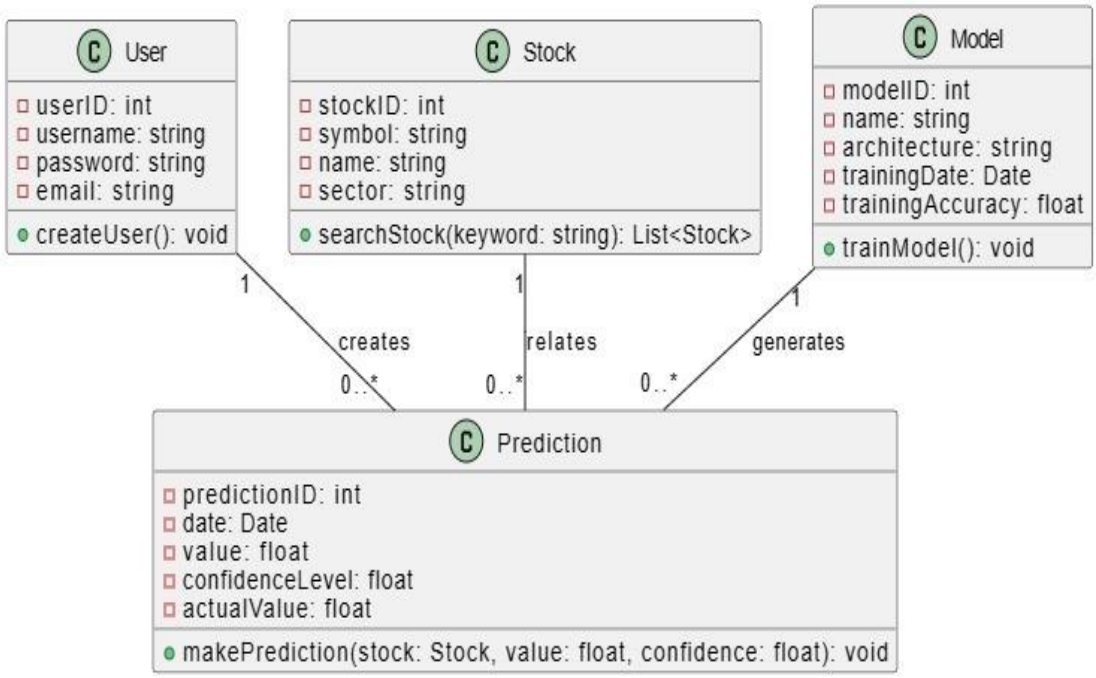


Figure: 4.5

Description:

Here are some key points to describe the class diagram for a stock prediction system that combines regression and LSTM (Long Short-Term Memory):

4.5.1 Data Flow:

- The class diagram likely includes classes representing data sources (e.g., historical stock prices, financial indicators), data preprocessing (e.g., normalization, feature extraction), and model training.
- Arrows connecting classes indicate data flow between them.

4.5.2 Regression Classes:

- There should be classes related to regression models (e.g., linear regression, polynomial regression).
- These classes handle features like historical stock prices, volume, and other relevant data.
- Regression models predict stock prices based on historical trends.

4.5.3 LSTM Classes:

- LSTM-related classes represent the neural network architecture.
- These classes manage sequences of historical data (e.g., stock prices over time).
- LSTM models capture temporal dependencies and patterns.

4.5.4 Model Evaluation:

- There might be classes for evaluating model performance (e.g., Mean Absolute Error, R-squared).
- These classes assess how well the combined regression-LSTM model predicts stock prices.

4.5.5 Integration:

- Classes connecting regression and LSTM components facilitate communication between the two models.
- Integration classes handle data transformation and feature extraction.

4.6 ACTIVITY DIAGRAM:

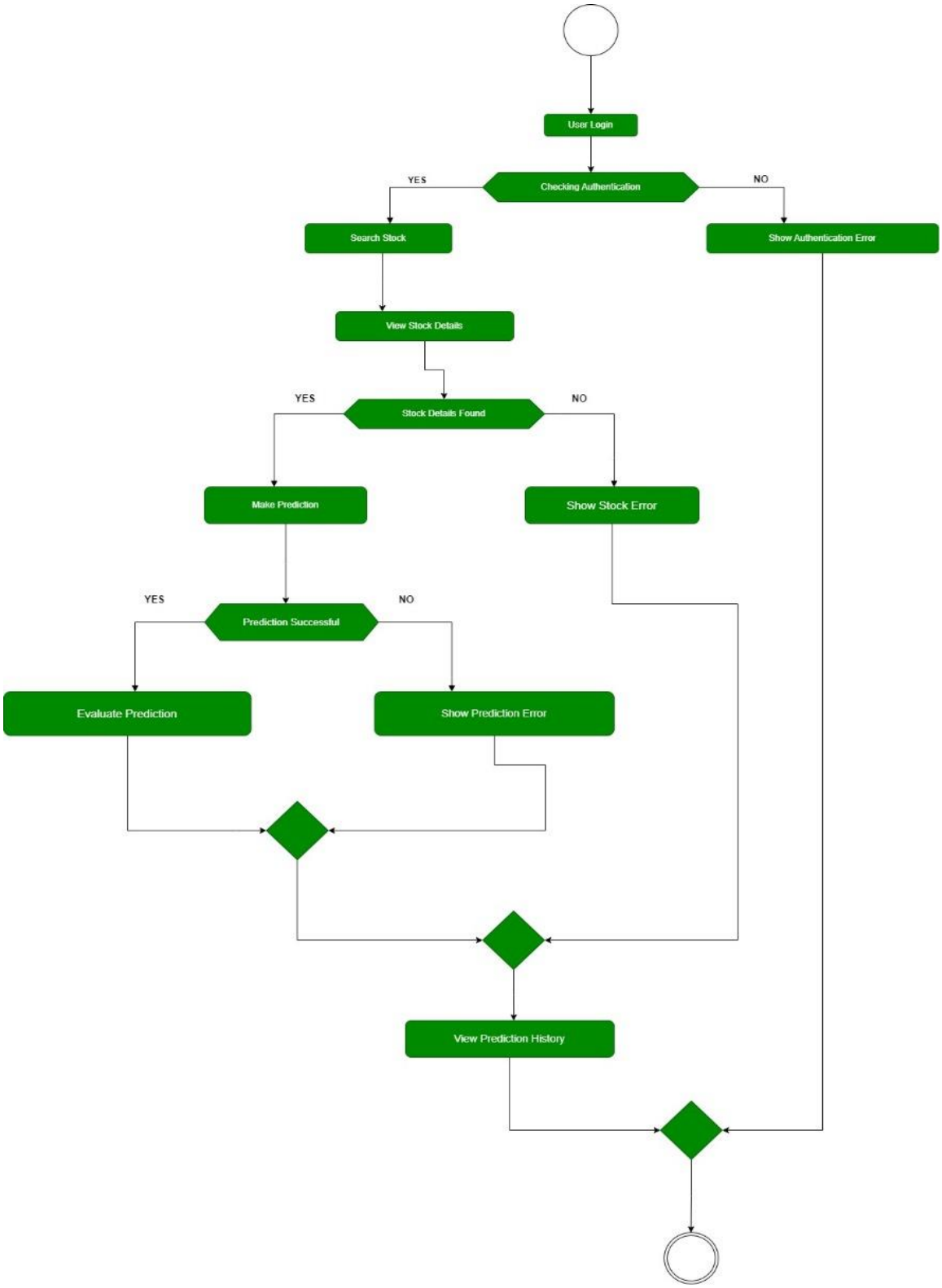


Figure:4.6

Here is the breakdown of activity diagram for the stock prediction system using regression and LSTM. Here are the key points:

4.6.1 Start of the Process:

- The user initiates the LSTM model for stock prediction.
- This step represents the beginning of the system.

4.6.2 Checking Algorithm Convergence:

- The system verifies whether the algorithm has converged.
- If successful, it proceeds to the next step; otherwise, it shows an error.
- Convergence ensures that the model has reached a stable state.

4.6.3 Inputting Stocks:

- Users input stock data into the system.
- This data serves as the basis for predictions.
- The system processes historical stock prices and other relevant information.

4.6.4 View Data Details:

- Users have the option to view data details.
- If there's an error in the stock data (e.g., missing values, inconsistencies), it is displayed.
- Otherwise, the system moves forward.

4.6.5 Extracting Stock Trends:

- The system analyzes historical stock data to extract trends.
- Trends may include moving averages, volatility, or other relevant indicators.
- These trends serve as input features for the prediction model.

4.6.6 Making Predictions:

- Using regression and LSTM, the system predicts future stock prices.
- Regression models capture linear relationships, while LSTM models handle temporal dependencies.
- If the prediction process fails (e.g., due to insufficient data), a prediction error is shown.

4.6.7 Evaluating Predictions:

- The system evaluates the accuracy and reliability of the predictions.

- Metrics such as Mean Absolute Error (MAE) or Root Mean Squared Error (RMSE) are used.
- Reliable predictions contribute to informed investment decisions.

4.6.8 View Prediction History:

- Users can access the history of predictions made by the model.
- This allows them to track performance over time.
- The system concludes the process after evaluation.

4.7 ER DIAGRAM

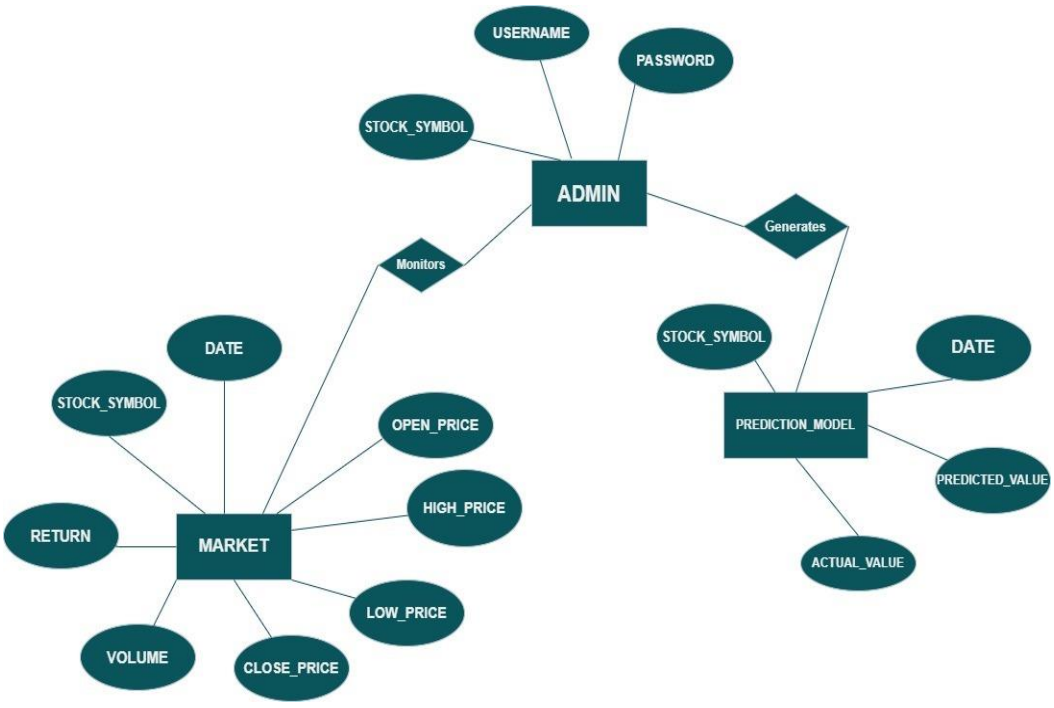


Figure:4.7

Let’s delve into the ER diagram for the stock prediction system using regression and LSTM. Here are the key points:

4.7.1 Entities:

4.7.1.1 Admin Entity: Represents the system administrator. It has attributes like "USERNAME" and "PASSWORD" for secure access.

4.7.1.2 Market Entity: Contains detailed information about stocks:

- Attributes include "OPEN PRICE," "HIGH PRICE," "LOW PRICE," "CLOSE PRICE," "VOLUME," "RETURN," "STOCK SYMBOL," and "DATE."
- These attributes provide comprehensive data for analysis and prediction.

4.7.1.3 Prediction Model Entity: Linked to the stock symbol and date:

- Generates predicted values based on actual stock data.
- Represents the dynamic predictive process.

4.7.2 Relationships:

4.7.2.1 Admin Monitors Market:

- Indicates that the admin oversees market data.
- Admin can access stock-related information.

4.7.2.2 Admin Generates Prediction Model:

- Depicts the admin's role in initiating predictions.
- The prediction model uses actual values to generate predicted values.

4.7.3 Attributes:

4.7.3.1 Market Entity Attributes:

- "OPEN PRICE," "HIGH PRICE," "LOW PRICE," "CLOSE PRICE," "VOLUME," and "RETURN" provide historical stock data.
- "STOCK SYMBOL" and "DATE" uniquely identify each stock record.

4.7.3.2 Prediction Model Attributes:

- Uses an "ACTUAL VALUE" attribute to generate a "PREDICTED VALUE."
- The prediction model refines its predictions over time.AC.

4.8 SEQUENCE DIAGRAM:

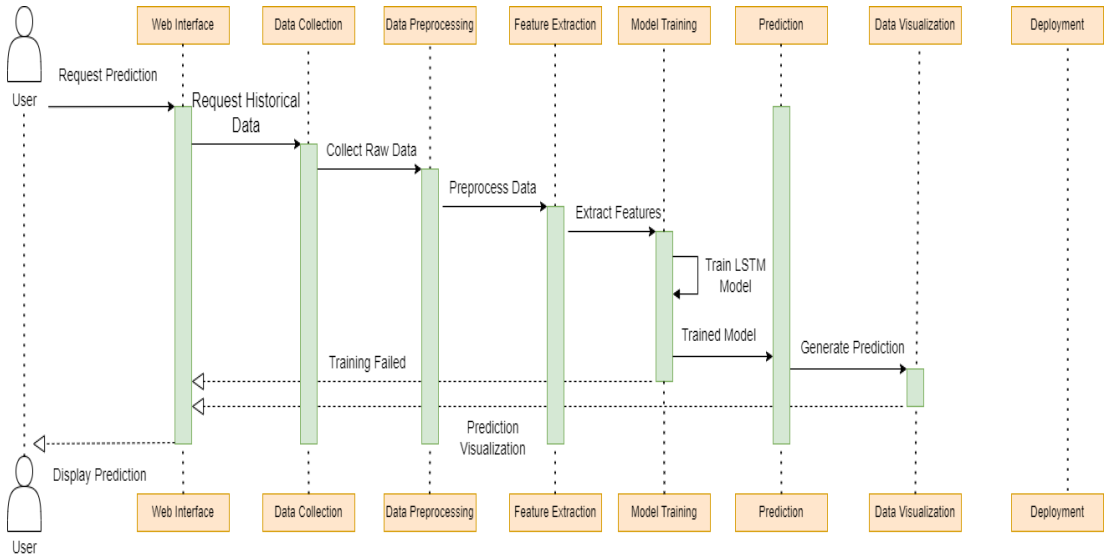


Figure: 4.8.

4.8.1 User Interaction:

- The leftmost lifeline represents the user.
- The user interacts with the web interface by making a request for a prediction.

4.8.2 Data Collection and Preprocessing:

- The web interface initiates data collection by requesting historical data.
- Raw data is collected and then pre-processed.
- Pre-processing involves cleaning, transforming, and organizing the data for further use.

4.8.3. Feature Extraction:

- After preprocessing, the system extracts relevant features from the data.
- These features are essential for training the model.

4.8.4. Model Training:

- The LSTM (Long Short-Term Memory) model is trained using the extracted features.
- If training fails (indicated by "Training Failed"), the process may be restarted.
- If successful, the trained model is ready for predictions.

4.8.5. Prediction Generation and Visualization:

- The trained model generates predictions based on new input data.
- These predictions are visualized for the user.

4.8.6. Deployment:

- Finally, the results are deployed back to the web interface for display to the user.

4.9 PACKAGE DIAGRAM

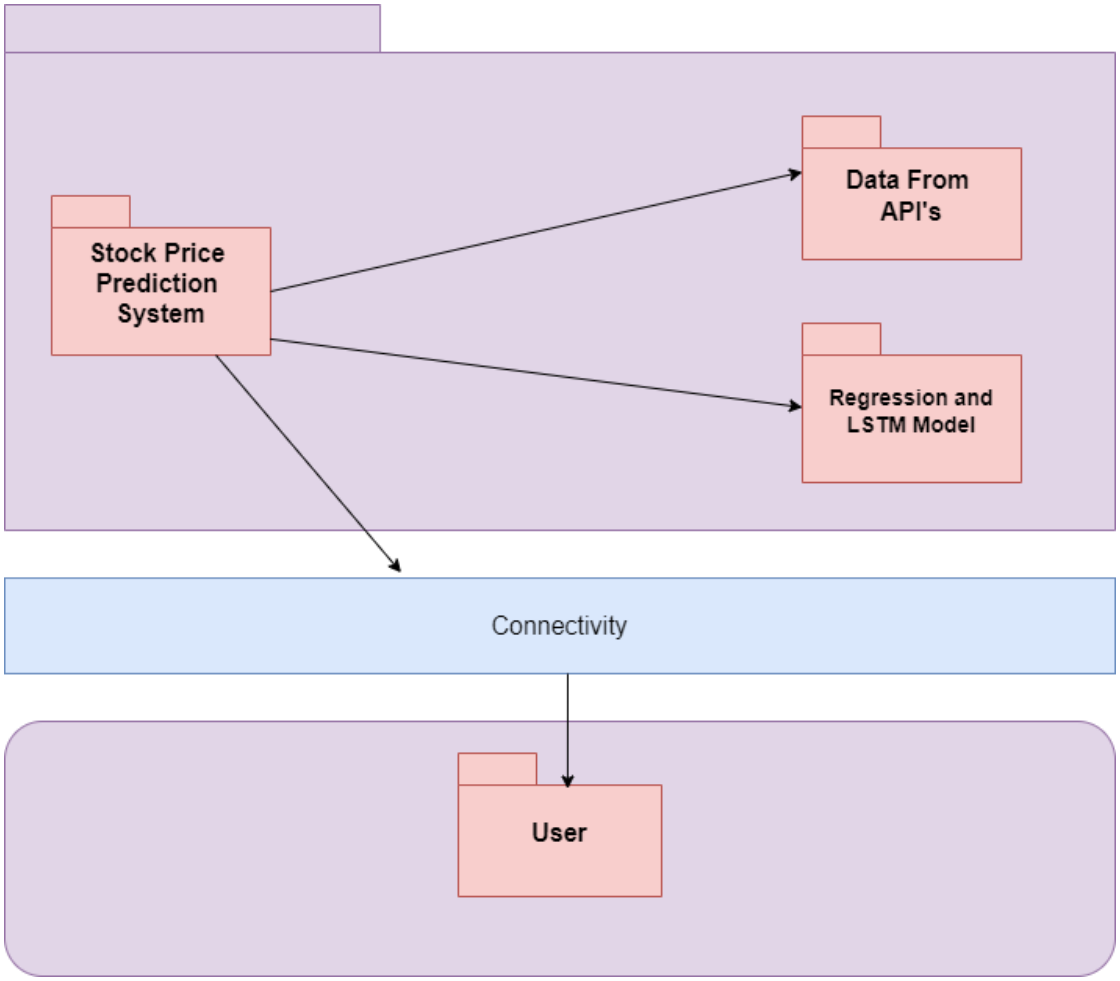


Figure: 4.9

Let's discuss the package diagram for a Stock Price Prediction System using LSTM and Regression Models. Here are the key points:

4.9.1 Stock Price Prediction System:

The central component orchestrates the entire prediction process. It receives data from external sources (such as APIs) and interacts with the prediction models. Its key responsibilities include data preprocessing, model integration, and result dissemination.

4.9.1.1 Data Preprocessing: This step involves cleaning, transforming, and preparing raw data for modelling. It ensures that the data is in a suitable format for

the regression and LSTM models.

4.9.1.2 Model Integration: The system combines regression and LSTM models to achieve accurate predictions. Regression models capture linear relationships between input features (e.g., historical prices) and the target (future price). LSTM models, on the other hand, handle time-series data effectively by learning patterns and dependencies over time.

4.9.1.3 Result Dissemination: The Stock Price Prediction System provides predicted stock prices to users. This could be through a web application, mobile app, or API.

4.9.2 Data From API's:

- Represents the data source where historical stock market data is obtained. Common APIs include Yahoo Finance, Alpha Vantage, or other financial data providers.
- The data includes daily stock prices, trading volumes, and other relevant metrics.
- The system fetches data from these APIs to train and validate the regression and LSTM models.

4.9.3 Regression and LSTM Model:

These models analyze historical data to predict future stock prices.

4.9.3.1 Regression Model:

- Utilizes techniques like linear regression.
- Captures linear relationships between input features (e.g., historical prices) and the target (future price).
- Simple and interpretable.

4.9.3.2 LSTM Model:

- A type of recurrent neural network (RNN).
- Handles time-series data effectively.
- Learns complex patterns and dependencies over time.
- Suitable for capturing non-linear relationships in stock prices.

4.9.4 Connectivity:

- The blue bar labelled "Connectivity" emphasizes the interaction between components.

- Users can access predictions through an interface (web app, mobile app, or API).
- The system ensures seamless communication between data, models, and user.

4.10 COMPONENT DIAGRAM

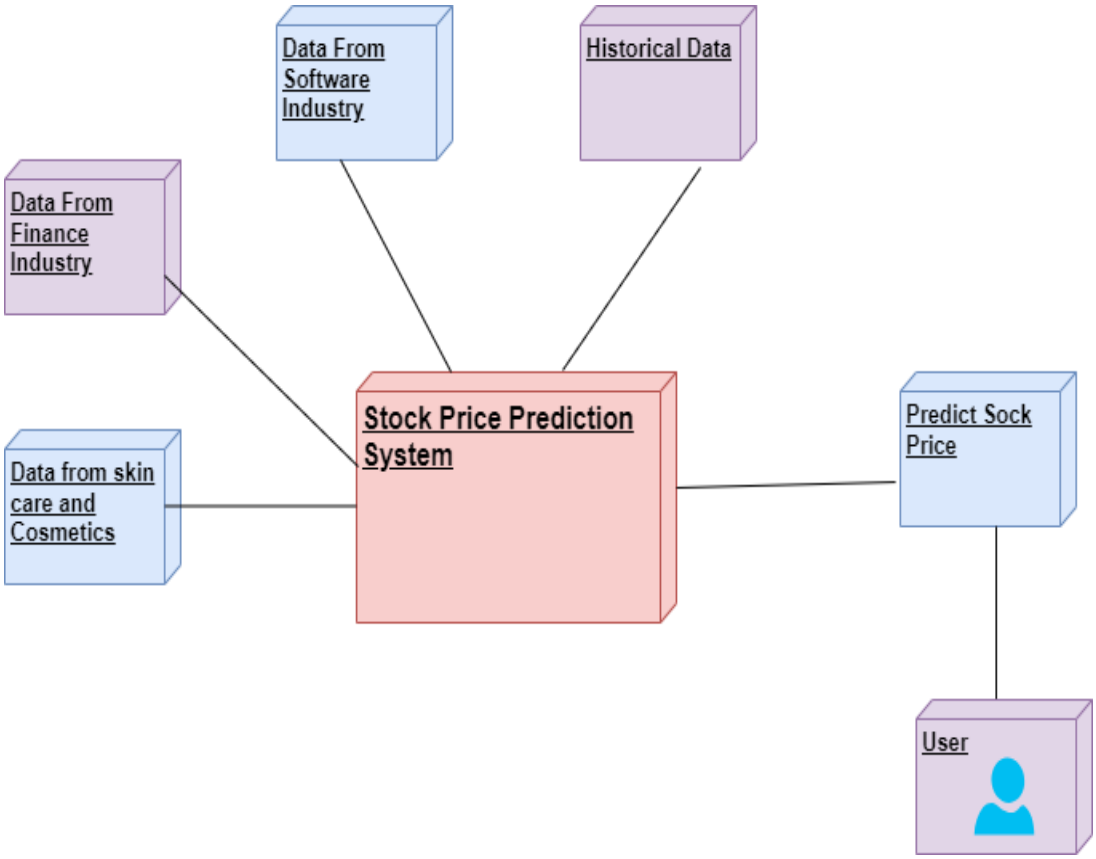


Figure: 4.10

Here's the component diagram for the "Stock Price Prediction System" and elaboration of its components.

4.10.1. Stock Price Prediction System (Central Component):

- The large pink rectangle labelled "Stock Price Prediction System" represents the core component of the diagram. This system aims to predict stock prices based on various inputs.
- It serves as the central hub for processing data and generating predictions.

4.10.2. Data Sources:

- **Data from Finance Industry:** The light purple rectangle in the top left corner represents financial industry data. This could include historical stock prices, economic indicators, and other relevant financial information.
- **Data from Software Industry:** Another light purple rectangle adjacent to the first one signifies software industry data. This might include software development trends, technology news, or other relevant data.
- **Historical Data:** The blue rectangle indicates historical stock performance data. This input is crucial for training predictive models.

4.10.3. Prediction Process:

- The arrow pointing away from the central system leads to a light blue rectangle labelled "Predict Stock Price." This component represents the prediction process.
- The system likely uses machine learning algorithms, statistical models, or other techniques to generate stock price predictions.

4.10.4. User Interaction:

- The figure of a user (depicted as a head and shoulders icon) in the bottom right corner suggests user interaction with the system.
- However, the specific nature of this interaction is not detailed in the diagram. Users might provide additional input, adjust parameters, or receive predictions.

4.10.5. Additional Data Source:

- The light purple rectangle labelled "Data from Skin Care and Cosmetics" at the bottom left corner represents data from a specific industry. It's interesting to see how diverse data sources contribute to stock price predictions.

In summary, the "Stock Price Prediction System" integrates data from various industries, historical records, and user interactions to generate stock price forecasts. The central system processes these inputs and produces predictions, which can be valuable for investors, traders, and financial analysts.

CHAPTER-5

IMPLEMENTATION

5.1. LSTM MODEL

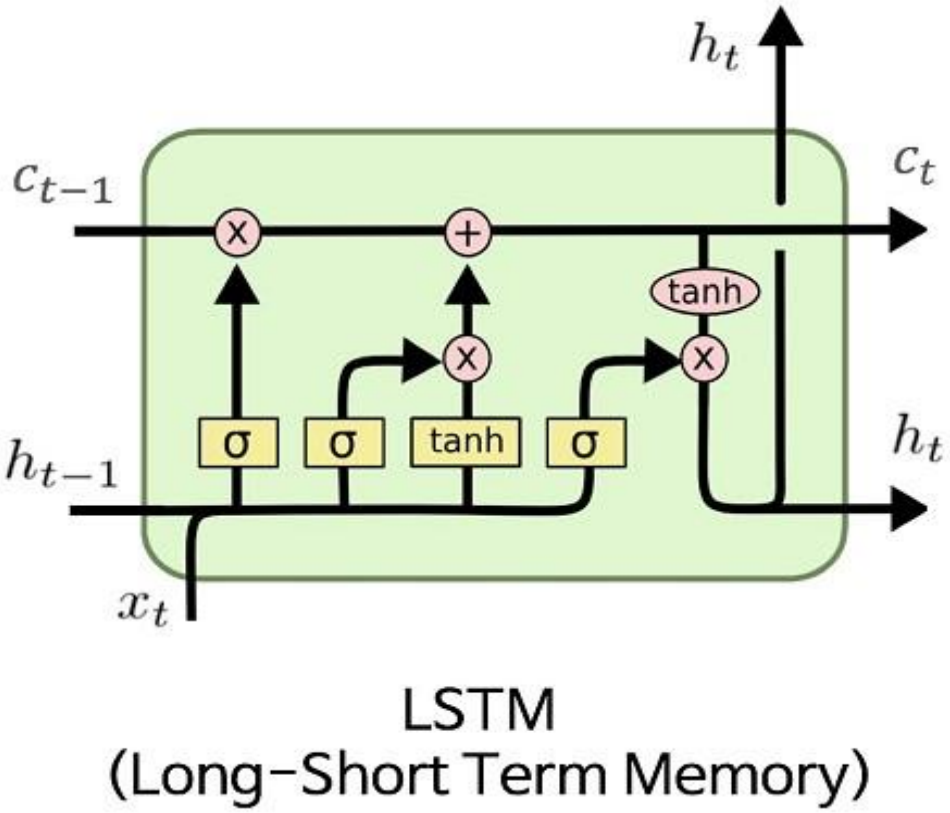


Figure: 5.1

Long Short-Term Memory is a kind of recurrent neural network. In RNN output from the last step is fed as input within the present step. It tackled the matter of long-term dependencies of RNN within which the RNN will not predict the word hold on within the long-term memory however can offer additional accurate forecasts from the recent info. Because the gap length will increase. RNN does not offer an economical performance.

LSTM will by default retain the knowledge for a long period of time. It is used for processing, predicting and classifying on the basis of time-series data.

5.1.1. Structure of LSTM:

- **Memory Cells:** LSTMs consist of memory cells that maintain a cell state and various gates (input gate, forget gate, output gate) that regulate the flow of information into and out of the cells.
- **Long-Term Dependencies:** LSTMs are designed to capture long-term dependencies in sequential data by allowing information to persist over multiple time steps.
- A memory cell contains three main gates:
    1. **Input gate-** a new value flows into the memory cell.
    2. **Forget gate-** a value remains in the memory cell.
    3. **Output gate-** value in the memory cell is used to compute the output.

5.1.2. Training:

- **Backpropagation Through Time (BPTT):** LSTMs are trained using BPTT, a variant of backpropagation suitable for sequential data. During training, the network learns to update its internal parameters (weights and biases) to minimize the difference between the predicted and actual values of the target variable.

- **Gradient Clipping:** To mitigate the vanishing/exploding gradient problem in deep neural networks, gradient clipping is often used during training to limit the magnitude of gradients.

5.1.3 Prediction:

Sequence Prediction: LSTMs can predict sequences of future values based on the input sequence of historical data. Each prediction is made by feeding the model's output back into itself as input for the next time step.

5.1.4. Hyperparameters:

- **Number of Layers**: The depth of the LSTM network, i.e., the number of LSTM layers stacked on top of each other.
- **Number of Units:** The number of memory cells (units) in each LSTM layer.
- **Learning Rate:** The rate at which the model parameters are updated during training.
- **Batch Size:** The number of training examples used in each iteration of training.
- **Sequence Length:** The number of time steps used as input to the LSTM network during training and prediction.

5.1.5. **Implementation**:

5.1.5.1 The first step in LSTM is to decide which information to be omitted from the cell in that particular step. It is decided with the help of a sigmoid function. It looks at the previous state (ht-1) and the current input (xt) and computes the function.

5.1.5.2 There are two functions in the second layer. The first is the sigmoid function, and the second is the tanh function. The sigmoid function decides which values to let through (0 or 1). The tanh function gives the weightage to the values passed, deciding their level of importance from -1 to 1.

5.1.5.3 The third step is to decide what will be the final output. First you need to run a sigmoid layer which determines what parts of the cell state make it upto the output. Then, you must put the cell state through the tanh function to push the values between -1 and 1 and multiply it by the output of the sigmoid gate.

5.2. Regression Model:

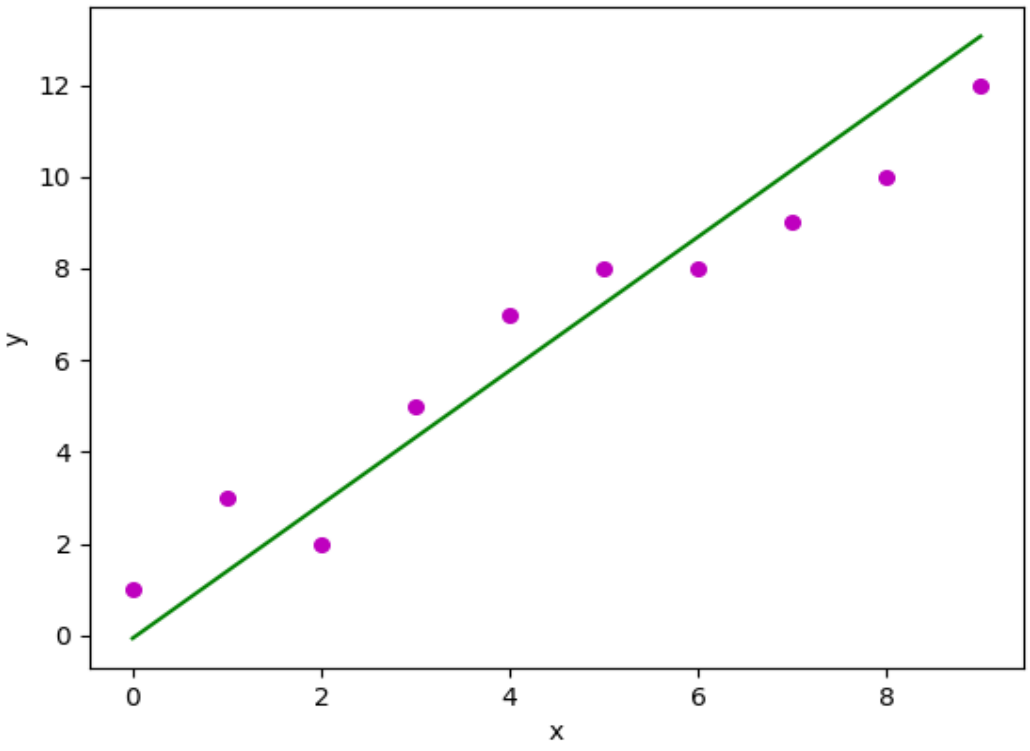


Figure: 5.2

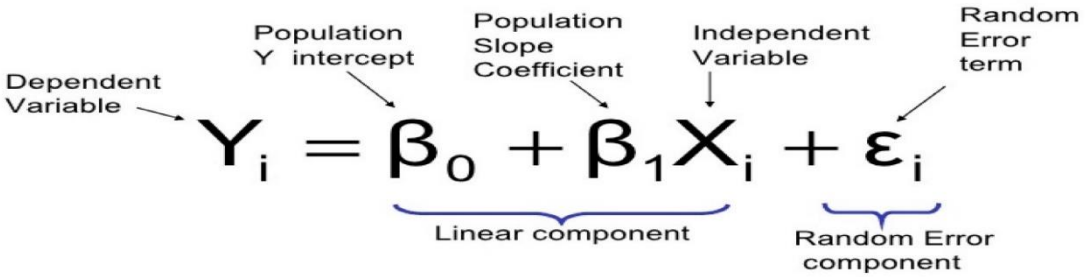
5.2.1. Working Of Linear Regression:

- **Model Representation:** In linear regression, the relationship between the input features 'X' and the target variable 'Y' is represented by a linear equation: $y = W_o + W_1X_1 + W_2X_2 + \ldots\ldots + WnXn + E$, where W_0, W_1,….., Wn are the model coefficients, X_1, X_2,….., Xn are the input features, and E represents the error term.

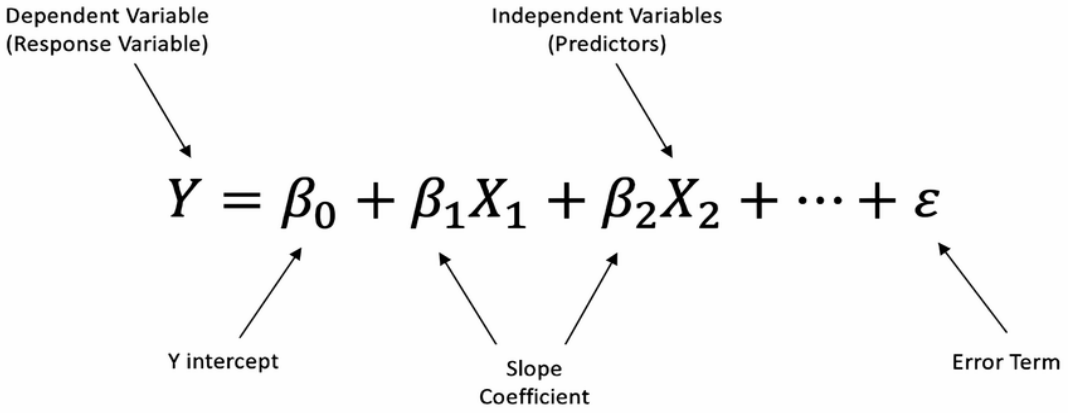
- **Training:** During training, the model learns the optimal values of the coefficients W_0, W_1,….., Wn that minimize the difference between the predicted and actual values of the target variable Y. This is typically done using an optimization algorithm such as ordinary least squares (OLS) or gradient descent.

- **Prediction:** Once trained, the model can be used to make predictions on new input data by plugging in the values of the input features into the learned linear equation.

- **Equation used:**



OR



Equation: 5.1

5.3. CNN architecture:

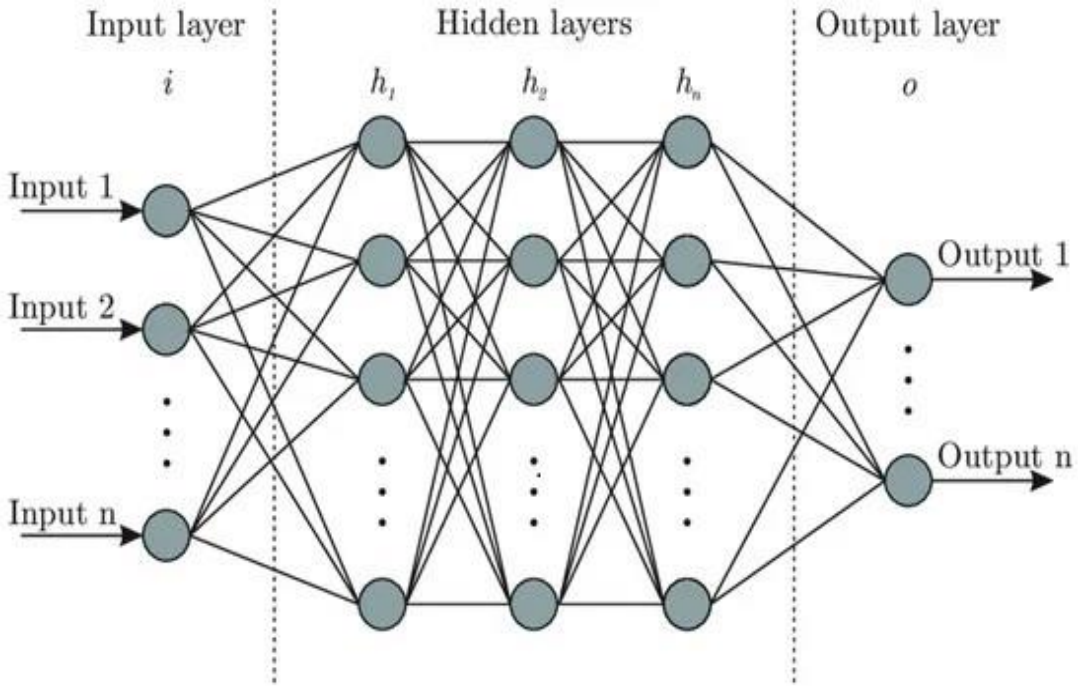


Figure 5.3

5.3.1 Layer of CNN model:

- Convolution
- MAX Pooling
- Dropout
- Flatten
- Dense
- Activation

5.3.2. Description:

5.3.2.1 Convolution: In the Convolution extract the featured from the input image. It given the output in matrix form.

5.3.2.2 MAX Pooling: In the MAX polling it takes the largest element from a rectified feature map.

5.3.2.3 Dropout: Dropout is randomly selected neurons are ignored during training.

5.3.2.4 Flatten: Flatten feed output into a fully connected layer. It gives data in list form.

5.3.2.5 Dense: A Linear operation in which every input is connected to every output by weight. It followed by a nonlinear activation function.

5.3.2.6 Activation: It used sigmoid function and predict the probability 0 and 1[13].

5.3.3. Working of CNN:

- **Data Representation:** Stock price data is typically represented as a time series, with each data point representing the price of a stock at a specific time. To apply CNNs, the time series data needs to be converted into a format that can be fed into the network. This can be achieved by treating the time series as an image, where the x-axis represents time and the y-axis represents the price values.

- **Feature Extraction:** CNNs excel at automatically learning hierarchical features from input data. In the context of stock price prediction, the CNN can learn to extract relevant patterns and features from the historical price data. This could include identifying trends, seasonality, or other patterns that may influence future price movements.

- **Convolutional Layers:** The convolutional layers in the CNN apply filters/kernels to the input data, extracting features at different levels of abstraction. In the case of stock price prediction, these filters could capture patterns in the time series data, such as short-term fluctuations or longer-term trends.

- **Pooling Layers:** Pooling layers down sample the feature maps produced by the convolutional layers, reducing the dimensionality of the data while preserving the most important features. This helps in making the model more robust to variations and noise in the data.

- **Flattening and Dense Layers:** After the convolutional and pooling layers, the resulting feature maps are flattened into a one-dimensional vector and passed through one or more dense (fully connected) layers. These layers perform classification or regression tasks based on the learned features, mapping them to the desired output (in this case, future stock prices).

- **Output Layer:** The output layer of the CNN produces the final predictions for future stock prices. Depending on the specific task, this could be a single output node for regression tasks (predicting a continuous value) or multiple output nodes for classification tasks (predicting discrete categories).

- **Training:** The CNN is trained using historical stock price data, where the input consists of sequences of past price observations, and the target output is the price at a future time step. During training, the network adjusts its parameters (weights and biases) using optimization algorithms like stochastic gradient descent to minimize the prediction error.

- **Evaluation and Testing:** After training, the performance of the CNN is evaluated on a separate validation or testing dataset to assess its accuracy and generalization ability. Common evaluation metrics for regression tasks include mean squared error (MSE) or mean absolute error (MAE).

- Overall, CNNs can capture complex patterns and relationships in sequential data like stock prices, making them a potentially powerful tool for prediction tasks. However, it's essential to carefully design the network architecture, preprocess the data appropriately, and fine-tune the model parameters to achieve optimal performance. Additionally, incorporating other features and techniques alongside CNNs, such as recurrent neural networks (RNNs) or attention mechanisms, may further improve prediction accuracy [15].

5.4 Hybrid Approach of LSTM + CNN

In the hybrid approach, the Convolutional Neural Networks (CNNs) offer benefits in choosing sensible options and Long Short-Term Memory (LSTM) networks have proven sensible skills to find out to learn sequential data. Each approach is reported to produce improved result. CNNs to possess to convolute filters over every input layer so as to get the simple options and CNNs have shown enhancements in computer vision, natural language processing and different tasks [14]. CNN may be a powerful tool to pick out features in order to improve the prediction accuracy [15]. The capabilities of LSTMs in learning data series by considering the previous outputs [16].

The multiple convolutional filters slide over the matrix to produce a new feature map and also the filters have numerous completely different sizes to generate different features. The Max-pooling layer is to calculate the most value as a corresponding feature to a particular filter. The output vectors of the Max-pooling layer become inputs to the LSTM networks to measure the long-run dependencies of feature sequences. One in all the benefits of the LSTMs is that the ability to capture the sequential data by considering the previous data. This layer takes the output vectors from the dropout layer as inputs. This layer includes a set number of units or cells and also the input of every cell is that the output from the dropout layer. The final output of this layer has the same number of units within the network the outputs from LSTMs are merged and combined in

one matrix then passed to a fully connected layer. The array is converted into a single output in the range between 0 and 1 using the fully connected layer, in order to be finally classified using sigmoid function [17].

CHAPTER-6

RESULT AND DISCUSSION

Loading and importing dataset

Out[4]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2018-02-05	262.000000	267.899994	250.029999	254.259995	254.259995	11896100
1	2018-02-06	247.699997	266.700012	245.000000	265.720001	265.720001	12595800
2	2018-02-07	266.579987	272.450012	264.329987	264.559998	264.559998	8981500
3	2018-02-08	267.079987	267.619995	250.000000	250.100006	250.100006	9306700
4	2018-02-09	253.850006	255.800003	236.110001	249.470001	249.470001	16906900

Figure: 6.1

Here it is returning top 5 values of it by using head() function.

Dataset Information

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1009 entries, 0 to 1008
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       1009 non-null   datetime64[ns]
 1   Open       1009 non-null   float64
 2   High       1009 non-null   float64
 3   Low        1009 non-null   float64
 4   Close      1009 non-null   float64
 5   Adj Close  1009 non-null   float64
 6   Volume     1009 non-null   int64
dtypes: datetime64[ns](1), float64(5), int64(1)
memory usage: 55.3 KB
```

Figure: 6.2

Defines total rows*cols= 1008*7 as described in picture and also returns datatype of each column and also checks for null values present in the dataset. The dataset, structured as a pandas Data Frame, comprises 1009 entries and spans 7 columns: Date, Open, High, Low, Close, Adj Close, and Volume. Each column contains 1009 non-null values, indicating the absence of missing data. The data types of the columns are as follows: Date is

datetime64[ns], Open, High, Low, Close, and Adj Close are all float64, and Volume is int64. The dataset's memory usage is 55.3 KB. This comprehensive dataset likely represents a time series of stock market data, providing essential information for financial analysis and modelling.

Stock prices are from February 2018 to February 2022.

```
Dataframe contains stock prices between 2018-02-05 00:00:00 2022-02-04 00:00:00
Total days =  1460 days
```

Out[7]:

	Date	Open	High	Low	Close	Adj Close	Volume
count	1009	1009.000000	1009.000000	1009.000000	1009.000000	1009.000000	1.009000e+03
mean	2020-02-05 07:30:58.870168320	419.059673	425.320703	412.374044	419.000733	419.000733	7.570685e+06
min	2018-02-05 00:00:00	233.919998	250.649994	231.229996	233.880005	233.880005	1.144000e+06
25%	2019-02-06 00:00:00	331.489990	336.299988	326.000000	331.619995	331.619995	4.091900e+06
50%	2020-02-06 00:00:00	377.769989	383.010010	370.880005	378.670013	378.670013	5.934500e+06
75%	2021-02-05 00:00:00	509.130005	515.630005	502.529999	509.079987	509.079987	9.322400e+06
max	2022-02-04 00:00:00	692.349976	700.989990	686.090027	691.690002	691.690002	5.890430e+07
std	NaN	108.537532	109.262960	107.555867	108.289999	108.289999	5.465535e+06

Figure: 6.3

We have been calculating date and time of the given dataset using pandas library and date-time function and summary statistics for our data using the describe function. The Data Frame contains Netflix stock prices from February 5, 2018, to February 4, 2022, covering 1,460 days, with columns like 'Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', and 'Volume'. Descriptive statistics provide key metrics such as mean, min, max, and quartiles. For further analysis, you can visualize closing prices with line or candlestick charts, calculate moving averages to identify trends, and analyze volatility through daily returns and rolling standard deviations. Advanced options include correlation analysis with other stocks or using machine learning models to predict future prices. Let me know your preferred direction for detailed assistance.

Box plot visually checker outlier

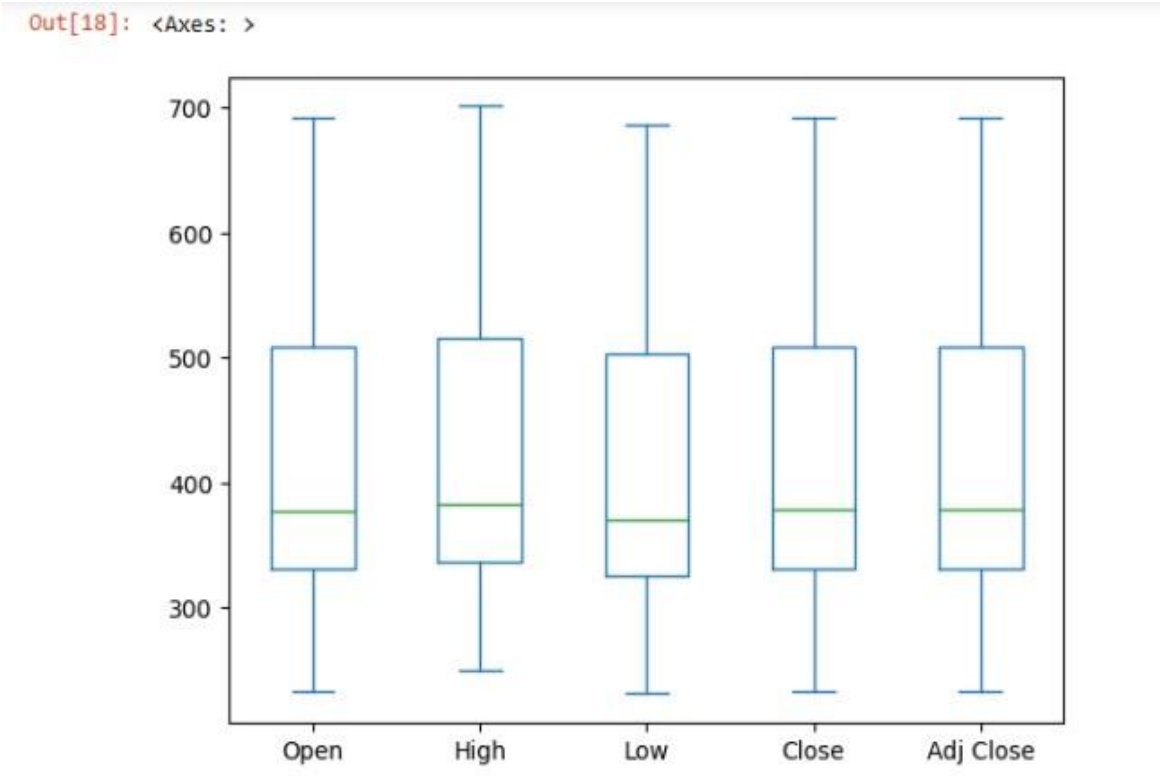


Figure: 6.4

Stock Prices of Netflix plotted here from 2018 to 2022 actual stock values

Figure: 6.5

Actual vs predicted values using Regression

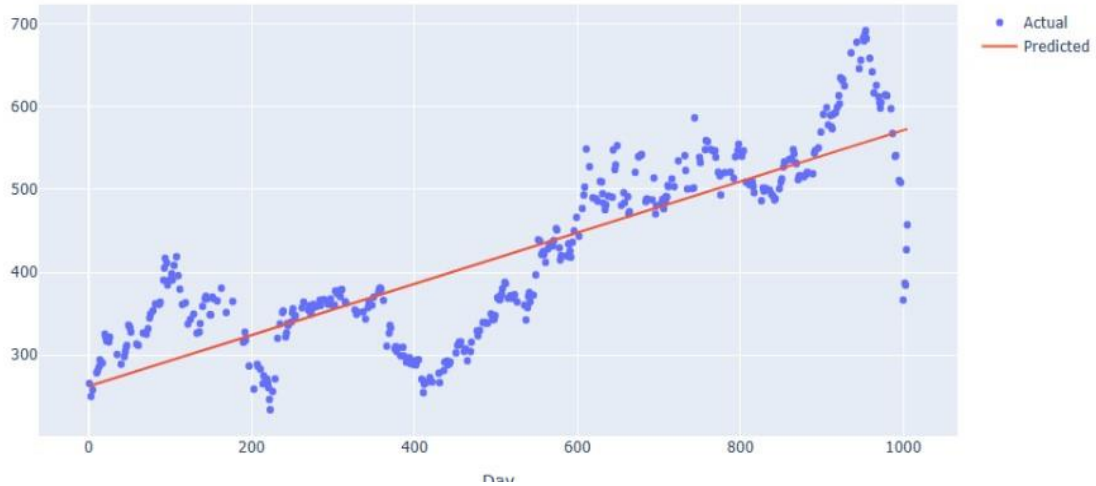


Figure: 6.6

Load the Training Dataset For LSTM

```
Out[5]: <bound method DataFrame.info of            Date        Open        High         Low       Close   Adj Close  \
0     2018-02-05  262.000000  267.899994  250.029999  254.259995  254.259995
1     2018-02-06  247.699997  266.700012  245.000000  265.720001  265.720001
2     2018-02-07  266.579987  272.450012  264.329987  264.559998  264.559998
3     2018-02-08  267.079987  267.619995  250.000000  250.100006  250.100006
4     2018-02-09  253.850006  255.800003  236.110001  249.470001  249.470001
...          ...         ...         ...         ...         ...         ...
1004  2022-01-31  401.970001  427.700012  398.200012  427.140015  427.140015
1005  2022-02-01  432.959991  458.480011  425.540009  457.130005  457.130005
1006  2022-02-02  448.250000  451.980011  426.480011  429.480011  429.480011
1007  2022-02-03  421.440002  429.260010  404.279999  405.600006  405.600006
1008  2022-02-04  407.309998  412.769989  396.640015  410.170013  410.170013

        Volume
0     11896100
1     12595800
2      8981500
3      9306700
4     16906900
...        ...
1004  20047500
1005  22542300
1006  14346000
1007   9905200
1008   7782400

[1009 rows x 7 columns]>
```

Figure: 6.7

Loading the training dataset is the initial and essential step in stock market prediction projects. This process involves importing historical stock price data, which typically includes parameters such as opening and closing prices, high and low prices, and trading volume. The data can be sourced from financial platforms like Netflix, Sun Pharma, ICICI Bank, TATA MOTARS or specialized financial databases. Once imported into the analysis environment, such as Python using libraries like Pandas, the dataset undergoes preprocessing and cleaning. This step is crucial for ensuring the accuracy and reliability of the model training and subsequent predictive analysis.

Use the Open Stock Price Column to Train Your Model

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1009 entries, 0 to 1008
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       1009 non-null   object
 1   Open       1009 non-null   float64
 2   High       1009 non-null   float64
 3   Low        1009 non-null   float64
 4   Close      1009 non-null   float64
 5   Adj Close  1009 non-null   float64
 6   Volume     1009 non-null   int64
dtypes: float64(5), int64(1), object(1)
memory usage: 55.3+ KB
```

Figure: 6.8

Utilizing the open stock price column to train your model is a strategic approach in stock market prediction. The open price is the initial trading price at the start of the trading session and can be a valuable predictor of market trends. By focusing on this column, the model can learn patterns and correlations specific to the market's opening behaviour. This data is processed and fed into machine learning algorithms, such as Linear Regression, Decision Trees, or Neural Networks, which then learn to make predictions based on the historical opening prices. This method enhances the model's ability to forecast future stock prices effectively.

Closing Price History

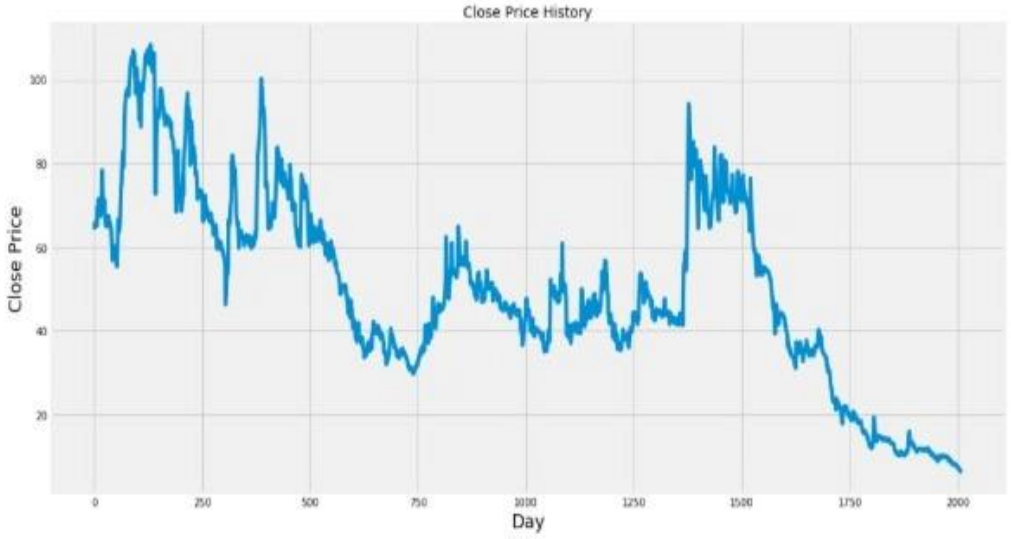


Figure: 6.9

Data Scaling

```
array([[0.56722278],
       [0.58292443],
       [0.57654561],
       ...,
       [0.00147203],
       [0.        ],
       [0.00294406]])
```

Figure: 6.10

Data scaling is a preprocessing step used to standardize the range of independent variables or features of data. It is an essential step in the data preprocessing pipeline, especially for algorithms that compute distances between data points, like k-nearest neighbours (KNN), support vector machines (SVM), and neural networks.

Dropout And LSTM

Model: "sequential_16"

Layer (type)	Output Shape	Param #
lstm_54 (LSTM)	(None, 60, 100)	42,400
dropout_44 (Dropout)	(None, 60, 100)	0
lstm_55 (LSTM)	(None, 60, 100)	80,400
dropout_45 (Dropout)	(None, 60, 100)	0
lstm_56 (LSTM)	(None, 60, 100)	80,400
dropout_46 (Dropout)	(None, 60, 100)	0
lstm_57 (LSTM)	(None, 100)	80,400
dropout_47 (Dropout)	(None, 100)	0
dense_16 (Dense)	(None, 1)	101

Total params: 283,701 (1.08 MB)

Trainable params: 283,701 (1.08 MB)

Non-trainable params: 0 (0.00 B)

Figure: 6.11

The model comprises four sequential LSTM layers, each containing 100 units, interspersed with dropout layers set at a dropout rate of 0.2 to prevent overfitting. The LSTM layers' configuration ensures that each layer processes the sequence data and passes it to the next layer, with the final LSTM layer feeding into a Dense layer with a single unit, which outputs the predicted stock price. This architecture is designed to capture temporal dependencies in the stock price data while maintaining a balance between model complexity and regularization to enhance predictive performance. The model's total trainable parameters amount to 283,701, indicating a considerable capacity to learn from the data.

Training Model Loss

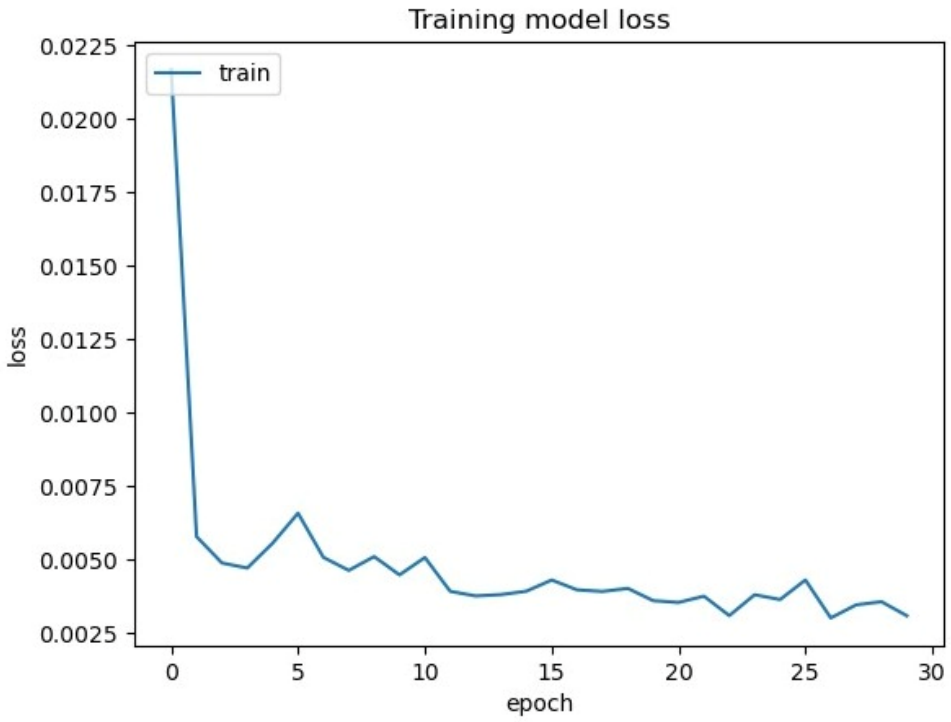


Figure: 6.12

The training model loss represents the discrepancy between the predicted values and the actual values during the training phase. It quantifies how well the model is learning from the data, with lower loss indicating better performance. Monitoring and minimizing loss are crucial for optimizing model accuracy and predictive capabilities.

The x-axis represents the number of epochs, which is set to 30, while the y-axis shows the training loss, measured using the mean squared error (MSE) metric. Initially, the loss starts at a relatively high value, indicating significant prediction errors at the beginning of training. As the training progresses, the loss sharply decreases, particularly within the first few epochs, suggesting rapid learning and improvement in the model's predictive accuracy. The curve continues to decline gradually and stabilizes, with minor fluctuations, indicating that the model is converging towards an optimal solution with diminishing returns on further training. This loss curve suggests effective training, with the model improving its performance over time.

Plot for real vs Predicted for Netflix using LSTM

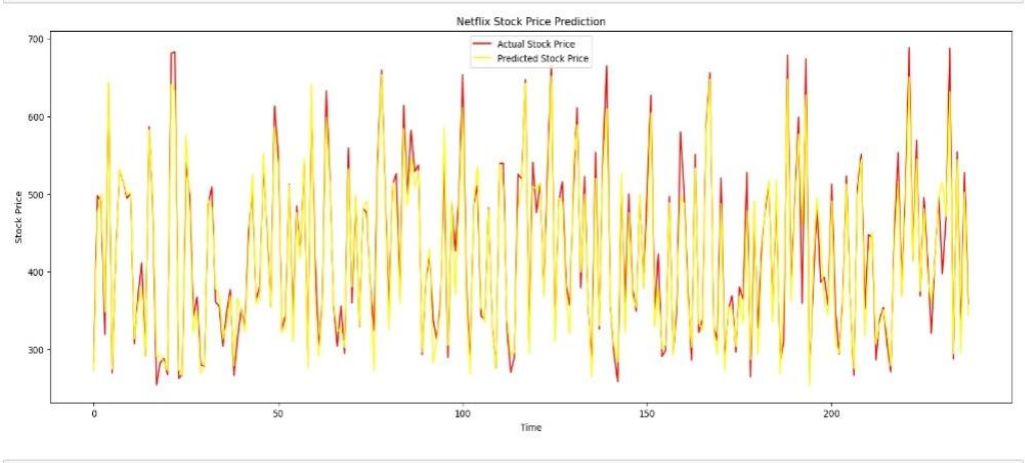


Figure: 6.13

Begin by extracting historical stock prices for Netflix, typically including open, high, low, and close prices, along with trading volume. Prepare the input data for the model by organizing it into x_train and y_train datasets, separating features and target variables. Reshape the data to match the input requirements of the model, ensuring compatibility and effectiveness during training. Once the model is trained, use it to predict future stock prices based on the prepared input data. Finally, plot the actual and predicted prices for Netflix in a visual representation, often using tools like Matplotlib or Plotly. This comparison provides insights into the model's performance and its ability to capture the underlying patterns in the stock market data. By visually inspecting the plotted prices, analysts can assess the accuracy of the predictions and make informed decisions regarding investment strategies or further model refinement.

RMSE AND R2 SCORES

```
In [205]:
from sklearn.metrics import mean_squared_error, r2_score
r2 = r2_score(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)

print("R2 Score:", r2)
#print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)

R2 Score: 0.9546609403376236
Root Mean Squared Error (RMSE): 24.17634361528765
```

Figure 6.14

The R-squared (R^2) score, also known as the coefficient of determination, is a key statistical measure used to evaluate the accuracy of a predictive model, particularly in the context of stock price prediction. An R^2 score ranges from 0 to 1, with higher values indicating a better fit between the predicted and actual stock prices. An R^2 score of 1 signifies that the model perfectly predicts the dependent variable, accounting for all the variability in the stock prices. Conversely, an R^2 score of 0 suggests that the model fails to capture any of the variability, essentially performing no better than a simple mean prediction. In the domain of stock price prediction, achieving a high R^2 score is challenging due to the inherent volatility and complexity of financial markets. Nonetheless, a higher R^2 score generally indicates a more reliable and accurate model, providing investors and analysts with greater confidence in the model's predictive power. However, it is important to consider that a high R^2 does not guarantee that the model is free from overfitting or that it will perform well on unseen data, hence other metrics and validation methods should also be employed to ensure robust and generalizable predictive performance.

Plot for real vs Predicted for Sun-pharma using LSTM

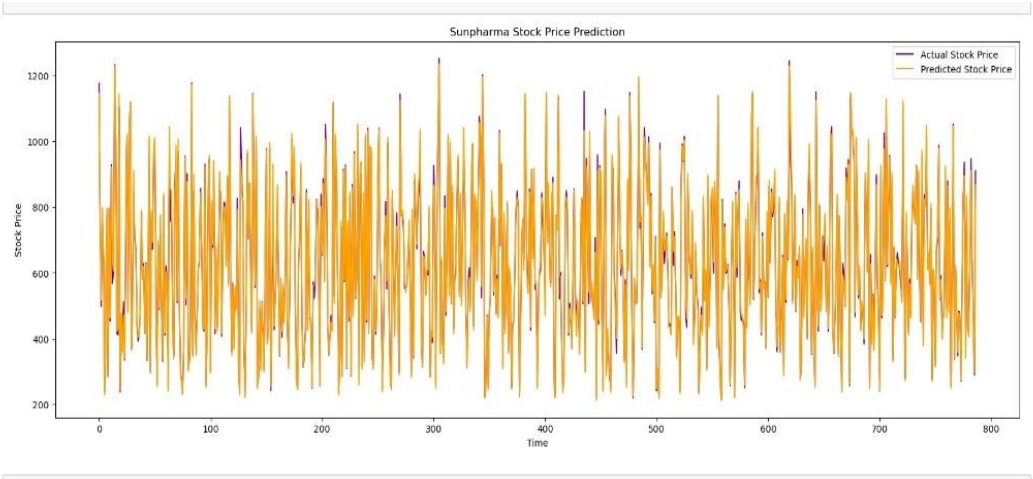


Figure 6.15

Sun-Pharma's historical stock prices are acquired and processed to construct a predictive model. Through data extraction, the historical stock data of Sun Pharma is collected. Following this, preprocessing steps involve organizing the data into input and target sets. These datasets, denoted as x_train and y_train, comprise past stock prices and corresponding future values, respectively. To capture sequential patterns in the data, the model architecture, often employing LSTM layers, is devised. After training, the model predicts future stock prices. A visual representation, plotting both actual and predicted prices, aids in assessing the model's efficacy. Any disparities between the two sets are scrutinized to refine the model further, enhancing its predictive prowess for Sun Pharma’s stock behaviour.

Plot for real vs Predicted for Microsoft using LSTM

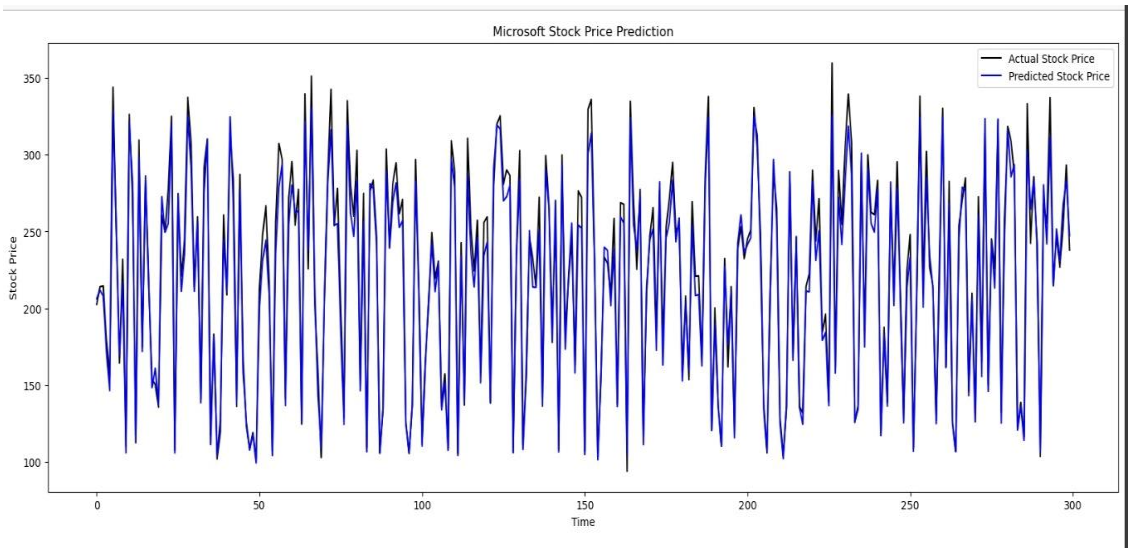


Figure 6.16

Microsoft historical stock data is systematically processed to train a predictive model. The data extraction phase involves gathering past stock prices of Microsoft. Subsequently, the data undergoes preprocessing to structure it into input sequences and target values. These datasets, referred to as x_train and y_train, respectively, are instrumental in training the model. Leveraging LSTM layers, the model is designed to capture temporal dependencies in the data. Post-training, the model generates forecasts for future stock prices. A graphical depiction, comparing actual and predicted prices, facilitates model evaluation. Discrepancies between these sets are examined to iteratively refine the model, optimizing its predictive capabilities Microsoft's stock movements.

CHAPTER-7

FUTURE SCOPE AND CONCLUSION

7.1. Future Scope

The future scope for LSTM models in stock prediction is promising and includes several avenues for improvement and innovation:

7.1.1. Enhanced Feature Engineering: Incorporating a wider range of features beyond historical prices, such as news sentiment analysis, company fundamentals, macroeconomic indicators, and alternative data sources like satellite imagery or social media activity, can provide richer input to LSTM models, potentially leading to better predictions.

7.1.2. Hybrid Models: Combining LSTM with other deep learning architectures or traditional machine learning techniques like convolutional neural networks (CNNs), attention mechanisms, or ensemble methods can leverage the strengths of different approaches, leading to more robust and accurate predictions.

7.1.3. Interpretable Models: Developing LSTM models that provide explanations or insights into their predictions can enhance trust and understanding among users and stakeholders. Techniques like attention mechanisms or layer-wise relevance propagation (LRP) can help to highlight the most influential features or time periods in the prediction process.

7.1.4. Uncertainty Estimation: Integrating uncertainty estimation methods into LSTM models can provide confidence intervals or probability distributions for predicted stock prices, enabling investors to make more informed decisions and manage risk effectively.

7.1.5. Real-time Prediction: Improving the efficiency and speed of LSTM models to enable real-time prediction of stock prices, potentially leveraging techniques like model compression, quantization, or hardware acceleration, can enable traders to capitalize on rapid market changes and opportunities.

7.1.6. Adversarial Robustness: Enhancing the robustness of LSTM models against adversarial attacks and data perturbations can mitigate the risk of malicious manipulation or exploitation in real-world applications, ensuring the reliability and integrity of stock predictions.

7.1.7. Domain Adaptation: Adapting LSTM models to different financial markets, asset classes, or trading strategies through transfer learning or domain adaptation techniques can broaden their applicability and scalability across diverse scenarios and environments.

By addressing these challenges and opportunities, LSTM models for stock prediction can continue to advance and evolve, providing valuable insights and decision support for investors, traders, and financial institutions in the dynamic and complex world of stock markets.

7.2. CONCLUSION

A challenging but fascinating field, stock price prediction using artificial intelligence (AI) combines financial information analysis with state-of-the-art quantifiable and computational techniques to estimate future stock costs. As we've shown throughout this discussion, the process involves obtaining accurate stock cost data, analyzing and preprocessing it, creating AI models, evaluating how well the models are working, and finally projecting future stock cost estimates. Finally, we will delve more into the significance, challenges, and prospects for stock cost prediction with machine learning.

First and foremost, there is no greater relevance than an accurate stock value projection. Such expectations are necessary for financial backers, dealers, monetary foundations, and even policymakers to make well-informed decisions about buying, selling, or holding stocks. A few advantages over traditional methods are provided by ML-based stock cost expectation models, such as the ability to quickly process large amounts of data, identify intricate examples, and adapt to shifting economic conditions. Examiners can unearth hidden experiences and produce more accurate forecasts by using machine learning (ML) calculations. This could lead to higher venture rewards and less financial risks.

However, despite its dedication, stock cost forecasting with machine learning has some challenges and limitations. The inherent unconventionality and instability of the financial business sectors is one of the main challenges. Numerous factors influence stock prices, such as macroeconomic indicators, global events, market sentiment, and, shockingly, unexpected events like pandemics or catastrophic disasters. ML models may struggle to accurately capture and record for this plethora of factors, leading to inaccurate expectations.

Information accessibility and quality is a further test. Verifiable stock value data is often insufficient, controversial, or subject to biases, all of which might affect how ML models display. Furthermore, the financial industry is dynamic and ever-evolving, meaning that machine learning models must be updated and retrained frequently to maintain their accuracy. Furthermore, overfitting—a phenomenon in which a model excels at gathering data but fails to add up to hidden data—represents a significant risk in stock cost prediction, necessitating strict approval and regularization procedures.

Furthermore, there are concerns about how interpretable ML models are for stock value anticipation. Though ML computations such as neural networks and information-gathering techniques can achieve impressive predictive accuracy, they often function as "hidden variables," making it challenging to decipher the logic underlying their predictions. This lack of interpretability has the potential to destroy confidence in the models and undermine their acceptance, especially in highly focused endeavours such as finance.

Despite these challenges, significant improvement in algorithmic techniques, computing power, and information accessibility has led to significant advancements in stock cost forecasting through machine learning. To improve the accuracy and interpretability of expectations, analysts and specialists are looking at innovative approaches including profound learning, assist learning, and feeling examination. New experiences and protocols are also being driven in this field by interdisciplinary collaborations involving experts in software engineering, information science, and finance.

In the future, stock cost forecasting with machine learning has enormous promise. We may anticipate that as innovation continues to accelerate, machine learning models will become more sophisticated, potent, and flexible, capable of handling ever-more-complex data and market components. Additionally, the integration of optional information sources such as satellite symbolism, internet entertainment, and IoT (Web of Things) promises to improve current models and provide additional insights on consumer behaviour and financial backing patterns.

To sum up, machine learning-based stock price prediction is an intriguing nexus of technology and finance that gives several chances for value generation and innovation. The unwavering pursuit of quality in data analysis, algorithm creation, and model interpretation will propel further advancement in this discipline even as obstacles remain. Financial institutions and investors may navigate the complex and volatile world of financial markets with greater agility by utilizing machine learning (ML). This will ultimately improve stakeholder outcomes and decision-making.

REFERENCES

[1] “Stock price prediction using LSTM, RNN and CNN-sliding window model - IEEE Conference Publication.” https://ieeexplore.ieee.org/document/8126078 (accessed Dec. 27, 2019).

[2] J. Jagwani, M. Gupta, H. Sachdeva, and A. Singhal, “Stock Price Forecasting Using Data from Yahoo Finance and Analysing Seasonal and Nonseasonal Trend,” in 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, Jun. 2018, pp. 462–467, doi: 10.1109/ICCONS.2018.8663035.

[3] I. Parmar et al., “Stock Dec. 2018, pp. 574–576, doi: 10.1109/ICSCCC.2018.8703332.

[4] Y. Lei, K. Zhou, and Y. Liu, “Multi-Category Events Driven Stock Price Trends Prediction,” in 2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS), Nanjing, China, Nov. 2018, pp. 497–501, doi: 10.1109/CCIS.2018.8691392.

[5] B. Jeevan, E. Naresh, B. P. V. kumar, and P. Kambli, “Share Price Prediction using Machine Learning Technique,” in 2018 3rd International Conference on Circuits, Control, Communication and Computing (I4C), Bangalore, India, Oct. 2018, pp. 1–4, doi: 10.1109/CIMCA.2018.8739647.

[6] M. Usmani, S. H. Adil, K. Raza, and S. S. A. Ali, “Stock market prediction using machine learning techniques,” in 2016 3rd International Conference on computer and Information Sciences (ICCOINS), 2016, pp. 322–327.

[7] J. Du, Q. Liu, K. Chen, and J. Wang, “Forecasting stock prices in two ways based on LSTM neural network,” in 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Mar. 2019, pp. 1083–1086, doi: 10.1109/ITNEC.2019.8729026.

[8] S. E. Gao, B. S. Lin, and C.-M. Wang, “Share Price Trend Prediction Using CRNN with LSTM Structure,” in 2018 International Symposium on Computer, Consumer and Control (IS3C), Dec. 2018, pp. 10–13, doi: 10.1109/IS3C.2018.00012.

[9] T. Gao, Y. Chai, and Y. Liu, “Applying long short term memory neural networks for predicting stock closing price,” in 2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, Nov. 2017, pp. 575–578, doi: 10.1109/ICSESS.2017.8342981. Stock Price Prediction Using Machine Learning 18CP808 32.

[10] R. Y. Nivetha and C. Dhaya, “Developing a Prediction Model for Stock Analysis,” in 2017 International Conference on Technical Advancements in

[11] Computers and Communications (ICTACC), Melmaurvathur, India, Apr. 2017, pp. 1–3, doi: 10.1109/ICTACC.2017.11.
[12] Z. Yeze and W. Yiying, “Stock Price Prediction Based on Information Entropy and Artificial Neural Network,” in 2019 5th International Conference on Information Management (ICIM), Cambridge, United Kingdom, Mar. 2019, pp. 248–251, doi: 10.1109/INFOMAN.2019.8714662.
[13] “Basic LSTM Unit Transfer Function Diagram from [10] | Download Scientific Diagram.” https://www.researchgate.net/figure/Basic-LSTM-Unit-Transfer-FunctionDiagram-from-10_fig8_308804546 (accessed Jun. 03, 2020).
[14] “Fig. 1: An example of CNN architecture.,” ResearchGate. https://www.researchgate.net/figure/An-example-of-CNN-architecture_fig1_320748406 (accessed Jun. 03, 2020).
[15] Y. Kim, “Convolutional Neural Networks for Sentence Classification,” in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, Oct. 2014, pp. 1746–1751, doi: 10.3115/v1/D14-1181.
[16] B. Athiwaratkun and K. Kang, “Feature Representation in Convolutional Neural Networks,” ArXiv150702313 Cs, Jul. 2015, Accessed: Apr. 14, 2020. [Online]. Available: http://arxiv.org/abs/1507.02313
[17] F. A. Gers, D. Eck, and J. Schmidhuber, “Applying LSTM to Time Series Predictable through Time-Window Approaches,” in Artificial Neural Networks — ICANN 2001, Berlin, Heidelberg, 2001, pp. 669–676, doi: 10.1007/3-540-44668-0_93.
[18] J. Han and C. Moraga, “The Influence of the Sigmoid Function Parameters on the Speed of Backpropagation Learning,” in Proceedings of the International Workshop on Artificial Neural Networks: From Natural to Artificial Neural Computation, Berlin, Heidelberg, Jun. 1995, pp. 195–201, Accessed: Apr. 13, 2020. [Online].
[19] “NSE Listed 1000+ Companies’ Historical Data.” https://kaggle.com/abhishekyana/nselisted-1384-companies-data (accessed Jul. 07, 2020).
[20] A. Tipirisetty, “Stock Price Prediction using Deep Learning,” p. 60.
[21] “Python Numpy Tutorial | Learn Numpy Arrays With Examples,” Edureka, oct. 14, 2023. https://www.edureka.co/blog/python-numpy-tutorial/ (accessed oct. 21, 2023).
[22] “Scikit-learn Tutorial: Machine Learning in Python – Dataquest.” https://www.dataquest.io/blog/sci-kit-learn-tutorial/ (accessed sep. 21, 2023).
[23] J. Brownlee, “Introduction to the Python Deep Learning Library TensorFlow,” Machine Learning Mastery, May 04, 2016. https://machinelearningmastery.com/introductionpython-deep-learning-library-tensorflow/ (accessed Apr. 21, 2020).
[24] “Home - Keras Documentation.” https://keras.io/ (accessed Apr. 21, 2020).
[25] “Project Jupyter.” https://www.jupyter.org (accessed Oct 19, 2023).

[26] K. Team, “Keras documentation: Search Keras documentation.” https://keras.io/search.html?query=lstm (accessed Nov, 07, 2023).
[27] K. Joshi, kartik-joshi/Stock-prediction. 2020.
[28] K. Team, “Keras documentation: The Sequential model.” https://keras.io/guides/sequential_model/ (accessed Jul. 07, 2020). Market Prediction Using Machine Learning,” in 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC).
[29] https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/ (accessed Mar, 19, 2024)