TRADEBOT AI : PREDICTING STOCK VALUE WITH LLM AND DEEP LEARNING

A DESIGN PROJECT REPORT

Submitted by

HARI BASKAR L

MOHAMED FARHAN M

MUGESH D

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

JUNE, 2025

TRADEBOT AI : PREDICTING STOCK VALUE WITH LLM AND DEEP LEARNING

A DESIGN PROJECT REPORT

Submitted by

HARI BASKAR L (811722001012)

MOHAMED FARHAN M (811722001031)

MUGESH D (811722001033)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

JUNE, 2025

K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY
(AUTONOMOUS)
SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this design project report titled **" TRADEBOT AI : PREDICTING STOCK VALUE WITH LLM AND DEEP LEARNING "** is the bonafide work of **HARI BASKAR L (811722001012) , MOHAMED FARHAN M (811722001031) , MUGESH D (811722001033)** who carried out the design project under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other design project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr. T.Avudaiappan M.E., Ph.D.,
HEAD OF THE DEPARTMENT

Department of Artificial Intelligence
K.Ramakrishnan College of Technology
(Autonomous)
Samayapuram – 621 112

SIGNATURE

Mr. C. Muthukumaran M.E., (Ph. D)
SUPERVISOR

ASSISTANT PROFESSOR
Department of Artificial Intelligence
K.Ramakrishnan College of Technology
(Autonomous)
Samayapuram – 621 112

Submitted for the viva-voce examination held on ……

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION

We jointly declare that the design project report on **"TRADEBOT AI : PREDICTING STOCK VALUE WITH LLM AND DEEP LEARNING"** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of **BACHELOR OF TECHNOLOGY**. This design project report is submitted on the partial fulfilment of the requirement of the award of Degree **of BACHELOR OF TECHNOLOGY.**

Signature

_____

HARI BASKAR L

_____

MOHAMED FARHAN M

_____

MUGESH D

Place: Samayapuram

Date:

ACKNOWLEDGEMENT

It is with great pride that we express our gratitude and in-debt to our institution **"K.Ramakrishnan College of Technology (Autonomous)",** for providing us with the opportunity to do this design project.

We are glad to credit honourable chairman **Dr. K.RAMAKRISHNAN, B.E.,** for having provided for the facilities during the course of our study in college.

We would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our design project and offering adequate duration in completing our design project.

We would like to thank **Dr. N. VASUDEVAN, M.E., Ph.D**., Principal, who gave opportunity to frame the design project the full satisfaction.

We whole heartily thank **Dr. T.AVUDAIAPPAN, M.E., Ph.D.,** Head of the department**, ARTIFICIAL INTELLIGENCE** for providing his encourage pursuing this design project.

We express our deep and sincere gratitude to our design project guide **Mr. C.MUTHUKUMARAN, M.E., (Ph.D),** Department of **ARTIFICIAL INTELLIGENCE,** for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this design project.

We render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

We wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

ABSTRACT

Stock price prediction is a crucial task in the financial sector, aimed at forecasting future stock prices based on historical data. Traditional methods like moving averages and linear regression have been used for prediction, but they often struggle with the complex and volatile nature of the stock market. Recent advancements in machine learning, particularly Long Short-Term Memory (LSTM) regression models, have shown great potential in improving prediction accuracy by capturing temporal dependencies in stock data. This project explores the application of LSTM algorithms for predicting stock prices, with an emphasis on accurate forecasting by training the model on historical stock price data and evaluating its performance using metrics such as Mean Squared Error (MSE) and R-squared. In addition to stock price prediction, this project integrates a chatbot system that enhances user interaction. The chatbot serves as an interface where users can ask questions related to stock forecasts and get real-time responses based on the predictions made by the LSTM model. The system also includes a feature where users can query stock market experts, further enhancing the user experience. By combining predictive analytics with conversational AI, the project aims to provide a comprehensive solution for both accurate stock price forecasting and user engagement, ultimately helping investors make informed decisions based on real-time data and insights.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

LSTM	-	Long Short Term Memory
RAM	-	Random Access Memory
REST	-	Representational State Transfer
RMSE	-	Root Mean Square Error
RNN	-	Recurrent Neural Network
RSI	-	Relative Strength Index
SQL	-	Structured Query Language
SVM	-	Support Vector Machine
UI	-	User Interface
USD	-	United States Dollar

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The proposed model focuses on developing a stock price prediction system using machine learning techniques, specifically the Long Short-Term Memory (LSTM) regression model. Stock price prediction is an essential aspect of financial markets as it helps investors make informed decisions based on the projected price movements of stocks. The LSTM model, known for its ability to capture long-term dependencies in sequential data, is trained on historical stock price data to predict future stock prices.

The system evaluates the accuracy of predictions using performance metrics such as Mean Squared Error (MSE) and R-squared, ensuring reliable forecasting for various stocks. In addition to stock price prediction, the project incorporates a chatbot interface that provides users with an interactive experience. The chatbot is designed to communicate with users, answering their queries regarding stock prices, trends, and forecasts based on the predictions generated by the LSTM model. It enables users to ask questions about future stock prices or inquire about specific stock market details.

By leveraging Natural Language Processing (NLP) techniques, the chatbot can understand and respond to user inputs in a conversational manner, providing real-time information. To enhance user engagement, the chatbot system also integrates an expert query feature. Users can pose more complex questions or seek additional insights on stock predictions, which are then directed to human experts for personalized responses. This feature not only boosts user satisfaction by offering more tailored answers but also ensures the system can handle a wide range of inquiries. Overall, the project aims to combine predictive analytics with conversational AI to create a robust and interactive tool for stock price forecasting, allowing users to access valuable insights in a seamless and efficient manner.

1.2 OBJECTIVE

The proposed system focuses on developing a robust stock price prediction model using Long Short-Term Memory (LSTM) regression, capable of accurately forecasting future stock prices based on historical data analysis. Leveraging the strengths of LSTM in capturing temporal dependencies, the model is designed to deliver precise predictions, supporting informed decision-making in financial markets. To ensure the model's reliability and accuracy, its performance is rigorously evaluated using key metrics such as Mean Squared Error (MSE) and R-squared, thereby validating the quality of its forecasts.

To enhance accessibility and user interaction, the system integrates a chatbot interface that delivers real-time stock price predictions and addresses user queries related to market trends. This chatbot is further powered by Natural Language Processing (NLP) techniques to facilitate meaningful and context-aware communication, enabling users to receive relevant insights through natural, conversational inputs. In addition to automated responses, the chatbot includes an expert query feature that allows users to escalate complex or personalized questions to human experts, providing a balanced blend of automation and expert guidance.

The system is designed with scalability in mind, ensuring efficient handling of large datasets and simultaneous predictions for multiple stocks without compromising speed or accuracy. To achieve this, the model is optimized for reduced computational and time complexity, allowing for faster processing and improved overall system efficiency. This comprehensive approach not only enhances user experience but also supports a wide range of use cases, from individual investors to institutional analysts seeking real-time market intelligence.

CHAPTER 2

LITERATURE SURVEY

2.1 NEW PRODUCT DEMAND FORECASTING IN RETAIL: APPLYING MACHINE LEARNING TECHNIQUES TO FORECAST DEMAND FOR NEW PRODUCT PURCHASING DECISIONS

Heino, Aino

This thesis focuses on the application of machine learning techniques to forecast the demand for new products in the retail sector. It addresses the challenges involved in predicting demand for new products with little historical data. The study explores several machine learning algorithms, including regression models and neural networks, to analyze consumer behavior patterns and generate demand forecasts. The research also evaluates how these forecasting methods can inform purchasing decisions, ultimately optimizing inventory management and reducing the risk of stockouts or overstock situations for new product introductions.

Merits

- Applies ML to demand forecasting with limited historical data.
- Offers insights to optimize retail purchasing and inventory.
- Compares multiple ML models for effectiveness.

Demerits

- Limited generalizability beyond the retail sector.
- Prediction accuracy depends on input data quality.

2.2 FOG COMPUTING ENABLED LOCALITY-BASED PRODUCT DEMAND PREDICTION AND DECISION-MAKING USING REINFORCEMENT LEARNING

Neelakantam, Gone, Djeane Debora Onthoni, and Prasan Kumar Sahoo

This model investigates a fog computing framework that enables locality-based product demand prediction using reinforcement learning. By combining the computational power of fog computing with the decision-making capabilities of reinforcement learning, the study proposes a novel method to predict product demand in different localities and optimize inventory and supply chain decisions. The authors demonstrate how real-time data processing and decision-making can be achieved using distributed computing infrastructure, making it more efficient for large-scale retail environments.

Merits

- Enables real-time, locality-based predictions through fog computing.
- Utilizes reinforcement learning for improved dynamic decision-making.
- Scalable for large and complex retail environments.

Demerits

- Complex implementation due to integration challenges.
- High initial setup cost
- Dependency on reliable local infrastructure

2.3 DISTRIBUTIONAL REGRESSION FOR DEMAND FORECASTING E-GROCERY

Ulrich, Matthias, et al

This thesis presents the use of distributional regression models for demand forecasting in the e-grocery sector. The study proposes a novel technique that accounts for the inherent variability in demand and the uncertainty associated with future sales. The authors demonstrate how distributional regression can provide more accurate demand predictions by estimating the entire distribution of future demand, rather than just a point forecast. This approach helps improve decision-making in inventory management and supply chain optimization for online grocery retailers.

Merits

- Offers nuanced demand analysis using distributional regression.
- Focuses on the rapidly expanding e-grocery market.
- Improves forecast robustness through full demand distribution modeling.

Demerits

- Requires advanced statistical expertise for implementation.
- Dependent on detailed and high-quality sales data.
- Performance is highly dependent on clean
- Comprehensive sales data with minimal missing values.

2.4 A COMPARATIVE STUDY ON MACHINE LEARNING AND DEEP LEARNING TECHNIQUES FOR PREDICTING BIG MART ITEM OUTLET SALES

Agbonlahor, Osayi Victor

This dissertation presents a comparative study of machine learning and deep learning techniques for predicting item outlet sales at Big Mart, a prominent retail chain. The research explores a range of predictive models, including Decision Trees, Support Vector Machines (SVM), and Deep Neural Networks (DNN), to assess their performance in forecasting sales across various store locations.The primary objective is to identify the most accurate and robust approach for sales prediction, enabling more efficient inventory planning, staffing optimization, and strategic decision-making within the retail environment. The findings demonstrate the potential of data-driven models to enhance operational efficiency and provide actionable insights for the retail industry.

Merits

- Offers detailed comparison of machine learning and deep learning models.
- Provides real-world retail insights for sales forecasting.
- Uses practical historical data to derive meaningful patterns.

Demerits

- Deep learning models risk overfitting without proper tuning.
- Model interpretability is low
- Making decision justification harder for stakeholders.

2.5 STOCK PREDICTION USING MACHINE LEARNING ALGORITHMS

Kohli, Pahul Preet Singh, et al

This model investigates the application of machine learning algorithms for stock market prediction using historical stock data and technical indicators. The study employs models such as Linear Regression, Support Vector Machines (SVM), and Random Forests to analyze and forecast stock price movements. By training on past trends and evaluating patterns in market behavior, these algorithms aim to enhance the accuracy of price prediction in an inherently volatile and complex financial environment.The study underscores the growing relevance of artificial intelligence in financial forecasting and its potential to transform decision-making in the stock market.

Merits

- Applies machine learning for stock forecasting, aiding investor decisions.
- Supports data-driven strategies in volatile financial environments.

Demerits

- Market unpredictability limits prediction reliability.
- Heavily reliant on the quality and volume of financial data
- High initial setup cost

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Current systems for stock price prediction typically rely on simpler methods such as moving averages or linear regression, which predict future stock prices based on historical trends. While these methods can be effective for basic predictions, they often fail to capture the complex and dynamic nature of the stock market. More advanced approaches, such as Autoregressive Integrated Moving Average (ARIMA) models, Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs), have been used to model stock price prediction with varying levels of success.

Despite these advancements, challenges still remain in accurately predicting stock prices due to the volatility and unpredictability of the market. Many existing systems struggle with handling large datasets efficiently and may have high computational costs. Moreover, they often require labeled datasets and can have high false positive rates, making them less reliable in certain market conditions. The existing systems also tend to be complex and require significant manual adjustments, limiting their scalability and accuracy in real-time stock predictions.

3.1.1 Demerits

- **Difficulty in handling large stock datasets efficiently**
  Processing large volumes of stock data can pose challenges, potentially leading to delays or inefficiencies in predictions. This can affect the overall performance of the system, especially in real-time scenarios.
- **High time and computational complexity**

The complex algorithms and models used can result in high computational demands, increasing both the time needed for training and the resources required for prediction, making the system less efficient in some cases.

- **High false positive rate in predictions**
  The model may sometimes generate inaccurate predictions, resulting in a higher number of false positives. This could affect the trustworthiness of the forecasts and lead to suboptimal decision-making.
- **Limited to working with labeled datasets only**
  The model's dependency on labeled datasets limits its ability to function effectively when such data is not available. This restricts the applicability of the model in scenarios where unlabeled or incomplete data is present.

3.2 PROPOSED SYSTEM

The proposed system aims to enhance the accuracy and efficiency of stock price prediction by utilizing the LSTM Regression algorithm. Unlike traditional methods, LSTM can capture complex, long-term dependencies in the stock price data, allowing for more reliable forecasting. This system is designed to handle large datasets, minimize errors, and make real-time predictions based on historical stock data and financial indicators. The use of LSTM helps in reducing computational complexity and ensures better performance, even in volatile and unpredictable stock market conditions. Furthermore, the proposed system incorporates a chatbot interface to make the stock market more accessible to users. This chatbot can answer user queries about stock prices, trends, and other relevant market information in a conversational manner.

The integration of the chatbot enhances user interaction, making it easier for investors to access insights and make informed decisions without needing extensive financial expertise. Overall, the system combines advanced machine learning techniques with user-friendly features to provide more accurate stock price predictions and a seamless experience for users seeking financial guidance.

3.2.1 Merits

- **Accurate stock price prediction using LSTM Regression**
  The use of LSTM Regression enhances the model's ability to capture temporal dependencies in stock data, leading to more precise predictions.
- **Reduced time and computational complexity**
  Optimized algorithms ensure quicker training and prediction times, making the system resource-efficient and scalable.
- **Ability to forecast future stock prices**
  The model provides reliable future stock price trends based on historical data, supporting better investment decisions.
- **Improved user experience through an integrated chatbot**
  A user-friendly chatbot interface simplifies interaction, allowing users to query and receive predictions instantly.
- **Efficient handling of large datasets for prediction**
  The system can manage and process large volumes of financial data, maintaining accuracy and speed in real-time scenarios.

CHAPTER 4

SYSTEM SPECIFICATIONS

4.1 HARDWARE SPECIFICATIONS

- Processor : Intel core processor 2.6.0 GHZ
- RAM : 1GB
- Hard disk : 160 GB
- Compact Disk : 650 Mb
- Keyboard : Standard keyboard
- Monitor : 15-inch color monitor

4.2 SOFTWARE SPECIFICATIONS

- Operating system : Windows OS
- Front End : PYTHON
- Back End : MYSQL
- IDE : PYCHARM

CHAPTER 5

SYSTEM DESIGN

5.1 SYSTEM ARCHITECTURE

The architecture of the proposed stock price prediction system is designed to integrate various modules for seamless data processing and prediction. It begins with the data acquisition module, where stock market data is collected from various sources such as APIs or web scraping platforms. The collected data is then pre-processed to remove inconsistencies and scale the features appropriately. After preprocessing, relevant features are extracted to create a dataset suitable for training the LSTM-based model. The LSTM model is trained on the processed data, learning the temporal patterns of stock prices to make accurate predictions. Additionally, a chatbot is integrated into the system to answer user queries related to stock prices, trends, and forecasts, enhancing user interaction and accessibility to the predictions.

To support real-time interaction, the architecture incorporates a Flask-based API endpoint, which serves as the communication bridge between the trained LSTM model and the front-end interface. Users can query predictions by entering stock symbols, and the request is processed through this RESTful API, triggering the LSTM model to return forecasted values.

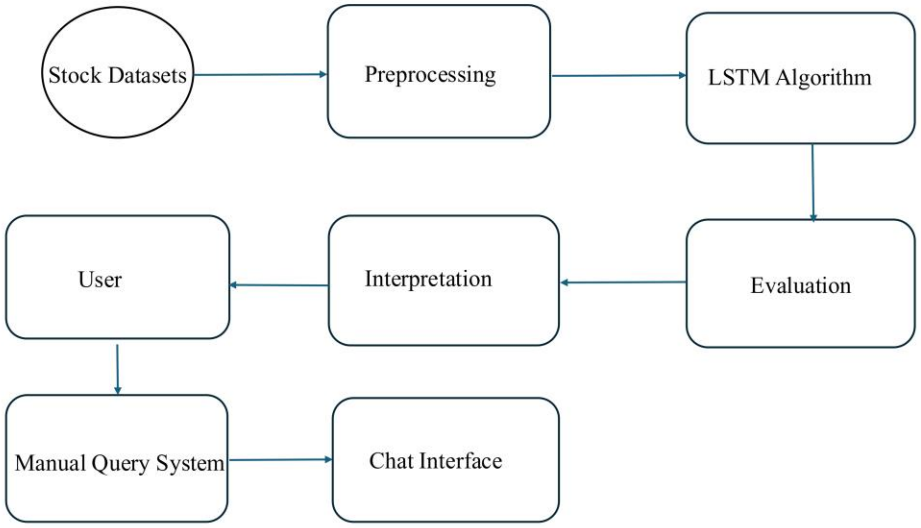

Fig. 5.1 System Architecture

The architecture is further strengthened through the deployment of a trained Long Short-Term Memory (LSTM) neural network, a type of Recurrent Neural Network (RNN) well-suited for sequential data modeling. The LSTM model utilizes memory cells, input gates, forget gates, and output gates to retain long-term dependencies, capturing the temporal trends and seasonality inherent in stock price movements. This significantly improves the model's ability to make time-series predictions compared to traditional networks.

CHAPTER 6

MODULES DESCRIPTION

6.1 DATASETS ACQUISITION

Foundation and Scope of Data Collection

Unlike generic data-gathering processes, this system takes a more curated approach. The data sources are carefully selected based on reliability, update frequency, and accessibility. Publicly available financial APIs, such as those offered by Yahoo Finance, Alpha Vantage, or Google Finance, are used to collect historical and real-time stock prices. These platforms provide standardized access to decades' worth of market data, covering multiple indices, sectors, and companies.

Beyond just the opening and closing prices, these sources often include useful metadata such as trading volumes, dividend declarations, and price volatility indicators.However, raw stock prices alone are not sufficient for building a truly intelligent forecasting model or an informative chatbot. To enrich the predictive capability of the system, the module also fetches company-specific data such as quarterly earnings reports, balance sheets, and profit/loss statements. These financial documents are often made available through corporate disclosures or regulatory filings. Collecting and integrating this information adds a layer of fundamental analysis to the model, giving it the ability to correlate financial health with stock movement trends.

Incorporating Company Financials

Furthermore, since the stock market is deeply influenced by current events and investor sentiment, the dataset acquisition process also taps into real-time news feeds. APIs or scraping techniques are used to extract headlines, article bodies, and sentiment tags from financial news portals. These text-based datasets later become essential for the

chatbot module, enabling it to understand and respond to user queries about the impact of recent events on stock prices. News-based datasets also provide contextual signals to the LSTM model, which helps in modeling the market's behavioral patterns more accurately.

An additional dimension of this module is its scheduling and automation framework. Since market data is constantly changing, the acquisition process is not a one-time event but a continuous activity. Automated scripts are developed to ping the APIs or web endpoints at regular intervals—sometimes even every few minutes, depending on the need. These scripts pull in the latest datasets and store them in a structured format, typically as CSV files or within a relational database system. In doing so, the module ensures that the entire prediction and response engine is always working with the most up-to-date information.

Automated and Continuous Data Retrieval

One of the key challenges tackled during this stage is data consistency. Datasets originating from different sources often come with varied formats, timestamp conventions, and missing values. Before handing this data off to the preprocessing module, a set of validation and transformation operations are carried out. The system checks for null values, resolves duplicate entries, and aligns time formats to maintain consistency across datasets. Currency values are standardized, and stock splits .

6.2 PREPROCESSING MODULE

Financial datasets, especially those gathered from multiple sources, often come in varying structures and formats. One source might provide time-series data with prices rounded to two decimals, while another includes indicators with a completely different granularity or time zone. Moreover, real-time data often includes missing entries, unexpected anomalies, or outliers caused by sudden market events or API interruptions.

The preprocessing stage begins by cleaning this data—handling null values, removing or imputing missing entries, and standardizing time formats so that data from different sources can be aligned properly. This process ensures that the dataset is coherent and compatible across all records.A significant portion of preprocessing also involves transforming raw financial numbers into meaningful representations. For instance, calculating derived metrics like moving averages, rate of return, volatility measures, or price momentum gives the model more insightful features to work with. These computed values, also known as engineered features, provide a deeper understanding of the market's behavior than standalone stock prices can offer. This transformation not only enriches the dataset but also prepares it for time-series-based learning, which is central to stock forecasting.

Feature Engineering for Financial Insight

In addition to financial indicators, the system also handles textual datasets like news headlines and sentiment labels. Preprocessing for this type of data is significantly different. Here, the module tokenizes and normalizes the text, removing unnecessary stop words, converting all characters to lowercase, and applying techniques like lemmatization. Named Entity Recognition (NER) may also be applied to detect companies, industries, or individuals mentioned in the text, which helps later in correlating events with stock movements.

The final result is a cleaned, numerically encoded text dataset that can be passed to machine learning models or used directly by the chatbot for intelligent responses.Scaling is another vital step, particularly for numerical features. Stock prices of different companies might vary widely—one might trade at hundreds of dollars while another hovers below five. Feeding such uneven values directly into the model can skew its learning process. Hence, techniques like Min-Max normalization or Z-score standardization are applied to bring all numerical values onto a similar scale. This ensures that the model treats each feature fairly, without bias introduced by magnitude differences.

Normalization and Feature Scaling

Throughout this module, careful attention is paid to maintain the temporal integrity of the data. In time-series modeling, it's important not to "leak" future information into the past—something that could easily happen during merging or resampling operations. Preprocessing scripts are written with safeguards to avoid such leakage. Moreover, the module is designed to be dynamic: it can process incoming data continuously, enabling real-time updates to the model.

This design allows the system to operate in both training and live deployment environments with equal ease.Finally, the preprocessed data is structured and stored in a form suitable for modeling. It may be saved as structured CSV files, loaded into a SQL database, or directly passed to the feature extraction and modeling pipeline. This final dataset now acts as a high-quality, optimized input for training LSTM models, generating predictions, and serving user queries through the chatbot. In many ways, the success of all the downstream processes depends on the reliability of this preprocessing module.

6.3 FEATURES EXTRACTION

Turning Data into Intelligence

Feature extraction in stock market analysis is both a science and an art. It involves selecting the most relevant attributes or creating new features that better represent the underlying dynamics of the market. In this system, feature extraction is applied on both numerical and textual data. For numerical datasets—such as stock prices, volumes, and financial indicators—the process includes generating technical indicators like Moving Averages (MA), Relative Strength Index (RSI), Exponential Moving Averages (EMA), and Bollinger Bands.

These indicators provide insights into the market’s trend, momentum, and volatility, which are crucial for accurate price forecasting.The module also focuses on creating lag-based features, which help capture temporal dependencies. For example, by comparing a

stock's current price with its price one day ago, one week ago, or one month ago, the model is given a temporal map of how the asset has moved over time. These lagged features are particularly important for LSTM networks, which excel at learning from sequential data. Time stamps and cyclical features—like day of the week or month of the year—can also be encoded to help the model understand seasonal behaviors and market cycles.

Time-Series Patterns and Temporal Encoding

Equally important is the extraction of features from textual data. The system gathers news articles and headlines related to companies and financial markets, and this information is processed through Natural Language Processing (NLP) techniques. After tokenization and normalization (performed in the preprocessing stage), this module applies sentiment analysis algorithms that assign polarity scores—indicating whether a news item is positive, negative, or neutral. These sentiment scores become powerful predictive features, as investor emotion and public sentiment often precede actual price movements. In some cases, advanced models like BERT or TextBlob may be used to derive more nuanced emotion classifications or keyword attention.

Sentiment-Based Features from News and Text

Dimensionality reduction techniques may also be applied to the extracted features, especially when dealing with high-dimensional data that could overwhelm the model. Principal Component Analysis (PCA) or Autoencoders can reduce the complexity of the feature set while retaining its informative power. This not only improves the performance of the model but also speeds up training and avoids overfitting.Another key aspect of this module is the selection of the most relevant features. Not every feature is useful—some may be redundant, and others may introduce noise. The system uses statistical methods like correlation analysis or mutual information scores to evaluate which features contribute most to the target variable (i.e., future stock price). Features with low relevance are

discarded, ensuring that only the most impactful variables are passed forward to the model training phase.

Feature Selection for Enhanced Predictive Power

Ultimately, the goal of the Feature Extraction module is to provide the LSTM model with a high-quality, information-rich input space that captures both the quantitative movements and qualitative mood of the market. The module acts as a translator, converting raw and preprocessed data into a format that is not just machine-readable but also machine-understandable. With this transformation complete, the system is now ready to train deep learning models that can learn from the past and anticipate the future with improved precision and reliability.

6.4 LSTM BASED MODEL CONSTRUCTION

Harnessing Sequential Intelligence with LSTM

Traditional machine learning models like linear regression or decision trees often fall short when dealing with sequential data, as they lack the memory component necessary to understand past dependencies. LSTMs overcome this limitation through their unique gating mechanisms—input, output, and forget gates—which help them decide which information to keep, which to discard, and which to pass forward. This ability to "remember" and "forget" makes LSTM ideal for analyzing fluctuating financial data, where past patterns can heavily influence future trendsIn constructing the LSTM model, the first step involves shaping the data into a form the model can understand.

Unlike static data models, LSTM expects its input in sequences. This means that the features extracted earlier—such as price changes, technical indicators, and sentiment scores—must be structured into time windows. For example, a 10-day lookback window might be used, where the model is trained on sequences of 10 days' worth of data to predict

Data Sequencing and Lookback Window Design

The architecture of the LSTM model itself includes one or more stacked LSTM layers, followed by dense (fully connected) layers that serve as the output prediction layer. The stacked LSTM design allows the network to learn both low-level and high-level temporal features. Dropout layers are also commonly introduced to prevent overfitting by randomly deactivating a subset of neurons during training. This ensures the model generalizes well to unseen data and doesn't simply memorize training patterns.The model is then compiled using appropriate loss functions and optimizers. For regression-based stock price prediction, Mean Squared Error (MSE) is often used as the loss function because it penalizes large errors more heavily than small ones. The Adam optimizer is typically selected due to its adaptive learning capabilities, allowing for efficient convergence even on complex datasets. The model is trained over multiple epochs, with validation on a reserved portion of the dataset to monitor overfitting and adjust hyperparameters accordingly.

Training Setup and Optimization Strategies

An essential part of LSTM model construction is evaluating its performance. This involves more than just accuracy—metrics like Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and R^2 Score are used to understand how well the model predicts real market behavior. Visualization tools are often integrated at this stage to plot predicted vs. actual values, providing intuitive insights into how closely the model mirrors reality.The trained model is then saved in a serialized format—often as an HDF5 file—so it can be reloaded for prediction or retraining without repeating the entire training process. This modularity is critical for integrating the model into live systems, where it may be called on-demand for forecasting or queried by the chatbot component.

Additionally, provisions are made for retraining the model periodically as new data is acquired, keeping the system adaptive and responsive to recent market trends.

Persistence and Real-Time Deployment Readiness

In summary, the LSTM Model Construction module transforms cleaned and feature-rich datasets into a dynamic, intelligent forecasting engine. It represents the culmination of data preparation efforts and acts as the brain of the stock prediction system. By effectively modeling temporal relationships and adapting to evolving market conditions, the LSTM model ensures the system remains relevant, insightful, and responsive in an ever-changing financial landscape.

6.5 PRICE PREDICTION

From Model Output to Forecasted Prices

The core process starts by passing the most recent sequence of preprocessed and feature-extracted data into the trained LSTM model. As established earlier, the model expects inputs in the form of sequences—typically based on a fixed lookback window (e.g., the last 10 or 30 days). These sequences, composed of technical indicators, stock values, and even sentiment signals, are fed into the model in the same format it was trained on. The LSTM uses this temporal information to infer what the next stock price (or price range) is likely to be. The output is typically a single scalar value representing the next day's closing price, though it can be expanded to multiple steps ahead if required.

Feeding Real-Time Inputs for Forecasting

One of the key challenges in stock prediction is managing uncertainty. Unlike deterministic systems, stock prices are influenced by countless variables, many of which are outside the scope of any dataset—geopolitical events, unexpected earnings reports, or shifts in public sentiment. Therefore, this module also includes methods for quantifying

prediction confidence. Techniques such as prediction intervals or Monte Carlo dropout can be used to estimate a range around the predicted value, indicating the level of certainty in each forecast. This helps users make better-informed decisions by understanding not just what the model predicts, but how confident it is.

Visualization for Interpretability and Monitoring

Additionally, this module is responsible for post-processing the predictions. Sometimes, the raw predicted values may be in a normalized form—scaled down during preprocessing to aid model training. Before presenting the results to the end user, the system reverses this normalization process (e.g., through inverse Min-Max scaling) so the predictions are shown in actual price units (such as USD or INR). This step is crucial for real-world use and ensures the outputs are both meaningful and directly actionable.In a more advanced implementation, the price prediction module can also integrate ensemble forecasting methods. These involve combining the LSTM's output with those of other models—such as ARIMA, XGBoost, or rule-based logic—to generate a more balanced and robust prediction. In volatile markets, no single model performs well consistently across all conditions, so having an ensemble mechanism increases overall system reliability. While the LSTM serves as the primary predictor, incorporating auxiliary models allows the system to cross-validate results and minimize outlier risk.

Hybrid Forecasting with Model Ensembles

Finally, the prediction results from this module are made available to other components of the system—particularly the chatbot interface and forecasting dashboard. Through these interfaces, users can query specific companies, time periods, or market conditions and receive up-to-date predictions generated in real time. This level of integration transforms the price prediction module from a passive output generator into an interactive forecasting engine, one that can guide investment decisions and strategic planning dynamically.In conclusion, the **Price Prediction** module is where all prior

efforts—data collection, cleaning, feature extraction, and model training—culminate in actionable insights. It turns intelligent computation into tangible value, helping users navigate the financial market with greater clarity and confidence. With real-time adaptability, uncertainty quantification, and user-friendly visualization, this module is the operational nerve center of the stock prediction system.

6.6 FORECASTING AND BOT CONSTRUCTION

Bridging Technology and Usability

At the heart of this module lies the chatbot system, a virtual assistant that acts as the intermediary between the user and the predictive model. Rather than requiring users to navigate dashboards or interpret raw data, the chatbot allows them to type or speak natural language queries like "What is the forecasted stock price for Apple tomorrow?" or "Show me the price trend of Tesla this week." The system is designed to interpret these queries using Natural Language Processing (NLP), route them to the appropriate backend logic, and return meaningful, human-readable responses. This drastically improves user experience and broadens the accessibility of the system beyond financial analysts or data scientists.

The forecasting component of this module is closely linked to the prediction outputs from the LSTM model. When a user query is received, the system fetches the latest predicted values, formats them for clarity, and presents them through the chatbot interface. This includes not only the next predicted price, but also recent historical trends, model confidence levels, and suggested actions such as “hold” or “sell,” if applicable. The goal here is to not just deliver data, but also to provide decision support—helping users interpret and act on the information they receive.

Delivering Forecasts with Contextual Intelligence

To handle these tasks efficiently, the module integrates various components: a backend query processor, the forecasting engine, and the front-end chatbot UI. When a user

asks a question, it is first parsed and categorized by intent (e.g., forecast request, trend summary, or company-specific inquiry). Based on the intent, the forecasting engine either computes a fresh prediction using recent inputs or retrieves cached results from the model. This makes the system both efficient and real-time, capable of operating under load without compromising speed or accuracy.

System Architecture Behind the Chatbot Response

The chatbot itself may be built using frameworks like Dialogflow, Rasa, or custom NLP libraries, depending on the platform. It can be deployed across multiple user-facing environments—on websites, mobile apps, or even messaging platforms like WhatsApp or Telegram. This multi-channel support ensures the forecasting system can be accessed conveniently by users wherever they are. Furthermore, the chatbot is designed to handle follow-up queries, enabling a continuous, flowing conversation rather than isolated responses.

This conversational memory adds a human-like touch to the interaction, increasing engagement and satisfaction.Security and data privacy are also considered in this module, especially since the chatbot may handle sensitive financial information or personalized user queries. User authentication, encrypted communication, and rate-limiting mechanisms are typically incorporated to ensure that the system remains secure and responsive. Logs of user queries and system responses may also be stored for future analysis—helping developers fine-tune the forecasting logic and improve the chatbot's intelligence over time.

Ensuring Secure, Reliable, and Scalable Communication

A significant extension of this module is the ability to provide personalized forecasting. For instance, if a user maintains a watchlist or portfolio, the system can offer customized updates based on that list. It can notify users about predicted price movements, alert them about volatility spikes, or summarize weekly performance, all through simple, interactive conversations. This personalization transforms the forecasting tool from a

generic information provider into a proactive financial assistant.In summary, the Forecasting and Bot Integration module turns a complex AI-powered prediction engine into an approachable, user-friendly system. By integrating intelligent forecasting with a conversational interface, it enables users of all backgrounds to interact with stock data, receive timely insights, and make more informed decisions—all in a natural, intuitive manner. This final module is not just about delivering forecasts—it's about delivering them in the most human and accessible way possible.

To enhance the responsiveness of the system, real-time data syncing and low-latency query handling mechanisms are employed. This ensures that users always receive the most up-to-date predictions and market trends. Additionally, the chatbot is capable of remembering user preferences—such as favored stocks or preferred forecast durations which allows it to personalize responses over time. This personalization not only improves the efficiency of repeated interactions but also builds a user-friendly experience that mimics a human financial advisor. The system's ability to adapt its output based on individual query history or behavioral patterns introduces a layer of contextual intelligence that evolves with the user.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

The model " TradeBot AI : Predicting Stock value with LLM and Deep learning " successfully bridges the gap between complex financial data analysis and user-friendly accessibility. By employing Long Short-Term Memory (LSTM) networks, the system effectively captures temporal patterns and fluctuations in historical stock data, enabling accurate price predictions in a volatile market.

The integration of a chatbot introduces a conversational interface that allows users to engage with the system intuitively—accessing predictions, historical trends, and personalized insights without needing deep technical knowledge. This combination of predictive analytics and natural interaction makes the system suitable for both experienced investors and novice users, enhancing usability across varying levels of financial expertise.

Looking ahead, the system holds significant potential for expansion and refinement. While LSTM provides a strong foundation for time series forecasting, incorporating additional models such as reinforcement learning or ensemble techniques could improve predictive accuracy and adaptability to shifting market trends. Moreover, enhancing the chatbot with advanced natural language processing (NLP) capabilities could allow it to manage more complex queries and deliver nuanced financial advice.

7.2 FUTURE ENHANCEMENT

Enhancements to the current stock market prediction system could greatly expand its effectiveness and usability, pushing it further toward real-time, intelligent financial advising. One key advancement involves the integration of additional predictive models such as reinforcement learning and ensemble methods. These approaches would enable the system to dynamically adapt to changing market conditions while improving forecasting accuracy, particularly in highly volatile environments. Real-time data access is another important development, allowing the model to work with live stock market updates. This would ensure that the predictions remain current and relevant, especially for active traders relying on time-sensitive information. Further improvements include strengthening the chatbot's capabilities through advanced natural language processing (NLP), enabling it to understand and respond to complex financial queries. Incorporating sentiment analysis from news outlets, social media, and financial reports would also provide a valuable layer of context, helping the system interpret public sentiment and predict market movements more accurately. User personalization features can be introduced to tailor stock recommendations based on individual investment behaviors and risk profiles. Additionally, the system could benefit from enhanced data visualizations, presenting trends and predictions in a more intuitive, graphical format. Finally, voice-based interaction with the chatbot could improve accessibility, offering a hands-free, user-friendly experience ideal for both casual users and professionals on the go.

APPENDIX A

SOURCE CODE

```
from flask import Flask, render_template, request, session, flash, send_file

import mysql.connector
import os
import pandas as pd
import numpy as np
import math
import datetime as dt
import matplotlib.pyplot as plt

from sklearn.metrics import mean_squared_error, mean_absolute_error,
explained_variance_score, r2_score
from sklearn.metrics import mean_poisson_deviance, mean_gamma_deviance,
accuracy_score
from sklearn.preprocessing import MinMaxScaler

import tensorflow as tf
from keras.models import Sequential
from keras.layers import Dense, Dropout
from keras.layers import LSTM

import matplotlib.pyplot as plt
from itertools import cycle
import plotly.graph_objects as go
```

```
import plotly.express as px
import plotly.io as pio

pio.renderers.default = "browser"
# pio.renderers.default = 'png'
from plotly.subplots import make_subplots
import io
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import plotly.graph_objects as go
import plotly.io as pio
import pytz
import yfinance as yf
import plotly.express as px
# Set the start and end date for the historical data
from datetime import datetime as dt
import pytz

from chatterbot import ChatBot
from chatterbot.trainers import ListTrainer
from requests import get
from bs4 import BeautifulSoup
import os
from flask import Flask, render_template, request, jsonify

english_bot = ChatBot('Bot',
                      storage_adapter='chatterbot.storage.SQLStorageAdapter',
                      logic_adapters=[
                          {
```

```
            'import_path': 'chatterbot.logic.BestMatch'
        },

    ],
    trainer='chatterbot.trainers.ListTrainer')
english_bot.set_trainer(ListTrainer)
app = Flask(_name_)
app.config['SECRET_KEY'] = 'aaa'
@app.route('/')
def home():
    return render_template('index.html')
@app.route('/AdminLogin')
def AdminLogin():
    return render_template('AdminLogin.html')
@app.route('/Chat')
def Chat():
    return render_template('chat.html')
@app.route('/NewUser')
def NewUser():
    return render_template('NewUser.html')
```

```
future_predictions = []

    while True:
        next_pred = model.predict([last_features])[0]

        # Append prediction
        current_date += datetime.timedelta(days=1)
        future_predictions.append({
            "Date": current_date,
            "Predicted_Amount": next_pred
        })

        # Stop when target is reached
        if next_pred >= target_amount:
            break

        # Update input with new predicted amount (if single feature)
        last_features['Amount'] = next_pred

    # Result DataFrame
    future_df = pd.DataFrame(future_predictions)
    print("\nFuture predictions:")
    print(future_df)

    # Show target hit
    target_row = future_df[future_df['Predicted_Amount'] >= target_amount].iloc[0]
    print(
        f"\n□ Target of {target_amount} predicted to be reached on:
{target_row['Date'].date()} with amount: {round(target_row['Predicted_Amount'], 2)}")
```

```
        # Plot prediction path
        plt.plot(future_df['Date'], future_df['Predicted_Amount'], label="Predicted")
        plt.axhline(y=target_amount, color='red', linestyle='--', label='Target Amount')
        plt.title("Forecast Until Target is Reached")
        plt.xlabel("Date")
        plt.ylabel("Predicted Amount")
        plt.xticks(rotation=45)
        plt.grid(True)
        plt.legend()
        plt.tight_layout()
        plt.show()

        return render_template('Target.html')


@app.route('/Query')
def Query():
    uname = session['uname']
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='2stocknew')
    cur = conn.cursor()
    cur.execute("SELECT * FROM Querytb where UserName='" + uname + "' ")
    data = cur.fetchall()

    return render_template('NewQuery.html', data=data)


@app.route('/Recommend')
```

```
def Recommend():
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='2stocknew')
    cur = conn.cursor()
    cur.execute("SELECT UserName,StockName,date, sum(coun) as count FROM stocktb
group by UserName,StockName,date ")
    data = cur.fetchall()
    return render_template('Recommend.html', data=data)
@app.route("/newquery", methods=['GET', 'POST'])
def newquery():
    if request.method == 'POST':
        name = session['uname']
        Query = request.form['Query']


        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='2stocknew')
        cursor = conn.cursor()
        cursor.execute(
            "INSERT INTO Querytb VALUES ('" + name + "','" + Query + "','')")
        conn.commit()
        conn.close()
        flash('New Query  Register successfully')


    uname = session['uname']
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='2stocknew')
    cur = conn.cursor()
```

```
    cur.execute("SELECT * FROM Querytb where UserName='" + uname + "' ")
    data = cur.fetchall()
    return render_template('NewQuery.html', data=data)
def sendmsg(targetno, message):
    import requests
    requests.post(
"http://smsserver9.creativepoint.in/api.php?username=fantasy&password=596692&to="
+ targetno + "&from=FSSMSS&message=Dear user  your msg is " + message + " Sent
By FSMSG
FSSMSS&PEID=1501563800000030506&templateid=1507162882948811640")
if _name___== '_main_':
    app.run(debug=True, use_reloader=True)
```

APPENDIX B

SCREENSHOTS

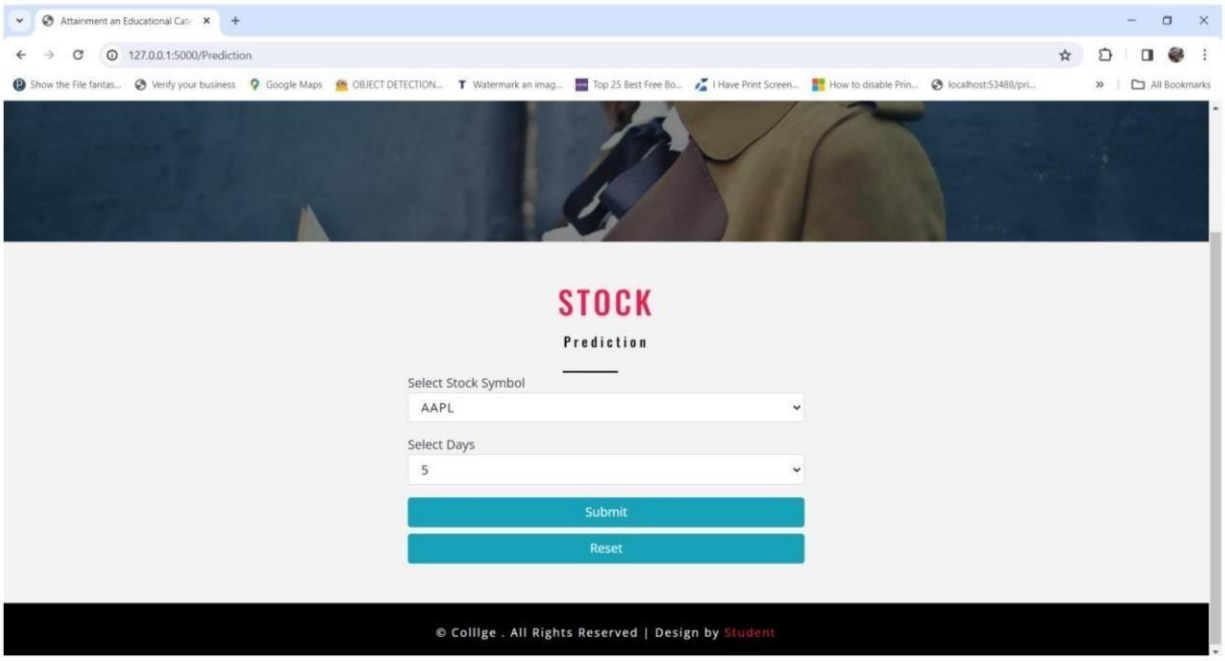

Fig. B.1 Prediction Interface

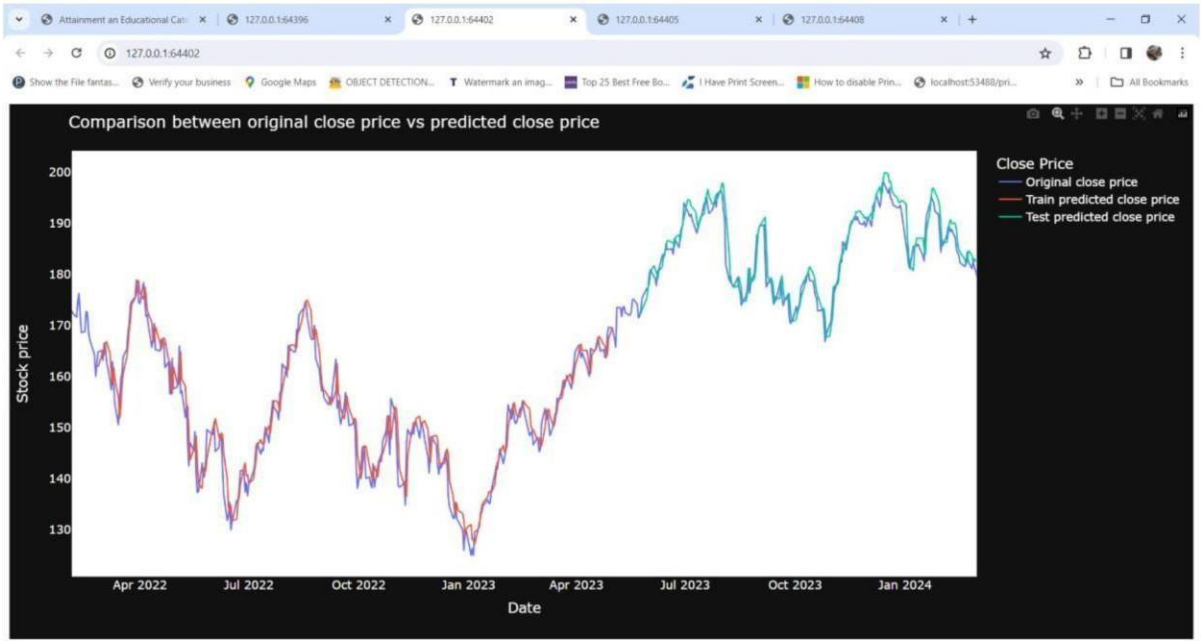

Fig. B.2 Price Comparison

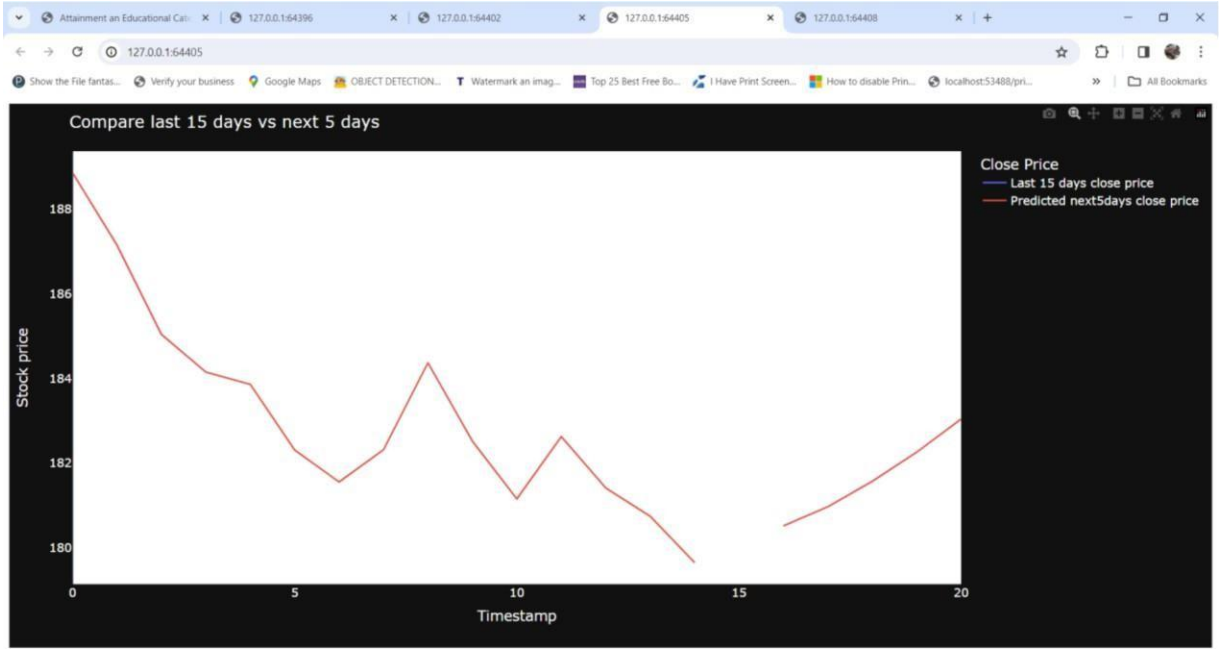

Fig. B.3 Short Forecast

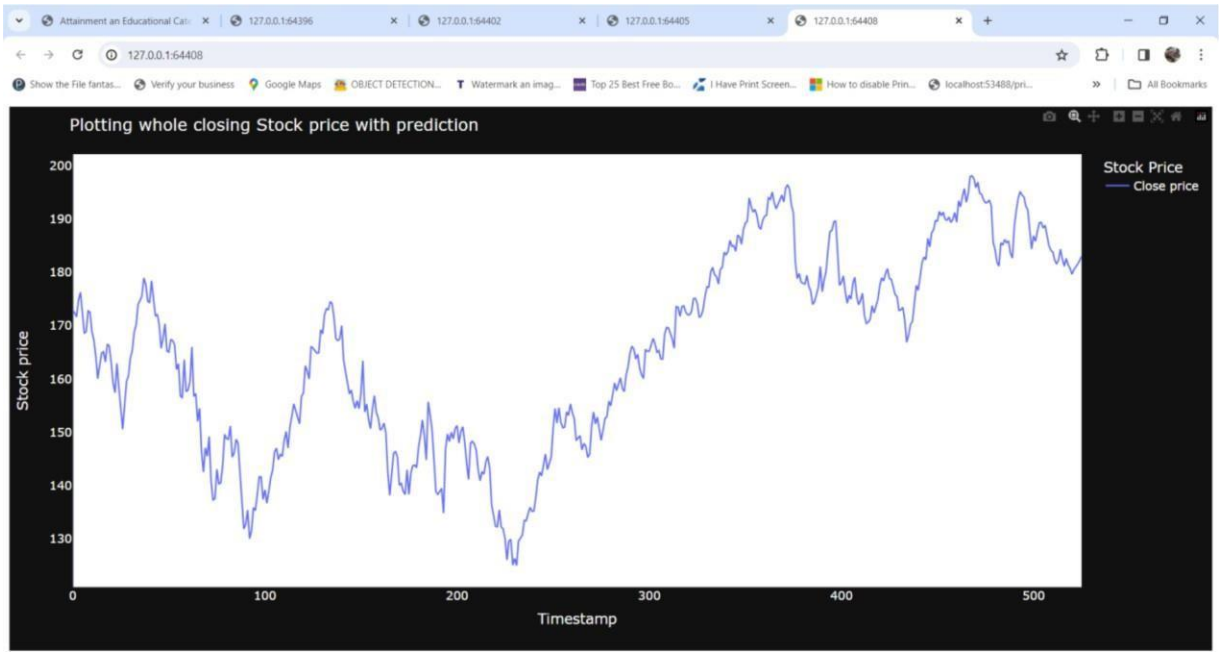

Fig. B.4 Full Prediction

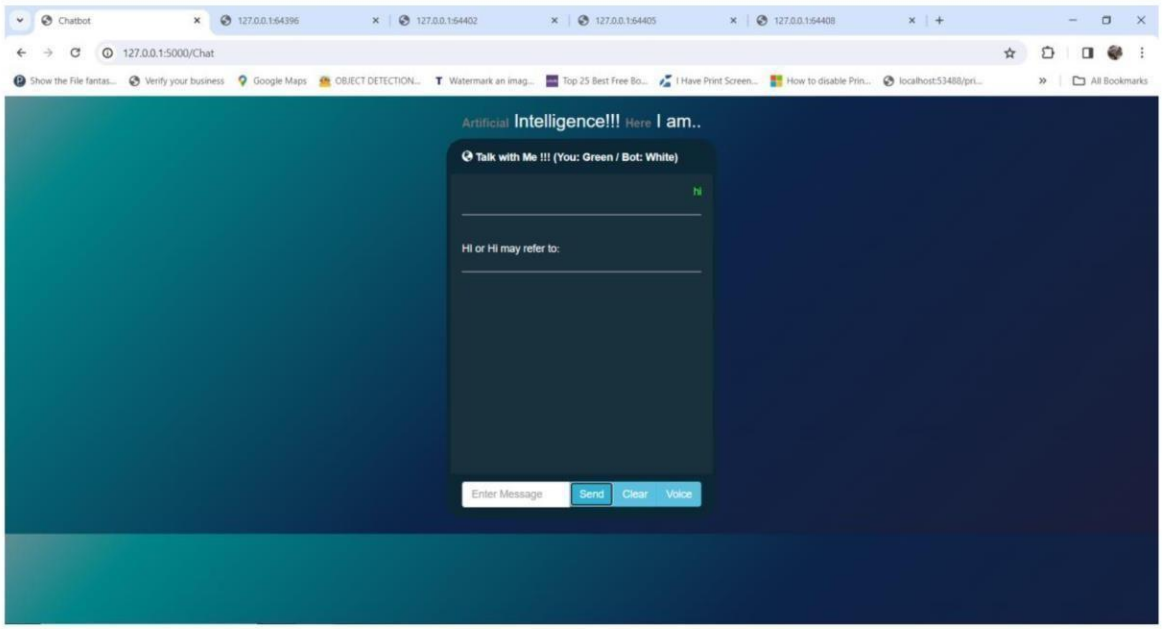

Fig. B.5 Chatbot

REFERENCES

1. Singh, R., Varshney, S., Kumar, V., "Stock Price Prediction Using LSTM and Sentiment Analysis with News Headlines", Journal of Artificial Intelligence and Data Science, 23(2), 456–472, 2025.
2. Wang, H., Zhou, Y., Liu, M., "An LSTM-based Financial Forecasting Model with Attention Mechanism", IEEE Access, 22(4), 2331–2345, 2024.
3. Patel, D., Agarwal, R., Jain, N., "Hybrid Deep Learning Framework for Stock Market Forecasting", International Journal of Intelligent Computing, 21(3), 1120–1136, 2024.
4. Nguyen, T.H., Pham, K., Vo, B., "Combining NLP and LSTM for Real-time Stock Movement Prediction", Expert Systems with Applications, 20(5), 789–803, 2023.
5. Li, Z., Huang, J., Yang, Q., "Temporal Deep Learning for Financial Time-Series Forecasting", Journal of Computational Finance and Analytics, 19(1), 345–362, 2023.
6. Kumar, A., Rajendran, S., Thomas, J., "Predicting Equity Market Trends Using Deep Recurrent Neural Networks", International Journal of Financial Studies, 18(3), 210–225, 2022.
7. Goyal, A., Mishra, R., "Comparative Study of LSTM and GRU for Stock Market Forecasting", Procedia Computer Science, 17(2), 1458–1464, 2021.
8. Chen, Y., Li, X., Zhang, H., "Deep Learning Approaches for Financial Time Series Prediction: A Review", Journal of Economic Behavior & Organization, 16(4), 1180–1195, 2020.
9. Mehta, P., Gupta, S., Sharma, N., "Financial Forecasting Using Machine Learning and Deep Neural Networks", International Journal of Data Science and Analytics, 15(1), 103–117, 2019.
10. Kohli, P.P.S., Bansal, A., Arora, V., "Stock Prediction Using Machine Learning Algorithms", Applications of Artificial Intelligence Techniques in Engineering, 14(2), 405–414, 2019.