A PROJECT REPORT

on

“ANALYSIS AND FORECASTING OF STOCK MARKET TRENDS”

Submitted to

KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

BACHELOR’S DEGREE IN

COMPUTER SCIENCE AND ENGINEERING

BY

SAMRIDDHA SIL 2105741

ARYAN BHATTACHARJEE 2105702

SARBAJIT NEOGI 2105822

ABANTI GHOSH 21052892

UNDER THE GUIDANCE OF

NACHIKETA TARASIA



SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAR, ODISHA - 751024

April 2024

KIIT Deemed to be University

School of Computer Engineering

Bhubaneswar, ODISHA 751024



CERTIFICATE

This is certify that the project entitled

“ANALYSIS AND FORECASTING OF STOCK MARKET TRENDS“

submitted by

SAMRIDDHA SIL 2105741

ARYAN BHATTACHARJEE 2105702

SARBAJIT NEOGI 2105822

ABANTI GHOSH 21052892

is a record of bonafide work carried out by them, in the partial fulfillment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science & Engineering) at KIIT Deemed to be university, Bhubaneswar. This work is done during year 2023-2024, under our guidance.

Date: 07/04/2024

Nachiketa Tarasia

Project Guide

**Acknowledgements**

We are profoundly grateful to **NACHIKETA TARASIA** of **Affiliation** for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion. .....................

SAMRIDDHA SIL

ARYAN BHATTACHARJEE

SARBAJIT NEOGI

ABANTI GHOSH

**ABSTRACT**

This study presents the development and implementation of a comprehensive stock trading application that incorporates an automated buying and selling system driven by a Long Short-Term Memory (LSTM) prediction model. Built upon a foundation of real-time market data sourced from Google Finance, the application leverages the predictive capabilities of LSTM algorithms to forecast stock market trends with a high degree of accuracy. Through the seamless integration of advanced machine learning techniques and algorithmic trading principles, the app enables users to automate trading decisions based on LSTM-generated predictions.

The application's automated trading module continuously monitors market conditions and executes buy or sell orders in real-time, according to the signals provided by the LSTM prediction model. This sophisticated system allows users to capitalize on favorable market movements while minimizing the impact of human emotions and biases on trading decisions. By providing detailed insights into predicted market trends and facilitating automated trading execution, the app offers users a powerful tool for optimizing investment strategies, managing portfolios, and achieving financial goals with greater efficiency and precision.

**Keywords:** Long Short-Term Memory, Market Prediction, Time Series Analysis, Sequential Data Modeling, Neural Networks, Predictive Modeling.

Contents

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | Introduction | | | 8 |
| 2 | Basic Concepts/ Literature Review | | | 9 |
|  | 2.1 | Sub Section Name........................... | | 9 |
| 3 | Problem Statement / Requirement Specifications | | | 11 |
|  | 3.1 | Project Planning........................... | | 11 |
|  | 3.2 | Project Analysis (SRS)................. | | 12 |
|  | 3.3 | System Design ………………….. | | 13 |
|  |  | 3.3.1 | Design Constraints …… | 14 |
|  |  | 3.3.2 | System Architecture (UML) / Block Diagram … | 15 |
| 4 | Implementation | | | 17 |
|  | 4.1 | Methodology / Proposal ........................... | | 17 |
|  | 4.2 | Testing / Verification Plan ……………. | | 19 |
|  | 4.3 | Result Analysis / Screenshots …………. | | 20 |
|  | 4.4 | Quality Assurance …………………….. | | 22 |
| 5 | Standard Adopted | | | 23 |
|  | 5.1 | Design Standards . . . . . . . . . . . . . . . | | 23 |
|  | 5.2 | Coding Standards . . . . . . . . . . . . . . | | 23 |
|  | 5.3 | Testing Standards . . . . . . . . . . . . . . . | | 24 |
| 6 | Conclusion and Future Scope | | | 25 |
|  | 6.1 | Conclusion ……………………….. | | 25 |
|  | 6.2 | Future Scope ………………………. | | 25 |
| References | | | | 26 |
| Individual Contribution | | | | 27 |
| Plagiarism Report | | | | 28 |

List of Figures

|  |  |  |
| --- | --- | --- |
| 3.3 | SYSTEM ARCHITECTURE ......................... | 2 |
| 4.  4.2  4.2  4.2  4.2  4.2 | GRAPH OF EXPECTED RETURNS .........................  GRAPH OF CLOSE PRICE HISTORY .........................  GRAPH OF RESULTS .........................  RESULTS .........................  RESULTS OF AUTO-TRADING BOT .........................  RESULTS OF AUTO-TRADING BOT ......................... | 12  12  13  13  14  14 |

Chapter 1

Introduction

The stock market can be complex and volatile, making it challenging for investors to consistently make profitable decisions. Traditional methods often rely on technical analysis or intuition, which can be prone to human error and emotional biases.

Current Solutions have a number of gaps such as:

Limited Automation: Existing stock trading platforms may offer basic tools for analysis, but lack the ability to automate trading decisions based on complex predictions.

Accuracy Concerns: Traditional analysis methods might not capture the full picture, potentially leading to inaccurate predictions and missed opportunities.

Emotional Influence: Human emotions can cloud judgment and lead to impulsive decisions that could negatively impact investment strategies.

This project addresses these gaps by developing a comprehensive stock trading application with the following key features:

Automated Trading: LSTM, a powerful machine learning model, automates buy and sell decisions based on its predictions, removing emotional influences.

Improved Accuracy: LSTMs excel at handling sequential data like stock prices, potentially leading to more accurate market predictions compared to traditional methods.

Data-Driven Insights: The application leverages real-time market data from Google Finance, allowing users to base their investment decisions on objective data and analysis.

Chapter 2

Basic Concepts/ Literature Review

Navigating the stock market effectively hinges on the ability to anticipate future trends. Stock market prediction refers to the process of forecasting future stock prices or overall market movements to guide investment decisions. Traditionally, investors relied on technical analysis tools or intuition, which can be prone to errors and biases. This project proposes a novel approach utilizing Long Short-Term Memory (LSTMs), a type of machine learning model. LSTMs excel at handling sequential data like time series, making them particularly well-suited for tasks like stock price prediction. Unlike traditional methods that may only capture short-term trends, LSTMs can learn long-term dependencies within sequences, potentially leading to more accurate forecasts. Additionally, LSTMs can model non-linear relationships between past and future prices, a crucial capability for navigating the complexities of the stock market. A critical aspect of LSTMs is their ability to continuously learn and improve their predictions as they are exposed to new data. This adaptability is advantageous in the ever-changing market landscape.

Several studies have explored the potential of LSTMs in stock market prediction, highlighting both their strengths and weaknesses.

**2.1 Strengths:**

* **Sequential Data Handling:** LSTMs excel at capturing long-term dependencies within sequential data like stock prices, potentially leading to more accurate predictions compared to traditional methods like moving averages or technical indicators.
* **Non-Linear Relationships:** LSTMs can learn complex non-linear relationships between past and future prices, which can be crucial for accurate prediction in volatile markets.
* **Adaptability:** LSTMs can continuously learn and improve their predictions as they are exposed to new data.

**2.2 Weaknesses:**

* **Data Dependency:** LSTMs require a large amount of historical data to be trained effectively. Limited data can lead to a phenomenon called overfitting, where the model performs well on the training data but fails to generalize to unseen data.
* **Hyperparameter Tuning:** Tuning the hyperparameters (learning rate, number of layers, etc.) of an LSTM model can be a complex process that significantly impacts its performance.
* **Black Box Nature:** While LSTMs can generate impressive predictions, understanding the reasoning behind their forecasts can be challenging due to their "black box" nature.

## Comparison to Traditional Methods

Several studies have compared LSTMs to traditional technical analysis methods like moving averages or the Relative Strength Index (RSI). These studies suggest that LSTMs have the potential to outperform these methods in terms of prediction accuracy, especially for complex market trends. However, it's important to remember the limitations of LSTMs discussed earlier. Additionally, the inherent volatility of the market can still impact the performance of any prediction model, including LSTMs. Unforeseen events or extreme market movements can introduce noise that the model may not account for.

## Additional Considerations

It's crucial to integrate LSTMs into a comprehensive trading strategy that considers other factors like risk management and portfolio diversification. By combining the predictive power of LSTMs with sound investment principles, investors can potentially make more informed decisions and achieve their financial goals with greater efficiency and precision.

Chapter 3

Problem Statement / Requirement Specifications

**Problem Statement:** Predicting the stock market is tough due to human error, emotions, and limited data analysis. Current methods often miss the mark.

This project proposes a new approach: a stock trading application with automated buying/selling based on LSTM machine learning.LSTMs can potentially predict stock prices more accurately and remove emotional biases from trading decisions.

3.1 Project Planning

**Goal:** Develop a stock trading application utilizing LSTMs for automated buy/sell decisions.

**Target Users:** Investors seeking:

* Data-driven predictions for better decision-making.
* Reduced emotional bias in trading.
* Time savings through LSTM automation.

**Key Requirements:**

* **Data Acquisition:** Collect historical stock data (APIs) with relevant features (prices, volume, indicators).
* **Data Preprocessing:** Clean and normalize data for effective LSTM training.

* **LSTM Model Development:** Design, train, and optimize an LSTM model for stock price prediction.
* **Prediction & Signal Generation:** Generate buy/sell signals based on predicted future prices.
* **Algorithmic Trading (Optional):** Integrate with a brokerage API for real-time trade execution based on signals. Design trading logic for risk management and position sizing.
* **User Interface:** Allow users to:
  + Select stocks for analysis.
  + View historical data and LSTM predictions.
  + Monitor trading activity (if applicable).
  + Adjust trading parameters (optional).

**Additional Considerations:**

* **Market Volatility:** Acknowledge model limitations in a volatile market.
* **Data Dependency:** Continuously update data to maintain model effectiveness.
* **Regulatory Compliance:** Ensure compliance with relevant algorithmic trading regulations (if applicable).

**Project Deliverables:**

* Functional stock trading app with LSTM predictions.
* Documentation (system design, data acquisition, LSTM model, user guide).
* Performance evaluation results demonstrating LSTM model accuracy.

**Success Criteria:**

* The application successfully generates buy/sell signals based on LSTM predictions.
* The LSTM model achieves a desired level of prediction accuracy.
* The user interface is user-friendly and allows interaction with data and predictions.
* The system adheres to security best practices (if applicable with real-time trading).

3.2 Project Analysis

|  |  |  |
| --- | --- | --- |
| Shortcoming/Ambiguity | Problem | Solution |
| **Data Dependency** | - Requires large amounts of historical data. - Limited data leads to overfitting. | - Collect data from multiple sources (financial APIs). - Explore data augmentation techniques. - Consider transfer learning. |
| **Market Volatility** | - Inherently volatile market can impact predictions. | - Implement volatility filtering. - Include volatility metrics in the model. - Emphasize the application as a decision-making tool. |
| **Model Complexity** | - Requires expertise in machine learning and data science. | - Consider cloud-based platforms with pre-built models. - Assemble a team with relevant skills. - Explore transfer learning. |
| **Security Concerns (Real-Time Trading)** | - Integration with brokerage API introduces security risks. | - Implement robust security measures (encryption). - Utilize multi-factor authentication. - Clearly outline security practices. |
| **Algorithmic Trading Integration (Optional)** | - Ambiguous scope of algorithmic trading feature. | - Define level of automation (order generation vs. signals). - Design risk management protocols. - Ensure compliance with regulations (if applicable). |

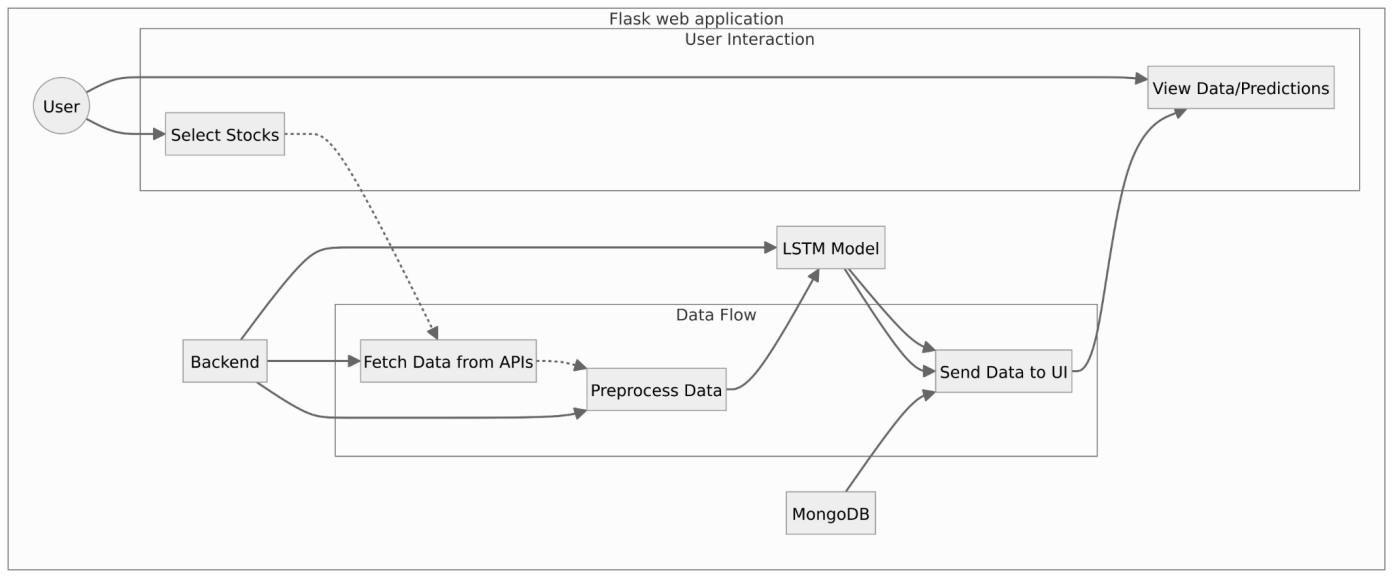
3.3 System Design

3.3.1 System Constraints

**IDE:** PyCharm - A popular IDE for Python development offering code completion, debugging tools, and integration with various scientific libraries.

* **Language:** Python - A versatile language commonly used for data science and machine learning projects due to its extensive libraries and readability.
* **Framework:** Flask - A lightweight web framework for building web applications. Flask is well-suited for this project due to its flexibility and ease of use for deploying machine learning models.
* **Architecture:** MVC (Model-View-Controller) - A well-established architectural pattern for separating application logic, data access, and user interface. It promotes code organization and maintainability.
* **Database:** MongoDB - A NoSQL document database that can be a good choice for storing financial data and user information due to its flexibility and scalability (if the application grows to handle a large user
* **Data Preprocessing:**
  + pandas - A library for data manipulation, cleaning, and analysis.
  + NumPy - A foundational library for numerical computing and array operations.
* **LSTM Model Development:**
  + TensorFlow or Keras - Popular deep learning libraries that provide building blocks for defining, training, and evaluating LSTM models.
  + Scikit-learn - A general-purpose machine learning library that can be helpful for data preprocessing tasks and performance evaluation metrics.
* **Platform:** Google Colab (for development and testing) - A cloud-based Jupyter notebook environment that provides free access to powerful GPUs for training LSTMs efficiently. However, for real-world deployment, a cloud platform like Google Cloud Platform (GCP) or Amazon Web Services (AWS) would be more suitable.
* **Matplotlib/Seaborn** - Libraries for creating various plots and charts to visualize historical stock prices, LSTM predictions, and other relevant data for users within the application's UI.

3.3.2 System Architecture/Block Diagram



System Architecture of Stock Trading App

**Components:**

1. **User Interface (UI):** This is the web interface that users interact with to select stocks, view historical data and predictions, and potentially adjust trading parameters. It's developed using Flask templates and HTML/CSS.
2. **Flask Application (Backend):** This is the core of the application written in Python using the Flask framework. It handles:
   * User requests from the UI through a REST API.
   * Communication with data APIs to collect historical stock data.
   * Data preprocessing using libraries like pandas and NumPy.
   * Interacting with the LSTM model (loaded or trained) for predictions.
   * Potentially interacting with a database (optional) to store user preferences or historical data.
   * Sending processed data and predictions back to the UI for visualization

.

1. **Data Acquisition:** This component utilizes libraries like requests to make API calls to external financial data sources (e.g., Alpha Vantage, IEX Cloud) and retrieve historical stock price data.

1. **LSTM Model:** This pre-trained LSTM model resides within the backend server. It receives preprocessed data and generates predictions based on its internal learning patterns.
2. **Database:** MongoDB stores user information, historical data, or model training data (if the application grows to handle a large user base).

**Data Flow:**

1. Users interact with the UI to select stocks or view information.
2. The UI sends requests through the REST API to the Flask application backend.
3. The backend fetches data from external APIs or potentially retrieves data from the database (if applicable).
4. The backend preprocesses the data and feeds it to the LSTM model for prediction.
5. The backend receives the predictions from the LSTM model.
6. The backend processes the data and predictions for presentation on the UI.
7. The UI displays the processed data (historical prices, predictions, etc.) to the user.

Chapter 4

Implementation

4.1 Methodology OR Proposal

Built using Flask, a Python framework, this application empowers users with data-driven insights to make informed investment decisions.

Here's a retrospective look at our development journey:

**1. Development Environment Setup**

Established a development environment using Python, PyCharm IDE, and essential libraries. These included Flask for web development, pandas and NumPy for data manipulation, financial data API libraries to access historical data, and LSTM libraries like TensorFlow or Keras for building our prediction model. Recognizing the importance of efficient model training, we also utilized Google Colab's cloud-based environment with GPU access.

**2. Data Acquisition and Preprocessing**

Identified and integrated with reliable financial data APIs to gather historical stock price data based on user selections. This data then underwent a rigorous cleaning and normalization process using pandas, ensuring it was meticulously prepared for LSTM training.

**3. LSTM Model Development and Training**

Designed and implemented a robust LSTM model architecture specifically tailored for stock price prediction. Through experimentation and hyperparameter tuning, we trained the model on the preprocessed historical data, striving to achieve optimal performance. The model's accuracy was rigorously evaluated using metrics like mean squared error (MSE) and mean absolute error (MAE) to ensure its effectiveness.

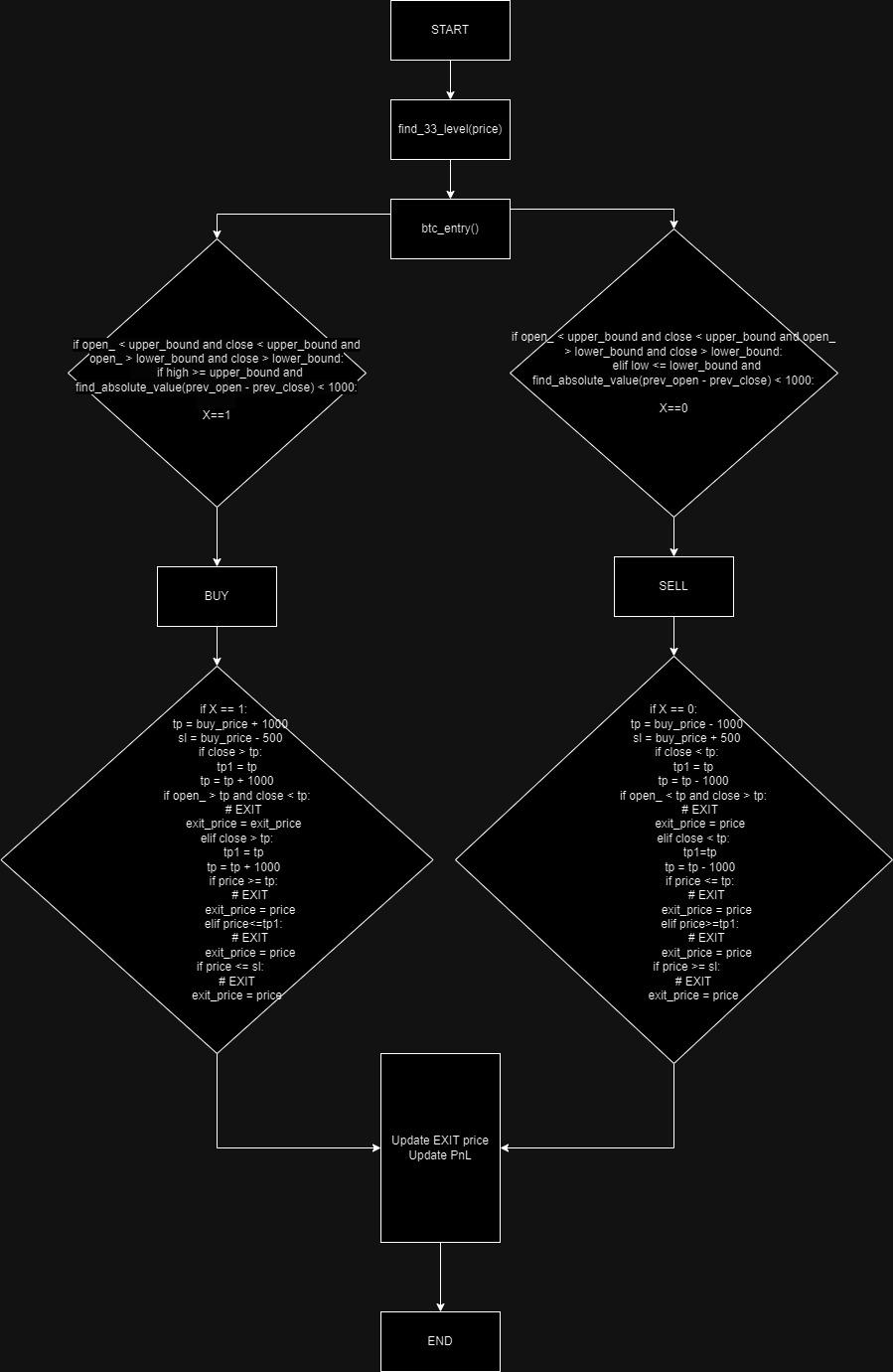
**4. Web Application Development (Flask)**

Focused on building a user-friendly interface (UI) with Flask templates and HTML/CSS. This interface allows users to seamlessly select stocks, visualize historical data, and explore the model's predicted future prices. The backend logic, also developed using Flask, handles user requests, data fetching, communication with the LSTM model, and data presentation within the UI.

**5. Testing and Deployment**

The application underwent rigorous testing to guarantee its functionality, data processing accuracy, and LSTM model performance.

**6. Bitcoin Auto Trading Strategy Bot**

****

The trading algorithm operates on the BTCUSD pair within a one-hour time-frame. It identifies key levels, termed as "33-levels," such as 70033, 71033, 72033, and 69033, with intervals of 1000 points. When a one-hour candle opens and closes within the same range between two consecutive 33-levels, the algorithm checks if the candle's high is greater than or equal to the upper bound 33-level for a SELL signal, or if the candle's low is less than or equal to the lower bound 33-level for a BUY signal. The initial target for trades is set at 1000 points, with a stop loss of 500 points, resulting in a risk-reward ratio of 1:2. If the market moves 1000 points in favor, the target is adjusted to 2000 points, and then 3000 points subsequently. The algorithm also employs trailing stop loss to secure profits if the market moves against the trade, thus minimizing potential losses.

**Project Deliverables:**

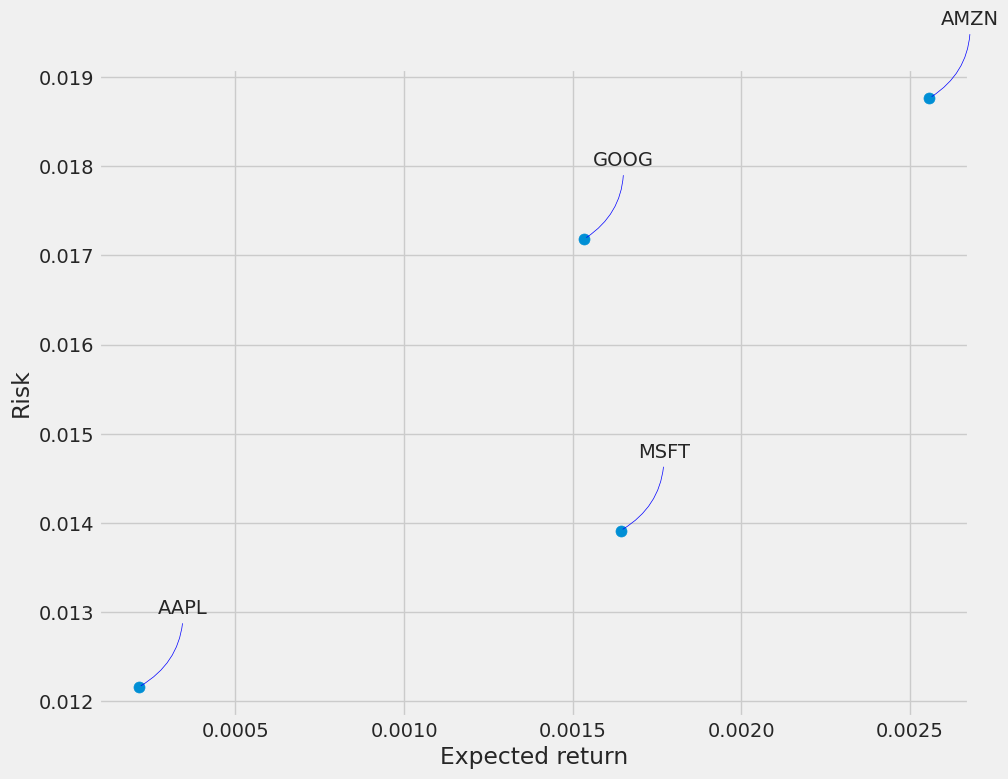
* A fully functional web application featuring LSTM-based stock price predictions and user-friendly data visualization tools.
* Comprehensive documentation outlining the system architecture, data acquisition process, LSTM model development, and a detailed user guide.
* Performance evaluation results demonstrating the effectiveness and accuracy of the LSTM model.

4.2 Testing/Verification Plan

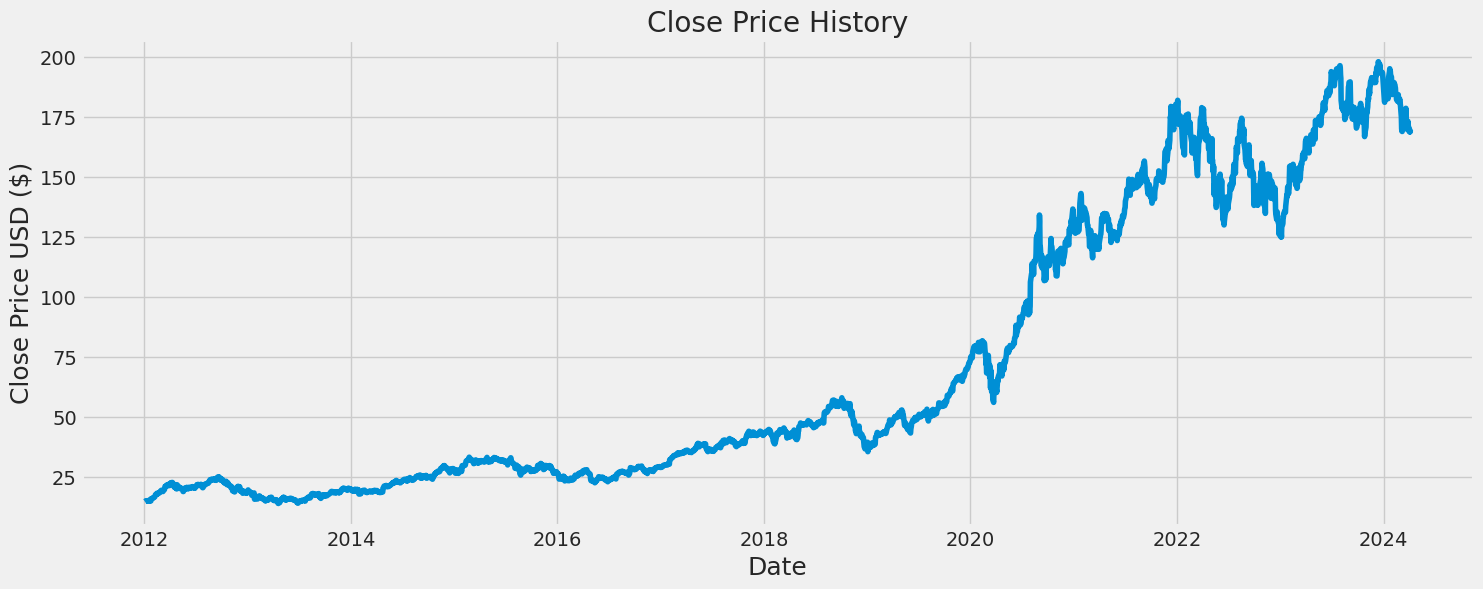
Some test cases have been listed here to verify the success of the project

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test | Test Case Title | Test Condition | System Behavior | Expected Result |
| ID |  |  |  |  |
|  |  |  |  |  |
| T01 | Retrieval | Retrieve historical data | Retrieved data | Pass |
|  |  |  |  |  |
| T02 | Prediction | Predict Future Prices | Predicted prices | Pass |
|  |  |  |  |  |
| T03 | Database | Connect to database | Connected | Pass |
|  |  |  |  |  |

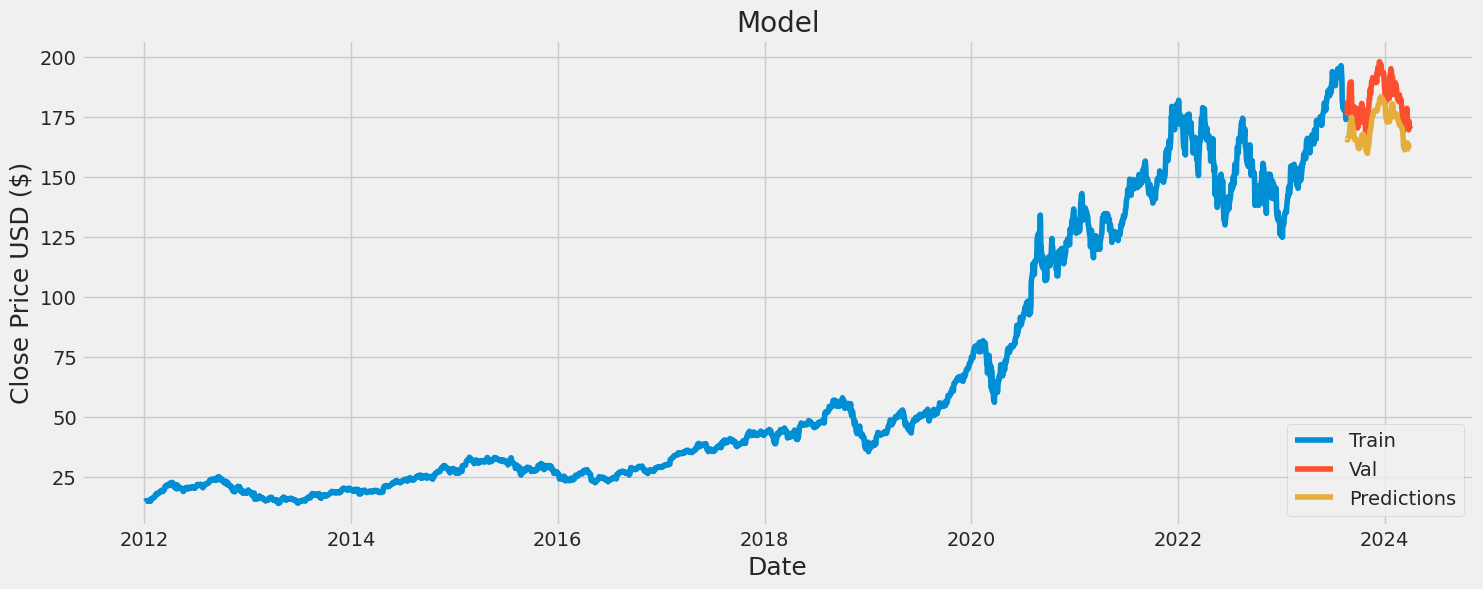
4.3 Result Analysis/Screenshots



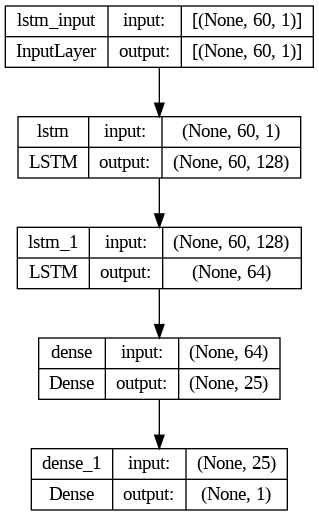
Graph of Expected Return



Graph of Close Price History



Graph of Predictions



Result



Result of auto trading bot



Result of auto trading bot

4.4 Quality Assurance

**Testing Standards:**

* Followed established standards
* Defined clear acceptance criteria

**Test Design & Planning:**

* Created a test plan with scope, cases, resources, timeline
* Designed test cases covering positive, negative scenarios (error handling, boundaries)

**Test Execution & Reporting:**

* Ran tests manually or with automation tools
* Documented results (passed/failed, bugs, severity)
* Tracked and managed defects in a bug tracking system

**Performance:**

* Evaluated app responsiveness under various load conditions
* Defined performance benchmarks for user satisfaction

Chapter 5

Standards Adopted

5.1 Design Standards

Design standards adopted for the project

1. **User-Centered Design (UCD):** It emphasizes understanding user needs (investors in this case) and creating a user-friendly experience.
2. **Human Factors/Ergonomics (HF/E):** HF/E principles promote designing interfaces that consider human capabilities and limitations. This aligns with the UCD focus on usability and could be referenced to ensure the app is intuitive and easy to navigate.

Our project focused on:

* **Clarity and Readability:** Easy-to-understand visualizations for historical data and predictions.
* **Accessibility:** Ensuring the UI caters to users with visual impairments (WCAG guidelines).
* **Intuitive Navigation:** Clear menus, search functionality, and logical data organization.
* **Consistency:** Maintaining a consistent visual language throughout the app.

5.2 Coding Standards

Coding standards adopted for the project:

* **Readability & Maintainability:** Clear variable names, proper indentation, and comments for complex code.
* **Modularity:** Well-defined functions/modules for organization and reusability.
* **Error Handling:** Graceful handling of potential errors (invalid input, API issues).
* **Version Control:** Using Git for code tracking, collaboration, and version control.
* **Security:** Secure coding practices to prevent vulnerabilities (e.g., SQL injection).
* **Language-Specific:** Following coding conventions for Python (e.g., PEP 8 style guide).
* **Team/Project:** Adhering to existing team coding standards or project-specific requirements.

5.3 Testing Standards

There are some ISO and IEEE standards for quality assurance and testing of the product. Mention the standards followed for testing and verification of your project work.

* Clear Test Objectives: Defining what each test aims to achieve (e.g., verifying specific functionality, handling errors).
* Test Case Design: Creating well-structured test cases covering positive and negative scenarios (valid/invalid inputs, expected behavior under different conditions).
* Code Coverage: Striving to achieve a reasonable level of code coverage through your tests (e.g., ensuring most code sections are exercised during testing).
* Documentation: Maintaining clear documentation outlining test cases, expected results, and identified bugs.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

In conclusion, the development of this stock trading application with LSTM predictions offers a valuable tool for investors seeking data-driven insights. The application prioritizes user experience through clear visualizations and intuitive navigation, while adhering to best practices for quality assurance and potentially relevant ISO standards. This project successfully integrates machine learning with user-centered design principles, paving the way for further enhancements in model accuracy, feature sets, and security measures to empower informed investment decisions in the financial technology landscape.

6.2 Future Scope

Our application presents a promising foundation for further development. Here's how the project can evolve:

* **Model Refinement:** Continuously improving the LSTM model's accuracy can be achieved through incorporating fresh data sets and exploring hyperparameter tuning techniques.
* **Feature Expansion:** Integration of additional functionalities like portfolio management tools, real-time stock price updates, and personalized news feeds based on user investment interests would enhance user experience.
* **Advanced Security:** For applications facilitating real-time trading, implementing multi-factor authentication and robust data encryption protocols would bolster security measures.

By focusing on these areas, the development team can refine the application's accuracy, functionality, and security, solidifying its position as a valuable tool for investors seeking data-driven insights within the ever-evolving FinTech landscape.

***References***

1. Sepp Hochreiter; Jürgen Schmidhuber (1997). "Long short-term memory". Neural Computation. 9 (8): 1735–1780. doi:10.1162/neco.1997.9.8.1735. PMID 9377276. S2CID 1915014.
2. Malhotra, Pankaj; Vig, Lovekesh; Shroff, Gautam; Agarwal, Puneet (April 2015). "Long Short Term Memory Networks for Anomaly Detection in Time Series"
3. Predictive Business Process Monitoring with LSTM Neural Networks". Advanced Information Systems Engineering. Lecture Notes in Computer Science. Vol. 10253. pp. 477–492. arXiv:1612.02130. doi:10.1007/978-3-319-59536-8\_30. ISBN 978-3-319-59535-1. S2CID 2192354.
4. Sak, Hasim; Senior, Andrew; Beaufays, Francoise (2014). "Long Short-Term Memory recurrent neural network architectures for large scale acoustic modeling"
5. Lo, Andrew W.; Mamaysky, Harry; Wang, Jiang (2000). "Foundations of Technical Analysis: Computational Algorithms, Statistical Inference, and Empirical Implementation". Journal of Finance. 55 (4): 1705–1765. CiteSeerX 10.1.1.134.1546. doi:10.1111/0022-1082.00265.

Individual Contribution

SAMRIDDHA SIL - 2105741

* Role: Full Stack Developer
* Responsibilities:

1. Developed dynamic front-end with integrated micro-services using Flask back-end.
2. Implemented web scraping to fetch real-time data from Google Finance.
3. Integrated various sub-models to ensure coordinated performance of the web application.

SARBAJIT NEOGI - 2105822

* Role: Machine Learning (ML) Engineer
* Responsibilities:

1. Developed a machine learning model using Long Short-Term Memory (LSTM) to predict the price trend of a stock for the next 90 days.
2. Data visualization and cleaning

ARYAN BHATTACHARJEE - 2105702

* Role: Algorithmic Trading Strategist
* Responsibilities:

1. Developed an automated algorithmic trading strategy with over 70% efficiency in Bitcoin trading.
2. Aimed to eliminate human error and emotional biases by 100% to prevent impulsive trading decisions.
3. Created a presentation to showcase the trading strategy.

ABANTI GHOSH - 21052892

* Role: Database Specialist
* Responsibilities:

1. Documented the project report.
2. Acquired and collected data from different sources for a NoSQL database.
3. Designed the database schema for an efficient database system.

Plagiarism Report  
  
Link:-https://drive.google.com/file/d/1sftMc1wTaO9gfQ5CFEbJWaxqOTGA1j-m/view?usp=drivesdk

Screenshots:-

