Project Report
on

STOCK VISUALIZATION AND FORECASTING

by

V S S VEERENDR KUMAR	*422179*
KARTHIKEYA MADHAVAN	*422134*
D ISSAC	*422135*

Under the guidance of

Mr. D. Prasad

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY, ANDHRA PRADESH
TADEPALLIGUDEM-534101, INDIA

MAY-2024

Department of Computer Science and Engineering

NATIONAL INSTITUTE OF TECHNOLOGY, ANDHRA PRADESH

Certificate

It is certified that the work contained in the thesis titled "STOCK VISUALIZATION AND FORECASTING" by "V S S Veerendra Kumar, bearing Roll No: 422179", "Karthikeya Madhavan, bearing Roll No: 422134" and "D Issac, bearing Roll No: 422135" has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Mr. D. Prasad
Computer Science and Engineering
N.I.T. Andhra Pradesh
May 2024

Place: Tadepalligudam
Date:

May 4, 2024

Table of Contents

Abstract

This study delves into the crucial realm of stock visualization and forecasting, recognizing its paramount significance in contemporary financial landscapes. With the escalating complexity of global markets and the abundance of data available, the need for robust analytical tools has become more pressing than ever. Through the introduction of a pioneering annotated dataset and the utilization of a Long Short-Term Memory (LSTM) network model combined with Min-Max normalization, this research endeavors to push the boundaries of stock market analysis.

In the current era, characterized by rapid technological advancements and the ubiquity of information, stock visualization and forecasting serve as indispensable pillars for informed decision-making in investment strategies. By harnessing the power of advanced machine learning techniques, this study endeavours to empower investors and analysts with accurate predictions and actionable insights. Moreover, the development of a user-friendly web interface further democratizes access to predicted data, bridging the gap between complex analytical methodologies and practical application.

The fusion of sophisticated predictive modeling with intuitive visualization tools not only augments the efficiency of stock market analysis but also facilitates more informed decision-making processes. In an environment where timeliness and accuracy are paramount, the proposed approach holds the potential to revolutionize how stakeholders navigate the intricacies of financial markets. Ultimately, by offering valuable insights and actionable predictions, this research aims to empower individuals and organizations to navigate the volatile terrain of the stock market with confidence and agility.

Chapter 1

Introduction

In today's dynamic financial landscape, where the interplay of geopolitical shifts and technological advancements exerts profound influence, the necessity for accurate stock visualization and forecasting has reached unprecedented levels. This study delves into the intricate realm of stock market dynamics, recognizing the pivotal role played by robust predictive models in guiding investment strategies and facilitating informed decision-making processes amidst the complexities of global markets.

Stock visualization and forecasting serve as essential compasses in the sea of financial uncertainties, empowering investors and analysts to navigate volatile trends and seize lucrative opportunities. As data availability burgeons and market intricacies deepen, the demand for sophisticated analytical methodologies becomes ever more pressing. Leveraging state-of-the-art machine learning algorithms and pioneering data visualization techniques, this study endeavors to equip stakeholders with actionable insights and precise predictions, thereby bolstering their confidence and adaptability in navigating the tumultuous waters of the stock market.

Central to this endeavor is the development of an innovative annotated dataset and the application of cutting-edge techniques, including Long Short-Term Memory (LSTM) networks and Min-Max normalization. These methodologies represent the vanguard of predictive modeling, enabling researchers to glean invaluable patterns from extensive historical stock data. By seamlessly integrating these advanced techniques with intuitive visualization tools, this study aims to redefine the boundaries of stock market analysis, furnishing investors and analysts with a comprehensive framework for deciphering market dynamics and making judicious decisions. Through the fusion of sophisticated predictive modeling with user-friendly interfaces, the ultimate goal is to democratize access to predictive data and bridge the chasm between intricate analytical methodologies and pragmatic application, thus empowering a broader spectrum of market participants to navigate the complexities of the stock market with precision

Chapter 2

Literature Survey

Quantitative finance has witnessed a paradigm shift with the advent of machine learning algorithms, which offer sophisticated methods for forecasting asset prices, managing portfolios, and optimizing investment strategies. Traditional linear models have been supplemented by complex algorithms like neural networks, gradient boosted regression trees, and support vector machines, enabling the detection of non-linear patterns and intricate relationships in financial data.

MOGHARa and HAMICHEb's study leverages LSTM, a specialized form of RNN capable of retaining long-term dependencies in sequential data, making it particularly suitable for time series forecasting. The authors utilize a dataset comprising daily opening prices of two stocks, GOOGL and NKE, extracted from Yahoo Finance. They partition the data into training and testing sets, employing mean squared error as the optimization metric during model training. Experiments are conducted with varying numbers of epochs to assess the impact on predictive performance.

The findings reveal that the choice of epochs significantly influences the model's ability to accurately forecast stock prices. Training with fewer data points but more epochs demonstrate improvements in forecasting accuracy. However, challenges arise from shifts in data nature, such as increased volatility or changes in asset behaviour, which may impact the model's performance. Despite these challenges, the LSTM-based approach shows promise in tracking the evolution of opening prices for both GOOGL and NKE assets.

In conclusion, MOGHARa and HAMICHEb's research underscores the potential of LSTM-based models in enhancing stock market predictions. The study contributes to the growing body of literature on the application of machine learning in quantitative finance, highlighting the importance of optimizing dataset selection and training epochs to maximize prediction accuracy. Future endeavours aim to refine model parameters and explore additional factors to further improve forecasting performance and adaptability to dynamic market conditions.

Chapter 3

Methodology

3.1 Problem Statement

Develop a web application aimed at visualizing and forecasting stock market data to assist investors in making informed decisions. In the ever-changing realm of stock markets, investors grapple with the daunting task of sifting through copious historical data, spotting trends, and making accurate predictions about future stock prices. Existing tools often lack the sophistication and user-friendliness required for such analysis, leaving investors without adequate support.

3.2 Model Architecture

3.2.1 Data Loading and Preprocessing

The initial phase involves loading historical stock price data using the Yahoo Finance API (yfinance). This data is essential for training and evaluating the LSTM model. Upon loading the data, preprocessing steps are applied to prepare it for model training. Specifically, the data is scaled using the MinMaxScaler to normalize it between 0 and 1. Normalization ensures that all features have the same scale, which is crucial for neural network training.

3.2.2 Model Building

The model comprises multiple LSTM layers stacked on top of each other, followed by dropout layers to mitigate overfitting. The input shape of the model is determined based on the number of time steps and features in the data.

The LSTM architecture is defined as follows:

$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}C_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}C_{t-1} + b_f) \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}C_t + b_o) \\
\tilde{C}_t &= \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\
C_t &= f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \\
h_t &= o_t \odot \tanh(C_t)
\end{aligned}
$$

3.2.3 Model Training

The model training phase involves training the LSTM model using the prepared training data. Training data consists of sequences of historical stock prices, with each sequence used to predict the next closing price based on the past prediction_days days of data. The model is trained iteratively over multiple epochs using batch training, where the weights of the model are updated to minimize the mean squared error loss.

3.2.4 Model Evaluation

After training, the model's performance is evaluated using test data to assess its ability to make accurate predictions. Metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and accuracy percentage are computed to quantify the model's performance. Additionally, the actual and predicted stock prices are plotted for visual inspection to gain insights into the model's predictions.

$$\text{Accuracy Percentage} = \left(\frac{\text{Number of accurate predictions}}{\text{Total number of predictions}}\right) \times 100\%$$

3.2.5 Model Saving and Loading

Trained models are saved to disk for future use using the save_model function. This allows the model to be reused without the need for retraining, saving computational resources and time. Additionally, models can be loaded from disk using the load_model function when needed.

3.2.6 Phase 6: Prediction

In the prediction phase, the trained model is used to forecast the next day's closing stock price. Real data from the last prediction_days days is collected, scaled, and fed into the model to generate a prediction. The prediction is then inverse-transformed to obtain the actual predicted stock price value, which can be used for decision-making purposes.

Overall, these phases encompass the complete workflow of building, training, evaluating, and utilizing an LSTM model for stock price prediction. Each phase plays a crucial role in ensuring the effectiveness and reliability of the model in forecasting future stock prices.

3.3 LSTM

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture designed to address the vanishing gradient problem in traditional RNNs and effectively capture long-term dependencies in sequential data. Unlike standard RNNs, LSTM networks incorporate specialized memory cells with self-connected units known as gates, which regulate the flow of information within the network. These gates, including the input gate, forget gate, and output gate, are equipped with activation functions that control the information flow by selectively updating the memory cell state. This architecture enables LSTM networks to retain information over long sequences, making them particularly well-suited for tasks involving time-series data, natural language processing, and sequential pattern recognition. By learning which information to remember, forget, and update, LSTM networks excel at modeling complex temporal relationships and have become a cornerstone in various applications requiring the processing of sequential data.

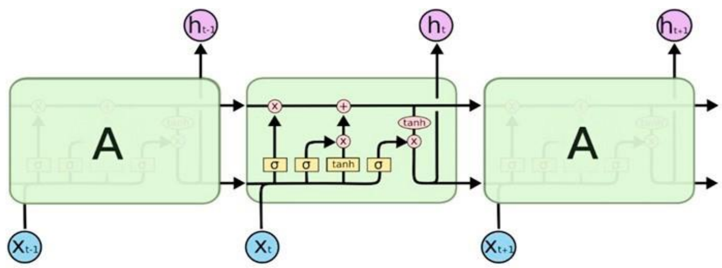


Figure 1. The internal structure of an LSTM.

Chapter 4

Experimental Setup and Results

4.1 Dataset Overview

The dataset retrieved from Yahoo Finance using Python typically consists of historical financial data for various assets, such as stocks, cryptocurrencies, and market indices. This dataset provides valuable information such as the opening price, closing price, highest price, lowest price, trading volume, and adjusted closing price for each trading day within a specified time range.

Using the yfinance library in Python, users can easily access this data through a simple and intuitive API. By specifying the ticker symbol of the asset of interest (e.g., 'AAPL' for Apple Inc.) and the desired time range, the **yf.download()** function retrieves the historical data as a pandas Data Frame.

Upon retrieval, the dataset can undergo preprocessing steps to prepare it for analysis or modeling. Common preprocessing techniques include scaling the data to normalize it, handling missing values, and splitting the data into training and testing sets for model evaluation.

Overall, the dataset obtained from Yahoo Finance serves as a valuable resource for various financial analyses, including but not limited to stock price prediction, trend analysis, and volatility modeling. Its rich historical data enables users to gain insights into market trends, make informed investment decisions, and develop predictive models to forecast future asset prices.

4.2 System Workflow

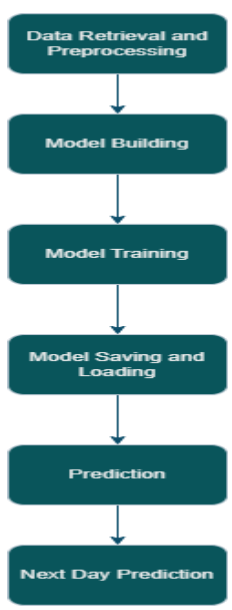


Figure 4.1: Steps for training the model

4.3 Dataset Extract and Preprocessing

Begins by retrieving historical stock price data from Yahoo Finance using the yfinance library. This data, essential for training and evaluating the LSTM model, is specific to a chosen company identified by its ticker symbol, such as 'AAPL' for Apple Inc. Retrieval parameters include a start date, typically set to the beginning of available historical data, and an end date, often set to the current date to capture the most recent information.

Once retrieved, the data undergoes preprocessing to prepare it for model training. This preprocessing primarily involves scaling using the MinMaxScaler from the sklearn.preprocessing module. Scaling normalizes the data between 0 and 1, ensuring consistent feature scales across the dataset, a prerequisite for effective neural network training. Each attribute in the dataset, including opening price, closing price, highest price, and lowest price, is independently scaled.

The training set is utilized to train the LSTM model, while the testing set evaluates its performance on unseen data, providing insights into its generalization capabilities.

4.4 Implementation Details

The implementation includes further steps for enhancing model evaluation and result visualization. After preprocessing, the dataset undergoes a splitting process, dividing it into training and testing sets. This separation allows for a comprehensive assessment of the model's performance on unseen data, helping to gauge its generalization capabilities and mitigate overfitting.

Furthermore, the accuracy of the model is calculated using various metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and accuracy percentage. The accuracy percentage specifically measures the proportion of predictions whose absolute error falls within a predefined threshold, often set at 5% for assessing the model's precision.

To aid in result interpretation, the code incorporates visualization capabilities using the matplotlib.pyplot library. This enables the plotting of actual and predicted stock prices, offering a clear graphical representation of the model's predictive performance. By visualizing the predictions alongside actual data, stakeholders can easily discern trends, patterns, and potential areas for improvement, thereby enhancing decision-making processes in investment and financial analysis contexts

4.5 Evaluation Metrics

4.5.1 Mean Absolute Error (MAE):

Mean Absolute Error (MAE) quantifies the average magnitude of errors between predicted ($\hat{y}_i$) and actual (y_i) values. It is calculated using the formula:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} |\hat{y}_i - y_i|$$

where:

- n is the total number of predictions.
- $\hat{y}_i$ is the predicted value.
- y_i is the actual value.

4.5.2 Mean Squared Error (MSE)

During model training, the mean squared error (MSE) loss function is used to measure the difference between the actual and predicted values:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Here, N is the number of samples, y_i is the actual closing price, and $\hat{y}_i$ is the predicted closing price.

4.5.3 *Root Mean Squared Error (RMSE):*

Root Mean Squared Error (RMSE) is the square root of the MSE and provides a measure of the spread of prediction errors in the same units as the predicted variable. It is calculated using the formula:

$$\mathrm{RMSE} = \sqrt{\mathrm{MSE}}$$

RMSE offers a more interpretable metric compared to MSE, as it is expressed in the same units as the predicted variable. Like MSE, lower RMSE values indicate better model performance.

4.5.4 *Accuracy Percentage:*

The accuracy percentage measures the proportion of predictions whose absolute error falls below a predefined threshold, typically set to 5%. It is calculated as follows:

$$\text{Accuracy Percentage} = \left(\frac{\text{Number of accurate predictions}}{\text{Total number of predictions}}\right) \times 100\%$$

where:

- Number of accurate predictions is the count of predictions with absolute error below the predefined threshold.
- Total number of predictions is the total count of predictions made.

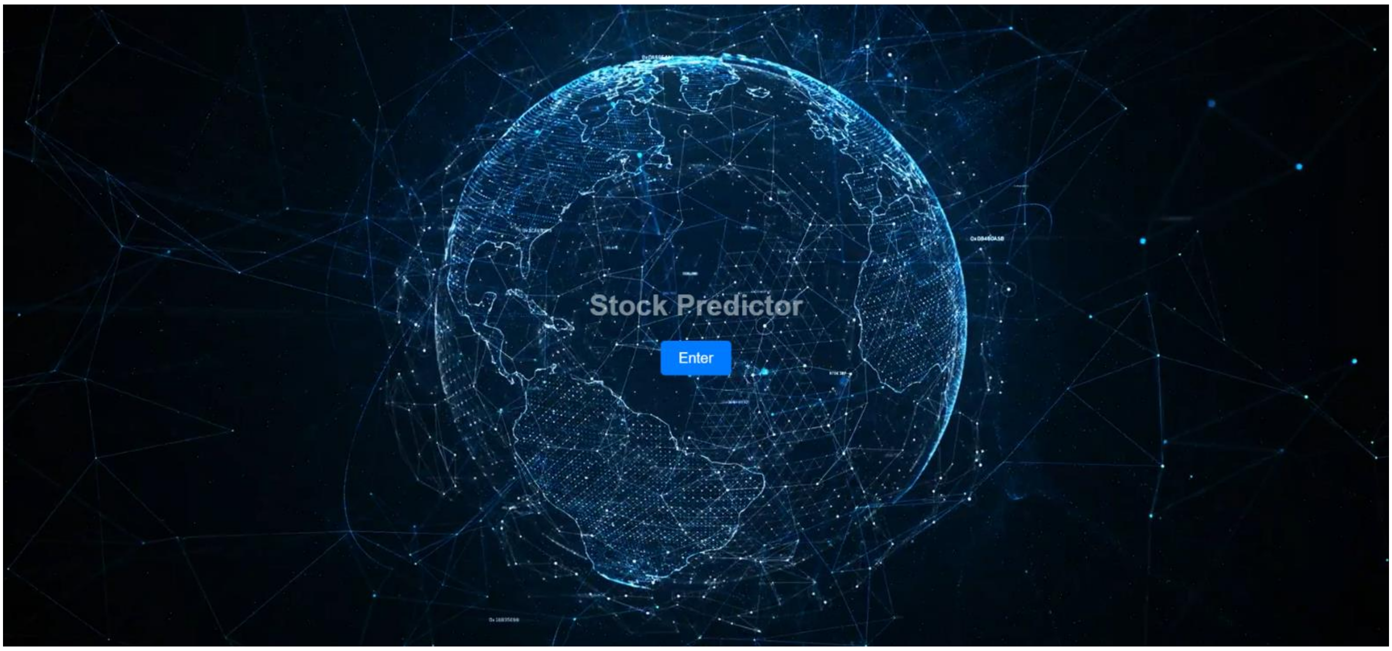
A higher accuracy percentage indicates a higher proportion of accurate predictions, suggesting better model performance in capturing the underlying patterns and trends in stock price movements.

Chapter 5

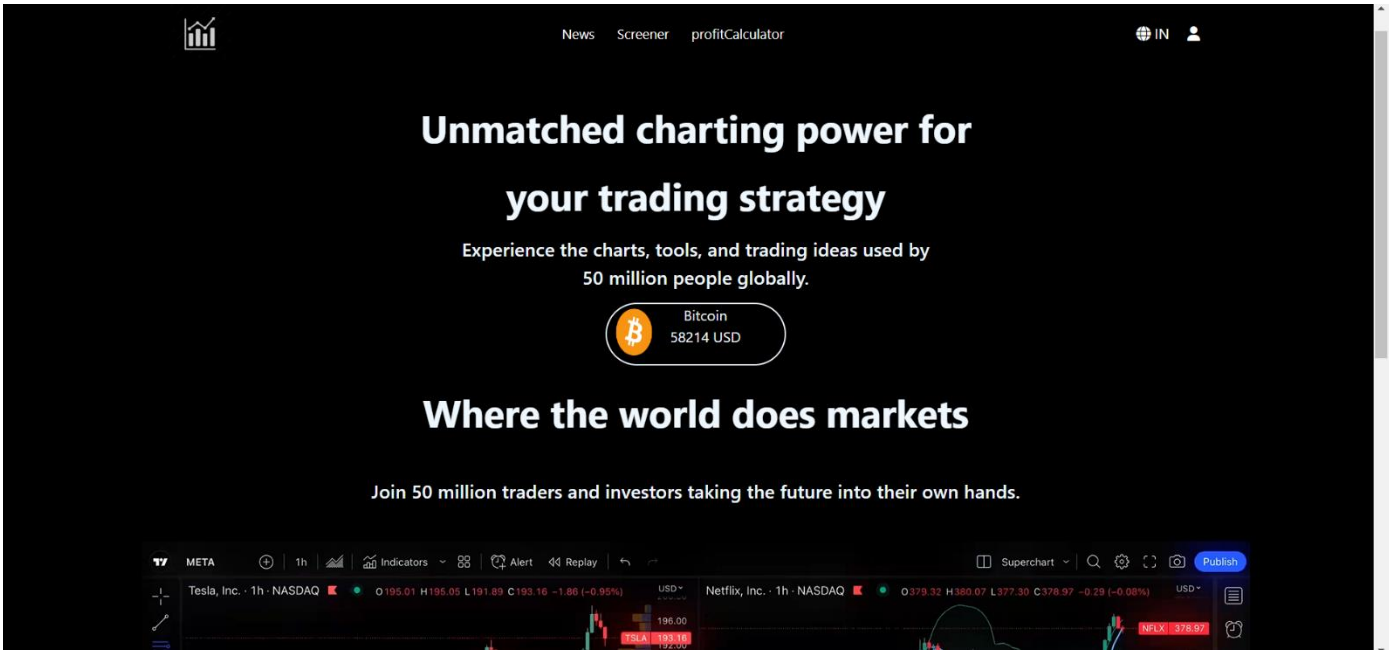
Web Interface

5.1 Fronted Pages

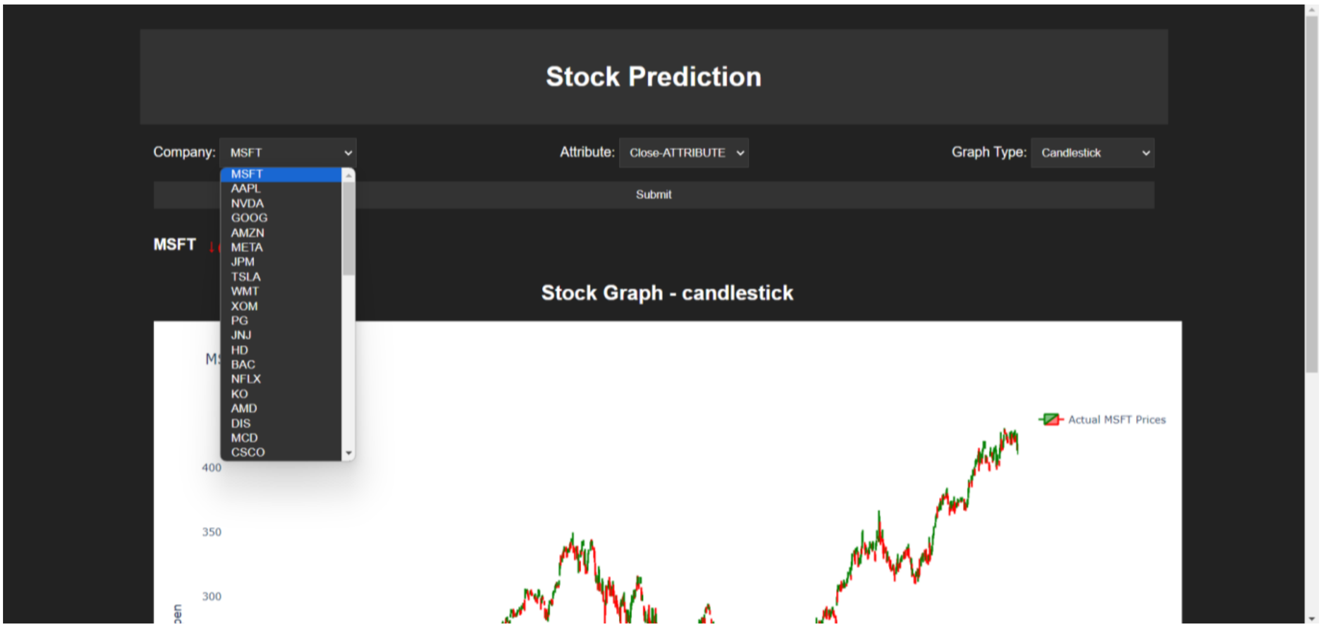
a) Welcome page:



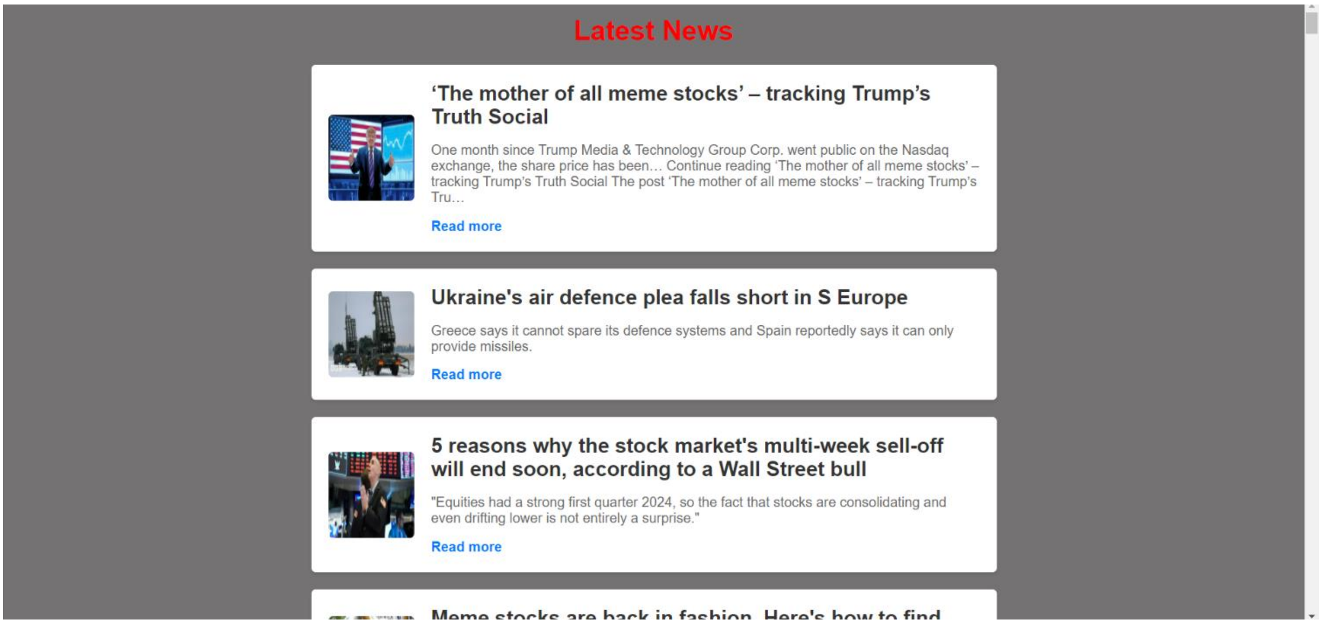
b) Home page:



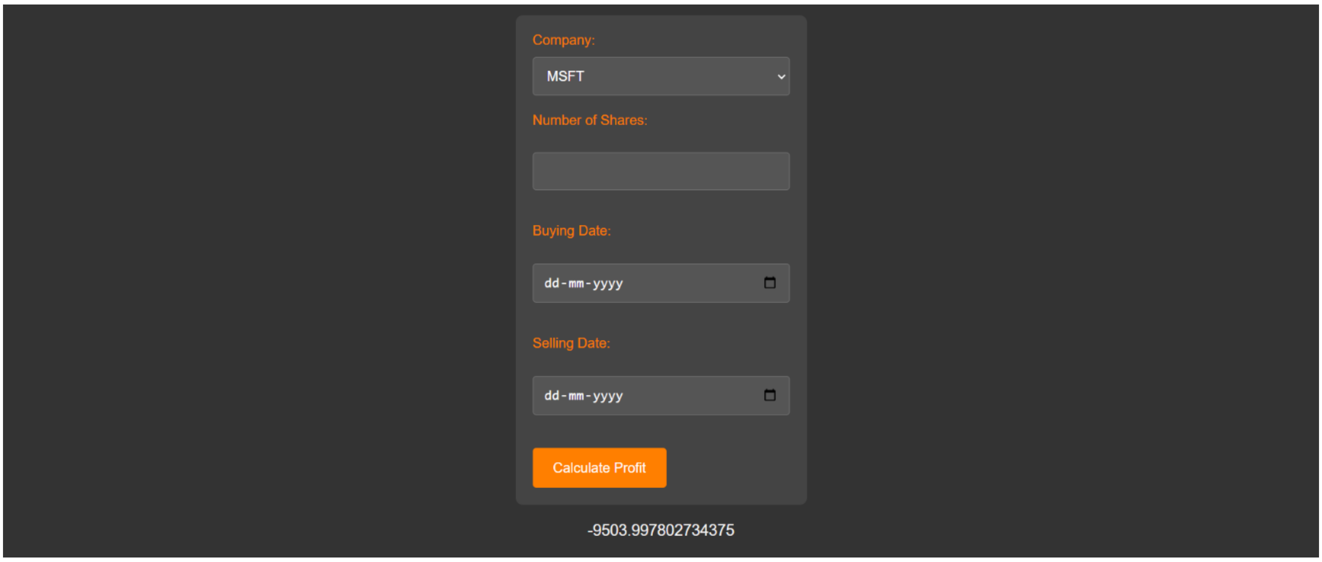
c) Prediction page:



d) News page:



e) Profit Calculator page:



5.2 Working Procedure

a) Welcome page:

A visually appealing welcome page for a website. It features a full-screen video background that plays automatically and loops continuously. The content, including a welcoming message and an "Enter" button, is centered on the page. The text color of the heading undergoes a subtle animation, transitioning between white and a lighter shade. When the "Enter" button is clicked, it triggers a JavaScript function that redirects the user to another page, likely the main page of the website, using Flask's **url_for** function. The combination of video background, animation, and interactive button provides an engaging introduction to the website, inviting users to explore further.

b) Home page:

A webpage for a stock visualization platform. It includes a navigation bar with links to different sections like Products, Community, Markets, News, Brokers, and More, each with nested dropdown menus for more specific options. The page also features sections for text content, including a headline promoting the platform's charting capabilities and a video section displaying market-related content. It incorporates Tailwind CSS for styling, Font Awesome for icons, and JavaScript for interaction, such as toggling the dropdown menus and redirecting users upon button click. Additionally, it utilizes Flask's **url_for** function to dynamically generate URLs for static assets and page redirects, enhancing the scalability and maintainability of the web application.

c) Prediction page:

A webpage for stock prediction functionality. It includes a form allowing users to select a company, attribute, and graph type for stock visualization. Upon submission, the form sends a POST request to "/home1" for processing. The page also displays information about the selected company, including a predicted value with an arrow indicator based on today's value, if available. Additionally, it features a container for displaying the stock graph, which is dynamically updated based on user input. The script section utilizes JavaScript to handle page load events, store and retrieve input values and graph data from session storage, and dynamically update the form inputs and graph display accordingly.

d) News page:

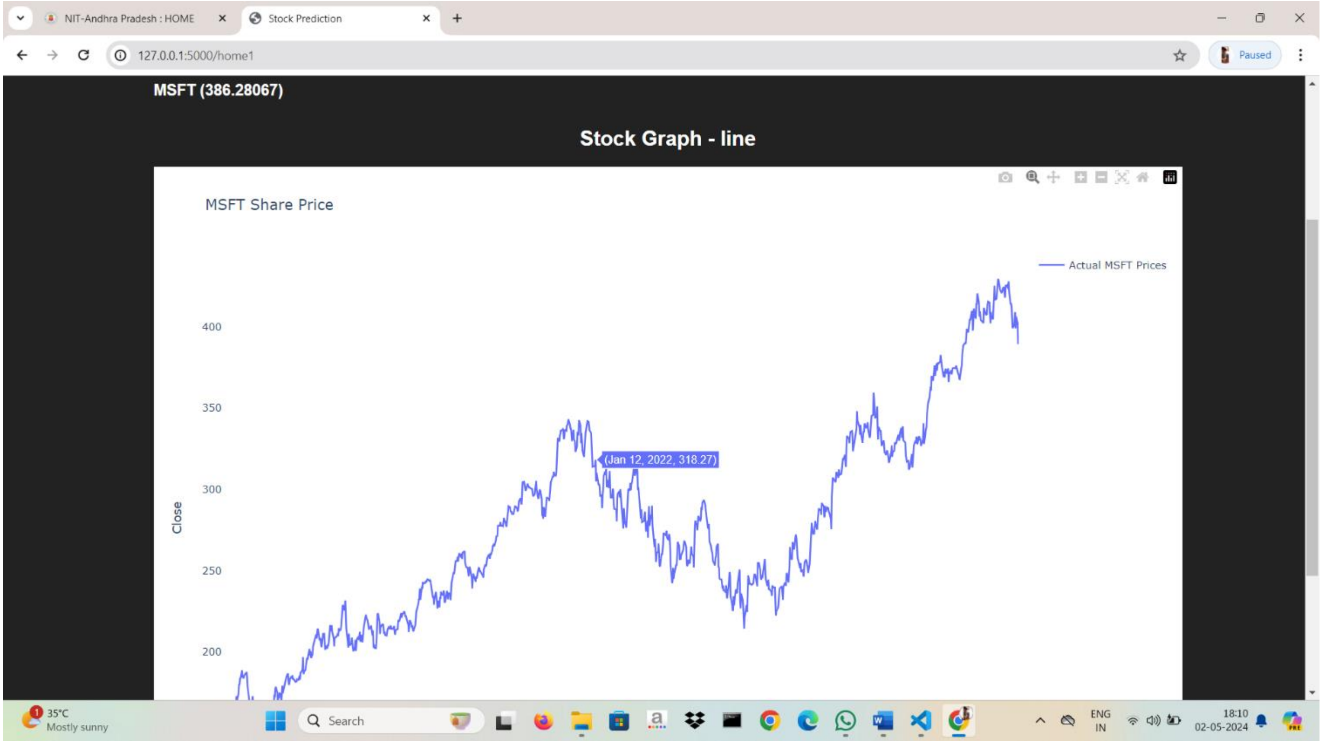
This HTML code template is designed for presenting news articles on a web page. It includes CSS styling for layout and visual presentation. The page structure consists of a title, "Latest News," and an unordered list (**<ul>**) where each article is displayed. Each article includes a link (**<a>**) to read more about it. The CSS styles define the layout, colors, and fonts for the elements. The code assumes dynamic content in the form of a list of articles, indicating its integration with server-side technologies for data rendering.

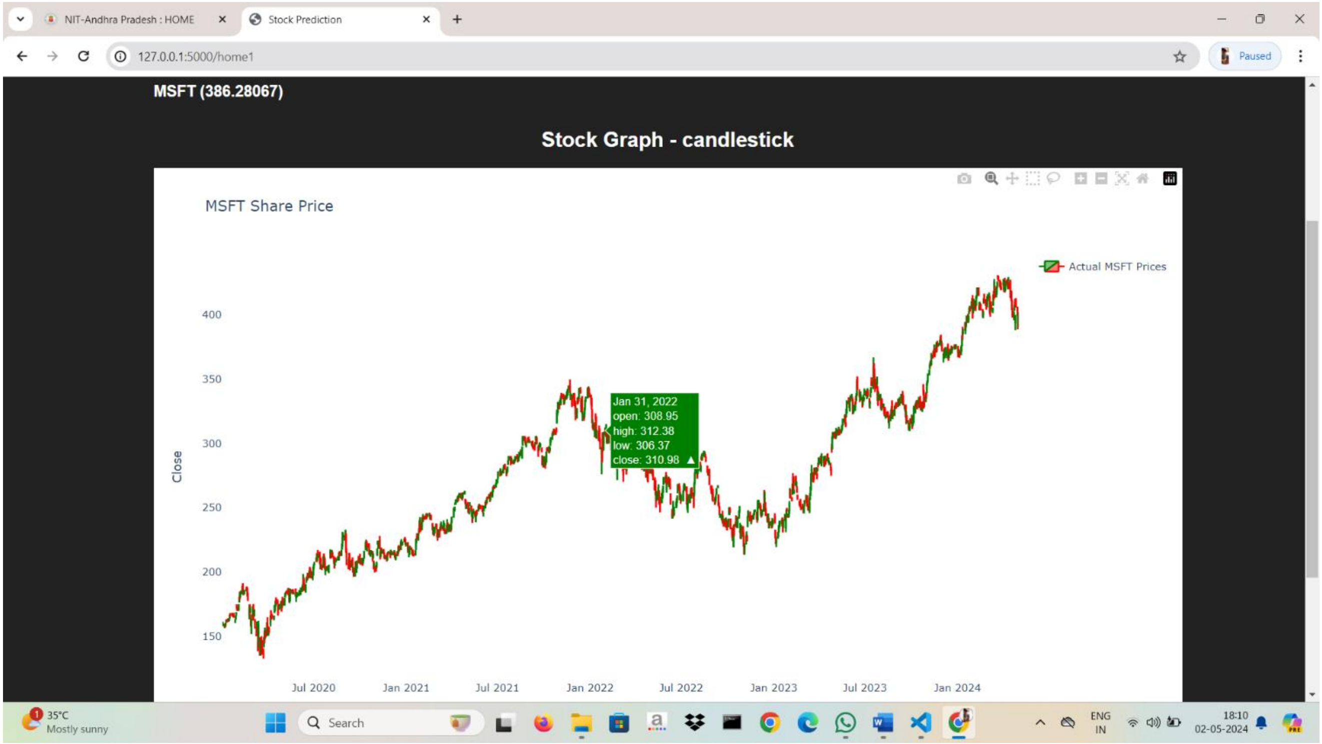
e) Profit Calculator page:

This HTML code template is for a form that calculates profit based on user input. It's structured with HTML elements and styled using CSS for visual presentation. The form includes input fields for selecting a company, entering the number of shares, and specifying buying and selling dates. Upon submission, the form sends data to a server-side endpoint for profit calculation. The CSS styles define the layout, colors, and fonts for the elements. The calculated profit is displayed below the form in a designated **<div>** with the id "profit".

Chapter 6

Result





Chapter 7

Conclusion and Future Work

Future work for this project could focus on several areas to enhance model accuracy and usability. Firstly, refining the LSTM architecture by experimenting with different configurations such as the number of layers, hidden units, and dropout rates could potentially improve predictive performance. Additionally, incorporating additional features such as technical indicators, sentiment analysis of news articles, or macroeconomic indicators could provide more comprehensive input data for the model, potentially leading to more accurate predictions. Moreover, implementing advanced techniques like attention mechanisms or hybrid models combining LSTM with other architectures could further enhance prediction capabilities. Furthermore, optimizing hyperparameters using techniques like grid search or Bayesian optimization could help fine-tune model performance. On the usability front, improving the web interface with features like real-time updating of predictions, customizable visualizations, and user-friendly tools for analysis could enhance the overall user experience.

References

[1] Stock Market Prediction Using LSTM Recurrent Neural NetworkAuthor links open overlay panelAdil Moghar a, Mhamed Hamiche b https://doi.org/10.1016/j.procs.2020.03.049

[2] HTML Tutorial (w3schools.com) for html

[3] https://www.w3schools.com/css for CSS

[4] JavaScript Tutorial (w3schools.com) for java script