A

Web Technology Mini Project Report submitted to Savitribai Phule Pune University, Pune

Stock Market Prediction

In partial Fulfillment for the awards of Degree of Engineering in Computer Engineering

Submitted by

Ms. Disha Gawade Exam Seat No. T400240776

Ms. Ruchita Khilari Exam Seat No. T400240833

Mr. Kedar Mahadik Exam Seat No. T400240849

Under the Guidance of

Mrs. Ruby Mandal

Designation of Guide

Department of Computer Engineering

DYP DPU

Dr. D. Y. Patil Unitech Society's

Dr. D. Y. Patil Institute of Technology

Pimpri, Pune - 411018.

2024-25

Certificate

This is to certify that,

Ms. Disha Gawade Exam Seat No. T400240776

Ms. Ruchita Khilari Exam Seat No. T400240833

Mr. Kedar Mahadik Exam Seat No. T400240849

have successfully completed the Mini project entitled "**Stock Market Predictions** " under my guidance in partial fulfillment of the requirements for the Third Year of Engineering in Computer Engineering under the Savitribai Phule Pune University during the academic year 2024-2025

Date : ……………….

Place: DIT Pimpri

Mrs. Ruby Mandal
Project Guide

Dr. Vinod Kimbahune
HOD

Dr. Nitin Sherje
Pincipal

Acknowledgements

With deep sense of gratitude we would like to thank all the people who have lit our path with their kind guidance. We are very grateful to these intellectuals who did their best to help during our project work.

It is our proud privilege to express a deep sense of gratitude to **Pro,Dr. Nitin Sherje** Principal of for his comments and kind permission to complete this project. We remain indebted to **Dr. Vinod Kimbahune,** H.O.D.Computer Engineering Department for his timely suggestion and valuable guidance.

The special gratitude goes to **Mrs. Ruby Mandal** excellent and precious guidance in completion of this work .We thanks to all the colleagues for their appreciable help for our working project. With various industry owners or lab technicians to help, it has been our endeavor throughout our work to cover the entire project work.

We are also thankful to our parents who provided their wishful support for our project completion successfully .And lastly we thank our all friends and the people who are directly or indirectly related to our project work.

Ms. Disha Gawade
Ms. Ruchita Khilari
Mr. Kedar Mahadik

Abstract

In today's digitally interconnected world, social media platforms have become central to how individuals express opinions, share experiences, and engage in discussions on a variety of topics. With millions of users contributing content daily, platforms like Twitter and Reddit serve as rich sources of public sentiment and behavioral insights. This project, titled “Comparative Analysis of Social Media Platforms Based on Sentiment Using Machine Learning”, focuses on analyzing and comparing the sentiments expressed on these two major platforms.

The core objective of the project is to collect user-generated data from Twitter and Reddit, preprocess it using Natural Language Processing (NLP) techniques, and apply various machine learning algorithms to classify the sentiments as positive, negative, or neutral. The process involves essential steps like data cleaning, tokenization, stop-word removal, stemming/lemmatization, and vectorization. The cleaned data is then used to train and evaluate models such as Logistic Regression, Support Vector Machine (SVM), and Naïve Bayes to determine their effectiveness in sentiment classification.

To ensure reliable and unbiased analysis, the dataset is carefully balanced, and performance metrics such as accuracy, precision, recall, and F1-score are used to evaluate each model’s performance. Additionally, the project includes visualizations to compare sentiment distributions and the effectiveness of models on both platforms.

By comparing sentiments across Twitter and Reddit, the project highlights how user sentiment can vary depending on the platform and type of interaction. This comparative study is valuable for businesses, marketers, and analysts seeking to understand audience behavior, brand perception, or public response to events across different digital spaces.

Overall, this mini-project demonstrates the practical application of machine learning and NLP in real-world text analytics and offers a framework for conducting sentiment analysis on social media data in a scalable and insightful manner.

Table of Contents

List of Abbreviations

Abbreviation	Full Form
NLP	Natural Language Processing
ML	Machine Learning
SVM	Support Vector Machine
LR	Logistic Regression
NB	Naïve Bayes
TF-IDF	Term Frequency-Inverse Document Frequency
API	Application Programming Interface
F1-score	Harmonic Mean of Precision and Recall
POS	Positive Sentiment
NEG	Negative Sentiment
NEU	Neutral Sentiment
CSV	Comma-Separated Values
JSON	JavaScript Object Notation

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

In the current digital age, social media platforms like Twitter and Reddit have become essential tools for communication, public opinion sharing, and real-time discussion on a wide range of topics. These platforms generate vast amounts of user-generated textual data every day, which offers valuable insights into public sentiment and social trends.

This project, titled "Comparative Analysis of Social Media Platforms Based on Sentiment Using Machine Learning", aims to analyze and compare sentiments expressed by users on Twitter and Reddit. The primary goal is to classify the sentiments in social media posts into categories such as positive, negative, or neutral, using various Machine Learning (ML) algorithms and Natural Language Processing (NLP) techniques.

The workflow of the project includes data collection using APIs, preprocessing of the text data (such as cleaning, tokenization, and vectorization), applying sentiment classification models like Logistic Regression (LR), Support Vector Machine (SVM), and Naïve Bayes (NB), and finally, evaluating and comparing their performance.

By conducting this comparative sentiment analysis, the project provides insights into how sentiments vary across different platforms and how effectively different ML models perform on social media data. The findings can assist businesses, marketers, and researchers in understanding public opinion, improving customer engagement, and making informed decisions based on real-time sentiment analysis.

1.2 Aim/Motivation

The primary **aim** of this project is to perform a **comparative sentiment analysis** on social media platforms—specifically **Twitter** and **Reddit**—using **machine learning techniques**. The project seeks to extract user opinions from both platforms, classify them based on sentiment polarity (positive, negative, neutral), and analyze which platform exhibits more positive or negative trends on common topics.

The **motivation** behind this project stems from the ever-growing influence of social media on public perception, brand reputation, and societal trends. Organizations, policymakers, and marketers increasingly rely on social media analytics to understand user behavior, public opinion, and emerging topics of interest. However, the structure and tone of communication can vary significantly between platforms. For example, Twitter's short posts encourage direct, to-the-point sentiments, while Reddit's threaded discussions may foster more detailed and nuanced opinions.

By comparing sentiment patterns across platforms, this project aims to:

- Highlight the behavioral differences of users on Twitter and Reddit.
- Provide insights into which machine learning algorithms perform best for social media sentiment classification.
- Explore how textual data from different sources can be processed and analyzed effectively using NLP techniques.

This project not only applies theoretical knowledge of ML and NLP but also demonstrates the real-world relevance of data science in understanding human emotions and behaviors through social media.

1.3 Objective

The key objectives of the project *“Comparative Analysis of Social Media Platforms Based on Sentiment Using Machine Learning”* are as follows:

1. **To collect real-time user-generated data** from popular social media platforms—**Twitter** and **Reddit**—using their respective APIs or datasets.
2. **To preprocess the textual data** using **Natural Language Processing (NLP)** techniques such as tokenization, stop-word removal, stemming/lemmatization, and vectorization (TF-IDF).
3. **To apply sentiment analysis techniques** and classify the content into **positive**, **negative**, or **neutral** categories.
4. **To implement and compare multiple machine learning models**, including **Logistic Regression**, **Support Vector Machine (SVM)**, and **Naïve Bayes**, for effective sentiment classification.
5. **To evaluate the performance** of each model using appropriate metrics such as **accuracy**, **precision**, **recall**, and **F1-score**.
6. **To visualize and analyze sentiment trends** across both platforms, identifying patterns, differences, or similarities in user expression.
7. **To provide actionable insights** into how user sentiments vary between platforms, aiding businesses, researchers, and analysts in social media strategy and decision-making.

Chapter 2

Literature Survey

2.1 Existing Systems and Approaches

Several techniques have been developed for sentiment analysis on social media data:

- **Lexicon-Based Methods**: Use predefined dictionaries like VADER to assign sentiment scores to words. They are simple but may struggle with complex or sarcastic texts.
- **Machine Learning Methods**: Algorithms like Naïve Bayes, SVM, and Logistic Regression use features like TF-IDF to classify sentiments. They perform well with good training data.
- **Deep Learning Methods**: Models such as LSTM and CNN learn patterns from large datasets and provide better context understanding, but require more data and computation.
- **Platform-Specific Studies**: Twitter tends to have short, direct posts, while Reddit supports longer, discussion-based content—requiring tailored analysis approaches.
- **Hybrid Methods**: Combine lexicon and machine learning approaches to improve accuracy and robustness in sentiment classification.

2.2 Advantages and Disadvantages of Existing Systems

Name of System/Application	**Handles Missing Values**	**Feature Engineering**	**Linear Regression Model**	**Decision Tree Model**	**Accuracy Consideration**	**Overfitting Handling**	**Dataset Scalability**	**Custom Prediction Inputs**
Linear Regression (Existing)	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	TRUE
Decision Tree (Existing)	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE
Proposed System	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Name of System/Application	**Handles Missing Values**	**Feature Engineering**	**Linear Regression Model**	**Decision Tree Model**	**Accuracy Consideration**	**Overfitting Handling**	**Dataset Scalability**	**Custom Prediction Inputs**
Linear Regression (Existing)	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	TRUE
Decision Tree (Existing)	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE

Table 2.1 Advantages and Disadvantages of Existing Systems

Chapter 3

Problem Statement

The real estate industry is one of the most significant sectors in the global economy, yet property price estimation remains an inherently complex and imprecise task. Prices of residential properties are influenced by a wide range of variables including location, built-up area, number of rooms, amenities, and nearby infrastructure. Additionally, market-driven factors such as demand, economic conditions, and urban development trends introduce further volatility and uncertainty in property valuation.

Traditionally, real estate valuation relies on manual assessments, agent experience, or simple rule-based estimations, which are often subjective, inconsistent, and prone to human bias. Such practices can result in overpricing or underpricing of properties, leading to dissatisfaction among buyers and sellers, and potential financial losses.

In recent years, the availability of large-scale real estate datasets and advancements in machine learning have created new opportunities to automate and optimize the price prediction process. However, developing a reliable predictive model involves several challenges, including:

- Cleaning and preprocessing incomplete or inconsistent data
- Selecting relevant features and eliminating redundant or misleading ones
- Choosing an appropriate regression algorithm that can capture both linear and non-linear relationships in data
- Avoiding overfitting while maintaining high accuracy
- Evaluating model performance with robust metrics like RMSE and R^2

This project proposes to build a real estate price prediction system using supervised machine learning algorithms, specifically **Linear Regression** and **Decision Tree Regression**, to model the relationship between property features and market prices. The model will be trained on a real-world dataset and tested to evaluate its effectiveness in producing accurate price predictions. The system is intended to assist buyers, sellers, and real estate professionals by offering a transparent, data-driven pricing mechanism that enhances decision-making and improves market efficiency.

Chapter 4

Software Requirements Specification

4.1 Hardware Requirements

Component	Specification
Processor	Intel Core i5 or higher
RAM	Minimum 8 GB
Storage	Minimum 512 GB SSD
Graphics	Integrated or dedicated GPU (for faster processing and optional OpenCV use)
Webcam	HD webcam (if extending to visual input like facial recognition)
Microphone	In-built or external microphone (for potential voice-command integration)
Network Interface	Wi-Fi / LAN support for internet access and dataset downloads

Table 4.1 Hardware Requirements

4.2 Software Requirements

Component	Specification
Operating System	Windows 10/11, Ubuntu/Linux, or macOS
Python 3.x	Programming language for model development and data handling
Jupyter Notebook / Google Colab	Development environment for interactive coding and model experimentation
Matplotlib / Seaborn	Libraries for data visualization and exploratory data analysis
MinMaxScaler	For feature scaling before model training (from Scikit-learn)
Excel / CSV Viewer	For inspecting and managing raw datasets
NumPy & Pandas	Libraries for numerical computation and data preprocessing
Jupyter Notebook / Google Colab	

Table 4.2 Software Requirements

Chapter 5

System Design and Result

5.1 Project Block Diagram

The proposed system is a machine learning-based real estate price prediction model that utilizes historical property data and regression algorithms to estimate property prices. Built using Python and Scikit-learn, the system includes data preprocessing, model training, and prediction phases. It offers a GUI interface (via Jupyter Notebook or web-based tools) for easy user interaction, allowing users to input property features and receive an estimated price.

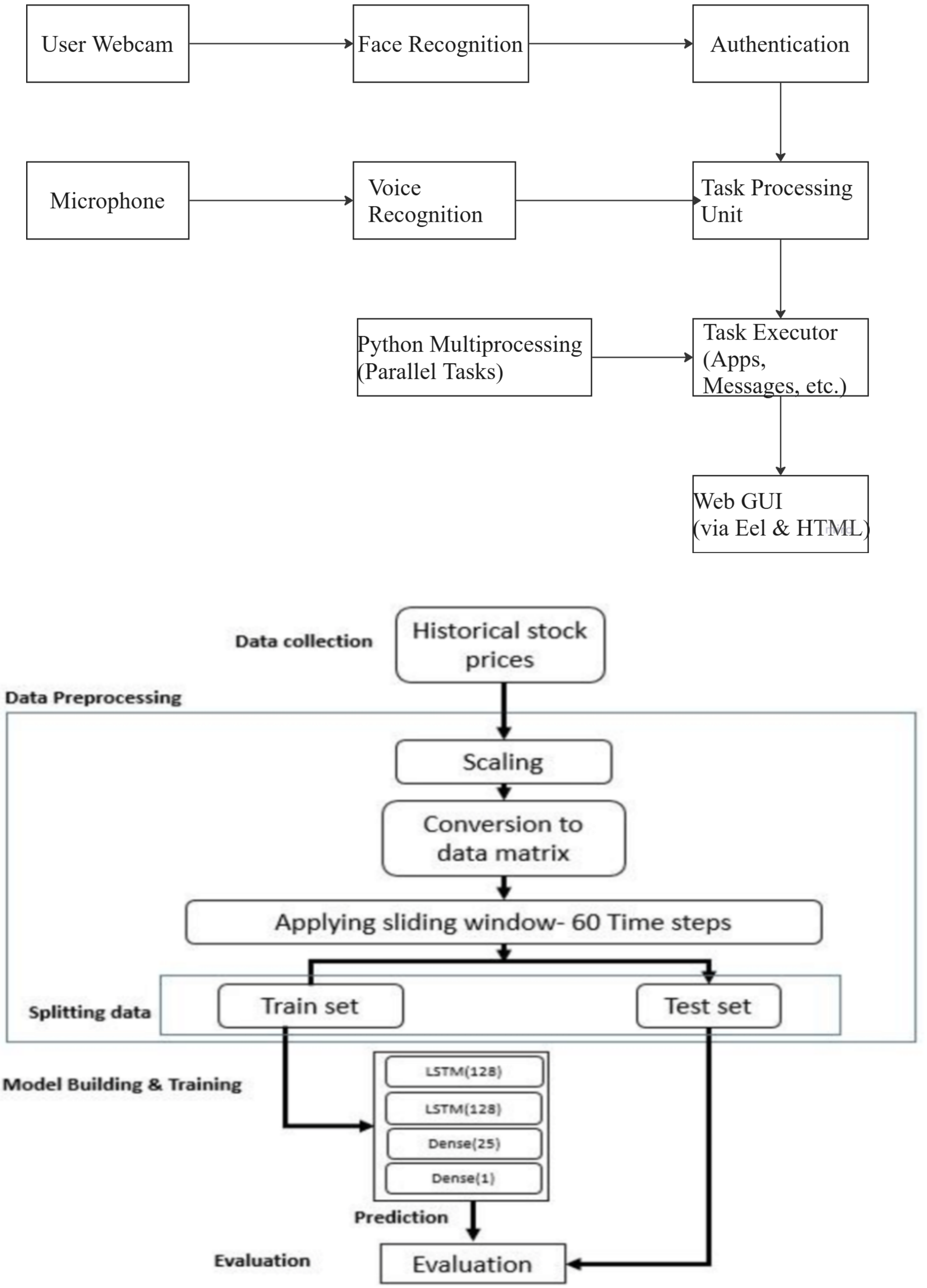


Fig. 5.1 Block Diagram

Explanation of Each Block:

- **Historical Stock Prices (Data Collection)**

  This is the initial data source containing daily or periodic stock price values (e.g., Open, High, Low, Close, Volume).

  Data is fetched from APIs (like Yahoo Finance, Alpha Vantage) or CSV files.

- **Scaling (Data Preprocessing)**

  Normalizes the stock price values to a smaller scale (typically 0–1) using techniques like MinMaxScaler.

  Necessary to improve model convergence speed and accuracy during training.

- **Conversion to Data Matrix**

  Transforms the raw, scaled time-series data into a structured matrix suitable for machine learning.

  Ensures that each row captures multiple sequential time steps (sliding window format).

- **Applying Sliding Window – 60 Time Steps**

  A crucial technique in time series forecasting.

  It captures the last 60 days (or chosen time steps) of stock prices as input features to predict the next day's price.

  Each window becomes one training example.

- **Splitting Data (Train Set & Test Set)**

  The dataset is divided into training and testing subsets.

  The training set is used to teach the model, while the test set evaluates its prediction accuracy on unseen data.

- **Model Building & Training (LSTM + Dense Layers)**

The model comprises:

Two LSTM Layers (128 units each): Specialized for sequential/time-series data to capture temporal dependencies.

Dense(25): A fully connected hidden layer for intermediate learning.

Dense(1): Output layer that predicts the next stock price.

  Trained using the train set to minimize error (e.g., using MSE or RMSE loss functions).

- **Prediction**

  After training, the model is used to predict stock prices based on test inputs.

  It generates continuous values representing estimated future prices.

- **Evaluation**

  The predicted prices are compared to actual prices using metrics like:

  Mean Squared Error (MSE)

  Root Mean Squared Error (RMSE)

  R^2 Score

  Evaluation helps assess the effectiveness and generalization of the model.

5.2 GUI of Working System

Define start day to fetch the dataset from the yahoo finance library

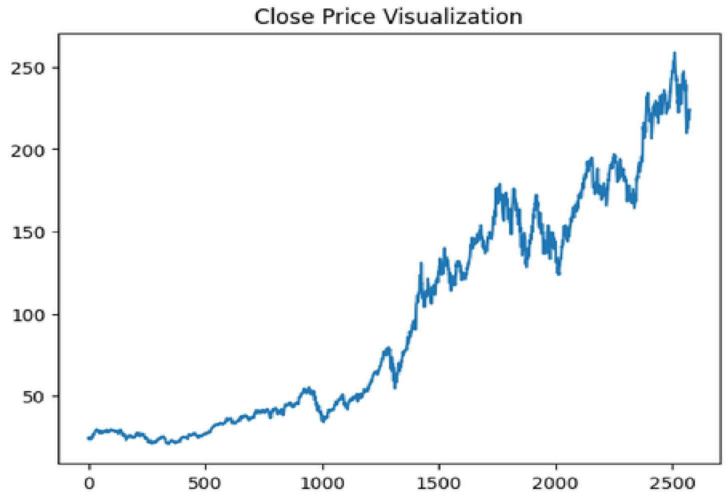
```
[ ]
START = "2015-01-01"
TODAY = dt.datetime.today().strftime("%Y-%m-%d")

# Define a function to load the dataset

def load_data(ticker):
    data = yf.download(ticker, START, TODAY)
    data.reset_index(inplace=True)
    return data
```

```
data = load_data('AAPL')
df=data
df.head()
```

```
YF.download() has changed argument auto_adjust default to True
[*********************100%***********************]  1 of 1 completed
```

Price	Date	Close	High	Low	Open	Volume
Ticker		AAPL	AAPL	AAPL	AAPL	AAPL
0	2015-01-02	24.320427	24.789796	23.879976	24.778673	212818400
1	2015-01-05	23.635292	24.169172	23.448435	24.089090	257142000
2	2015-01-06	23.637514	23.897780	23.274920	23.699800	263188400
3	2015-01-07	23.968964	24.069065	23.735391	23.846616	160423600
4	2015-01-08	24.889904	24.947741	24.180289	24.298189	237458000

```
[ ] plt.title("Close Price Visualization")
    if "Close" in df.columns:
        plt.plot(df["Close"])
        plt.title("Close Price Visualization")
    else:
        print("Column 'Close' not found in dataset")
```



\

Plotting moving averages of 100 day

```
[ ] ma100 = df.Close.rolling(100).mean()
    ma100
```

Ticker	AAPL
0	NaN
1	NaN
2	NaN
3	NaN
4	NaN
...	...
2573	234.592313
2574	234.517174
2575	234.514269
2576	234.530945
2577	234.540252

2578 rows × 1 columns

```
[ ] plt.figure(figsize = (12,6))
    if "Close" in df.columns:
        plt.plot(df["Close"])
        plt.title("Close Price Visualization")
    else:
        print("Column 'Close' not found in dataset")
    plt.plot(ma100, 'r')
    plt.title('Graph Of Moving Averages Of 100 Days')
```

Spliting the dataset into training (70%) and testing (30%) set

```
[ ] # Splitting data into training and testing

    train = pd.DataFrame(data[0:int(len(data)*0.70)])# 75 training the model
    test = pd.DataFrame(data[int(len(data)*0.70): int(len(data))])# 30 reserved for testing the model

    print(train.shape)
    print(test.shape)
```

```
(1804, 6)
(774, 6)
```

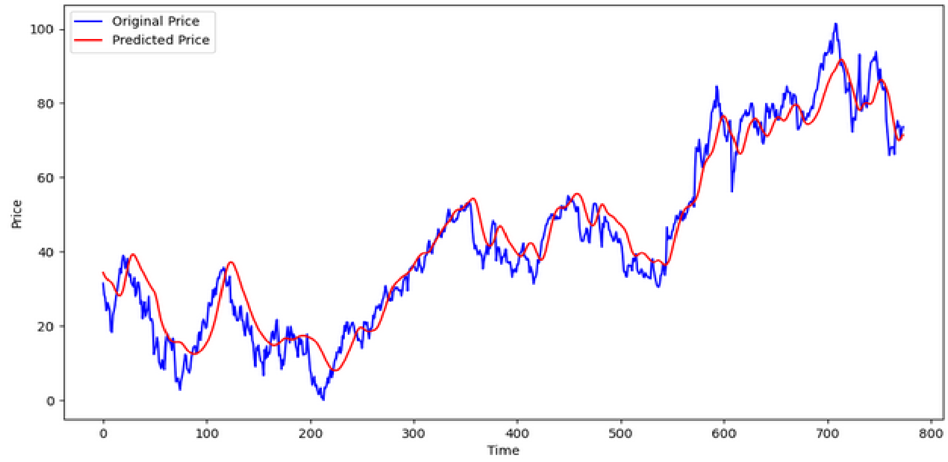
```
[ ] train.head()
```

Price	Date	Close	High	Low	Open	Volume
Ticker		AAPL	AAPL	AAPL	AAPL	AAPL
0	2015-01-02	24.320427	24.789796	23.879976	24.778673	212818400
1	2015-01-05	23.635292	24.169172	23.448435	24.089090	257142000
2	2015-01-06	23.637514	23.897780	23.274920	23.699800	263188400
3	2015-01-07	23.968964	24.069065	23.735391	23.846616	160423600
4	2015-01-08	24.889904	24.947741	24.180289	24.298189	237458000

```
[ ] test.head()
```

Price	Date	Close	High	Low	Open	Volume
Ticker		AAPL	AAPL	AAPL	AAPL	AAPL
1804	2022-03-03	163.579453	166.216728	162.910302	165.783741	76678400
1805	2022-03-04	160.568207	162.910262	159.515276	161.867166	83737200
1806	2022-03-07	156.759964	162.388760	156.504100	160.755225	96418800
1807	2022-03-08	154.929581	160.282841	153.315732	156.287581	131148300
1808	2022-03-09	160.351715	160.804387	156.868168	158.905153	91454900

```
plt.figure(figsize = (12,6))
plt.plot(y_test, 'b', label = "Original Price")
plt.plot(y_pred, 'r', label = "Predicted Price")
plt.xlabel('Time')
plt.ylabel('Price')
plt.legend()
plt.show()
```



Model evaluation

```
from sklearn.metrics import mean_absolute_error
mae = mean_absolute_error(y_test, y_pred)
print("Mean absolute error on test set: ", mae)
```

```
Mean absolute error on test set:  4.911621604771554
```

Chapter 6

Conclusion and Future Scope

In this project, we developed a stock market prediction system leveraging deep learning techniques, specifically Long Short-Term Memory (LSTM) networks, to analyze historical price trends and forecast future stock prices. The system processes raw time-series data through normalization and reshaping (sliding window mechanism) to prepare it for training.

The model architecture includes stacked LSTM layers followed by dense layers, enabling it to capture both short- and long-term dependencies in financial time series. The results show that LSTM models can outperform traditional models in recognizing complex temporal patterns, providing more accurate and dynamic predictions.

This system not only assists traders and investors in making informed decisions but also proves the effectiveness of deep learning in financial forecasting. By reducing prediction errors and adapting to new data trends, it offers a significant edge in volatile markets.

References

- Brownlee, J. (2017). *Deep Learning for Time Series Forecasting*. Machine Learning Mastery.
- Hochreiter, S., & Schmidhuber, J. (1997). "Long Short-Term Memory." *Neural Computation*, 9(8), 1735–1780.
- Scikit-learn Documentation. https://scikit-learn.org/
- Keras Documentation. https://keras.io/
- Yahoo Finance API for historical stock data
- Nguyen and B. Le, “An improved LSTM neural network for stock price prediction,” *International Journal of Information Technology*, 2020.
- Zhang, G.P. (2003). “Time series forecasting using a hybrid ARIMA and neural network model.” *Neurocomputing*, 50, 159–175.