A Project Report

on

AI Driven Options Trading

Submitted in partial fulfillment of the requirements of the degree of Bachelor in Engineering by

Abhishek Malaviya (20UF15594CM027)

Nimish Dhumale (20UF16049CM011)

Nishit Kotian (20UF16065CM144)

Jay Vala (21US16318CM019)

Under the guidance of

Prof. Vaishali Chavan

Department of Computer Engineering

Shah and Anchor Kutchhi Engineering College

Chembur, Mumbai – 400088.

2023 – 2024

Mahavir Education Trust's

SHAH & ANCHOR KUTCHHI ENGINEERING COLLEGE

Chembur, Mumbai - 400 088

Department of Computer Engineering

UG Program in Computer Engineering is re-accredited by N. B. A. New Delhi from AY 2022-23 for 3 years up to 30.06.2025. Awarded 'A' Grade (3.16 CGPA) by N. A. A. C. w. e. f. 20.10.2021

CERTIFICATE

This is to certify that the report of the project entitled

AI Driven Options Trading

is a bonafide work of

Abhishek Malaviya (20UF15594CM027)
Nimish Dhumale (20UF16049CM011)
Nishit Kotian (20UF16065CM144)
Jay Vala (21US16318CM019)

submitted to the

UNIVERSITY OF MUMBAI

during semester VIII in partial fulfilment of the requirement for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER ENGINEERING.

Prof. Vaishali Chavan
Guide

Prof. Uday Bhave	Dr. Bhavesh Patel
Head of the Department	Principal

graphicx multirow float

Mahavir Education Trust's

SHAH & ANCHOR KUTCHHI ENGINEERING COLLEGE

Chembur, Mumbai - 400 088

Department of Computer Engineering

UG Program in Computer Engineering is re-accredited by N. B. A. New Delhi from AY 2022-23 for 3 years up to 30.06.2025. Awarded 'A' Grade (3.16 CGPA) by N. A. A. C. w. e. f. 20.10.2021

Attendance Certificate

Date: 04/05/2024

To,
The Principal,
Shah and Anchor Kutchhi Engineering College,
Chembur, Mumbai-88

Subject: Confirmation of Attendance

Respected Sir,
This is to certify that Final year students Abhishek Malaviya , Nimish Dhumale , Nishit Kotian , Jay Vala have duly attended the sessions on the day allotted to them during the period from 17/01/2024 to 22/04/2024 for performing the Project titled AI Driven Options Trading. They were punctual and regular in their attendance. Following is the detailed record of the student's attendance.

Attendance Record:

Date	Abhishek Malaviya	Nimish Dhumale	Nishit Kotian	Jay Vala
	Present / Absent	Present / Absent	Present / Absent	Present / Absent
17/01/2024	Present	Present	Present	Present
01/02/2024	Present	Present	Present	Present
13/02/2024	Present	Present	Present	Present
29/02/2024	Present	Present	Present	Present
25/03/2024	Present	Present	Present	Present
08/04/2024	Present	Present	Present	Present
22/04/2024	Present	Present	Present	Present

Prof. Vaishali Chavan

Approval for Project Report for B. E. Semester VIII

This project report entitled AI Driven Options Trading by Abhishek Malaviya , Nimish Dhumale , Nishit Kotian and Jay Vala is approved for semester VIII in partial fulfilment of the requirement for the award of the degree of Bachelor of Engineering.

Examiners

1. ______________________________

2. ______________________________

Guide

1. ______________________________

2. ______________________________

Date: 04/05/2024

Place: Mumbai

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Name of the Student	**Roll No.**	**Signature**
Abhishek Malaviya	20UF15594CM027	
Nimish Dhumale	20UF16049CM011	
Nishit Kotian	20UF16065CM144	
Jay Vala	21US16318CM019	

Date: 04/05/2024

Place: Mumbai

Acknowledgement

We would like to express our sincere gratitude to all those who have supported and guided us throughout the process of conducting this report on AI Driven Options Trading. This endeavor would not have been possible without their valuable contributions and assistance.

We are thankful to our college Shah and Anchor Kutchhi Engineering College for considering our project and extending help at all stages needed during our work of collecting information regarding the project.

We are deeply indebted to our Principal Dr. Bhavesh Patel and Head of the Computer Engineering Department Prof. Uday Bhave for giving us this valuable opportunity to do this project. We express our hearty thanks to them for their assistance without which it would have been difficult in finishing this project synopsis and project review successfully.

We take this opportunity to express our profound gratitude and deep regards to our guide Prof. Vaishali Chavan for her exemplary guidance, monitoring and constant encouragement throughout the course of this project.

This work would not have been possible without the collective efforts of these individuals and organizations. While any shortcomings in this report are solely our responsibility, their contributions have significantly enriched its content.

Abstract

The AI-Driven Options Trading (AI DOT) project represents a cutting-edge endeavor aimed at revolutionizing the world of options trading. With a strong focus on automation and intelligent decision-making, the project leverages advanced prediction algorithms and real-time market data analysis to empower traders. This abstract provides a concise overview of the project's objectives, key features, and the transformative potential it holds for traders seeking to optimize their strategies and navigate the complex landscape of options trading.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The world of financial markets is characterized by constant evolution and the pursuit of more effective, data-driven trading strategies. In this dynamic landscape, the AI-Driven Options Trading Bot (AI-DOT) project emerges as a beacon of innovation and sophistication. This project has its roots in extensive research, culminating in the selection of advanced prediction algorithms derived from a range of trading applications. These algorithms serve as the bedrock for an intelligent trading bot designed to automate options trading.
This project aims to develop a trading strategy using artificial neural networks (ANN) and recurrent neural networks (RNN). By leveraging machine learning models, we seek to predict market movements and generate trading signals for options trading. The project involves data collection, preprocessing, model training, signal generation, and potential real-time trading execution. Through this endeavor, we aim to explore the effectiveness of using advanced neural network models for financial forecasting and decision-making in the options market.

1.1 Background

AI-based options trading involves the use of artificial intelligence and machine learning algorithms to make automated and data-driven decisions when buying or selling options in financial markets. These systems analyze vast amounts of historical and real-time data to identify patterns, trends, and potential trading opportunities, enabling traders to make more informed and efficient decisions while managing risk. AI-based options trading has gained popularity for its ability to quickly process information and adapt to changing market conditions, potentially improving trading performance and reducing human bias in decision-making.

1.2 Motivation

Potential for Financial Independence: Algorithmic options trading can provide us with the opportunity to achieve financial independence and security. Success in this field can lead to significant profits and wealth generation.
Risk Management: Developing algorithms involves understanding risk management thoroughly. It can teach us valuable skills that extend beyond trading, including risk assessment and mitigation.
Independence: we have the autonomy to develop our strategies and execute them without being dependent on others.
Potential for Passive Income: Once our algorithms are set up and running smoothly, they can generate income with relatively little ongoing effort. This can free up our time for other pursuits or projects.

Chapter 2

Literature Review

Sr No	Author/Title/Year	Idea Presented in the Paper	Remarks
1.	Pramod B.; Mallikarjuna Shastry P. Stock Price Prediction Using LSTM. REVA University, Bengaluru. Appliedscience, 2020[1]	The paper presents a method for predicting stock prices LSTM. This paper provides a proposed system which includes collection of historical data, preprocessing, feature extraction, splitting data into training and testing data, get trained data and predict stock price.	This document explores using a proper schematics of the project to ensure data consistency. It outlines the process, shares results, and emphasizes the importance of the proper architecture of project.
2.	Sayandeep Ghosh; Sudhanshu Kumar; Atharva Deshmukh; Akshay Kurve; Dr.Rashmi Welekar. Options Trading using Artificial Neural Network and Algorithmic Trading. International Journal of Next-Generation Computing - Special issue, Vol. 13, No. 5. Research Gate — November 2022.[2]	The document talks about two key tools for stock market prediction: the Relative Strength Index (RSI) and Moving Average Convergence and Divergence (MACD). RSI helps spot overbought or oversold stocks, while MACD tracks market trends.	These tools, RSI and MACD, are valuable in stock market analysis as they aid in assessing stock conditions and market trends, enabling better trading decisions.

Sr No	Author/Title/Year	Idea Presented in the Paper	Remarks
3.	Dehong Liu;Yucong Liang;Lili Zhang;Peter Lung;Rizwan Ullah. Implied volatility forecast and option trading strategy. International Review of Economics and Finance (2020). Elsevier Inc October 2020.[3]	The paper proposes a methodology for forecasting implied volatility and developing option trading strategies, emphasizing the importance of accurate volatility prediction for effective options trading.	The paper provides valuable insights into the significance of implied volatility forecasting in option trading, offering potential enhancements to trading strategies through improved volatility prediction.
4.	Kavinnilaa J;Hemalatha E; Minu susan Jacob; Dhanalakshmi R. Stock Price Prediction Based on LSTM Deep Learning Model. KCG College of Technology, Karapakkam, Chennai, TAMILNADU, IEEE 2021[4]	This paper suggests a way to use LSTM for stock price prediction. It shows how this method can help make profits in real markets. This paper uses Rectified Linear Unit (ReLU) as an activation function to solve exploding gradients problem. MAE, RMSE and MSE values are calculated to evaluate the model's performance.	The paper's idea of using LSTM and ReLU for stock price prediction is quite promising. It suggests a way to create LSTM and incorporate ReLU activation for increasing model's accuracy , which could be valuable for our project.
5.	Ananda Chatterjee; Hrisav Bhowmick; Jaydip Sen. Stock Price Prediction Using Time Series, Econometric, Machine Learning, and Deep Learning Models. IEEE December 2021 [5]	This paper demonstrates a set of time series, econometric, and various learning-based models for stock price prediction. One time series model (Holt-Winters Exponential Smoothing), one econometric model (ARIMA), two machine Learning models (Random Forest and MARS), and two deep learning-based models (simple RNN and LSTM) have been included in this paper.	This paper provides a comprehensive analysis of various models for stock price prediction, including time series, econometric, machine learning, and deep learning approaches. It offers valuable insights into the strengths and limitations of each method, contributing to the ongoing discussion on effective forecasting techniques in finance.

Sr No	**Author/Title/Year**	**Idea Presented in the Paper**	**Remarks**
6.	Hum Nath Bhandari; Binod Rimal; Nawa Raj Pokhrel; Ramchandra Rimal; Keshab R. Dahal; Rajendra K.C. Khatri. Predicting stock market index using LSTM. IEEE May 2022 [6]	The paper presents a predictive model for stock market indices using Long Short-Term Memory (LSTM) networks, with a focus on forecasting accuracy and performance evaluation, demonstrating potential for improved investment decision-making.	The authors concluded that the LSTM network could effectively extract meaningful information from the financial time series data. Based on prediction accuracy and daily returns after transaction costs, LSTM outperforms random forests, standard deep networks, and logistic regression.
7.	Mr.P.Ramu;Ms.A.Vani Priya;Ms.D.Roopa Sree ;Ms.S.Sankeerthana. Stock Price Prediction Using Time Series. IRJET, June 2021, [7]	The paper ”Stock Price Prediction Using Time Series” proposes a method for forecasting stock prices through the application of time series analysis, leveraging historical data and predictive modeling techniques, with potential implications for investors and financial analysts seeking to enhance decision-making processes.	Through the application of time series analysis, the study showcases its effectiveness in predicting stock prices, thereby providing valuable insights that can aid investors and financial analysts in making informed decisions.

Sr No	Author/Title/Year	Idea Presented in the Paper	Remarks
8.	XIAOJIAN WENG; XUDONG LIN; SHUAIBIN ZHAO. STOCK PRICE PREDICTION BASED ON LSTM AND BERT. arXiv September 2021 [8]	The paper explores a novel approach merging the strengths of Long Short-Term Memory (LSTM) and Bidirectional Encoder Representations from Transformers (BERT) models for predicting stock prices, utilizing LSTM to capture temporal dependencies and BERT to encode textual information, thereby enhancing prediction accuracy by integrating both numerical and textual data sources.	The paper proposes a novel approach combining Long Short-Term Memory (LSTM) and Bidirectional Encoder Representations from Transformers (BERT) models for stock price prediction, demonstrating improved accuracy and robustness compared to traditional methods, by leveraging the sequential patterns captured by LSTM and the contextual information extracted by BERT.
9.	Muyang Ge; Shen Zhou; Shijun Luo; Boping Tian. 3D Tensor-based Deep Learning Models for Predicting Option Price. arXiv September 2021 [9]	The paper proposes a novel approach utilizing 3D tensor-based deep learning models to predict option prices, integrating various features and factors into a comprehensive framework to enhance accuracy and efficiency in financial derivatives pricing.	The paper proposes novel 3D tensor-based deep learning models for predicting option prices, offering a promising approach that integrates multi-dimensional data representations to enhance accuracy and efficiency in financial option pricing tasks.

Sr No	Author/Title/Year	Idea Presented in the Paper	Remarks
10.	Armin Lawi; Hendra Mesra; Supri Amir. Implementation of Long Short-Term Memory and Gated Recurrent Units on grouped time-series data to predict stock prices accurately. Researchgate July 2022 [10]	The paper proposes the utilization of Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) on grouped time-series data as a predictive model for accurately forecasting stock prices, aiming to enhance predictive performance by leveraging the capabilities of these recurrent neural network architectures.	The paper demonstrates the successful application of Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) models on grouped time-series data, achieving accurate predictions of stock prices, thus highlighting the effectiveness of these techniques in capturing temporal dependencies and patterns within financial data.

Table 2.1: Literature Review

2.1 Survey of Existing system

AlgoTrader Industry: Financial Technology (Fintech) Overview: AlgoTrader is a leading fintech company specializing in algorithmic trading software. Their platform integrates artificial intelligence and machine learning to provide options traders with advanced tools for risk management, analytics, and strategy customization.

Deep Blue Capital Industry: Hedge Fund (Quantitative Trading) Overview: Deep Blue Capital is a quantitative hedge fund that employs cutting-edge AI and machine learning models for options trading. Renowned for its expertise in volatility-based strategies, the firm excels in adapting to evolving market conditions and exploiting inefficiencies. Deep Blue Capital's strategies aim to capture alpha in both bullish and bearish markets through data-driven approaches.

Qplum Industry: Fintech/Asset Management Overview: Qplum is a fintech company offering AI-powered portfolio management services and algorithmic trading solutions. Transparency and risk management are core principles, with the company providing investors insights into their portfolios. Qplum's AI models are designed to optimize investment decisions, including options trading, while aligning with investors' risk preferences and financial goals.

Citadel Securities Industry: Financial Services (Market Making) Overview: Citadel Securities is a prominent market maker and liquidity provider with a strong presence in options markets. Leveraging advanced AI algorithms, the company enhances market liquidity and efficiency. Citadel Securities is known for maintaining competitive spreads, executing high volumes of options trades, and contributing to market stability through high-frequency trading strategies.

2.2 Limitation of Existing system or research gap

1. AlgoTrader Industry: Financial Technology (Fintech) Complexity: The software may be complex for novice traders to fully utilize, requiring a learning curve. Market Risk: While it can assist in risk management, it cannot entirely eliminate market risks inherent in options trading. Data Quality: The accuracy and quality of the AI predictions are highly dependent on the quality of the data fed into the system.

2. Deep Blue Capital Industry: Hedge Fund (Quantitative Trading) Market Volatility: Extreme market volatility can challenge AI models, potentially leading to unexpected outcomes. Overfitting: Developing AI models that perform well historically but fail in real-world scenarios due to overfitting is a risk. Data Availability: Access to high-quality data is crucial, and limitations in data sources can impact model performance.

3. Qplum Industry: Fintech/Asset Management Market Risk: Even with AI, investments are subject to market fluctuations, and losses can occur. Model Accuracy: The accuracy of AI models depends on historical data, and unforeseen events can impact their performance. Complexity: AI-powered investing can be complex for some investors, making it important to provide user-friendly interfaces.

4. Citadel Securities Industry: Financial Services (Market Making) Market Liquidity: In illiquid markets, maintaining competitive spreads and executing trades can be challenging. Technological Risks: Dependency on technology introduces risks related to system outages and cyber threats. Market Events: Black swan events and extreme market conditions can stress AI models and trading strategies.

2.3 Problem Statement and Objective

Problem Statement

The increasing complexity and volatility of financial markets, coupled with the growing demand for options trading strategies, present a need for a robust options trading software.

Market Data Integration: The software should seamlessly integrate with various financial data sources to provide real-time market data, including options prices, underlying assets, volatility metrics, and other relevant information.
Use of Neural Networks: By leveraging machine learning models and neural networks, we seek to predict market movements and generate trading signals for options trading.
Risk Management: Implement risk management tools by increasing accuracy of the model to help traders assess and mitigate risks associated with their options positions.

Objective

- Objective 1: Develop a Profitable Options Trading Algorithm To design and implement an algorithm that uses historical options market data, technical indicators, and risk management strategies to consistently generate profitable trading signals. To optimize the algorithm's parameters and trading strategies through backtesting.
- Objective 2: Incorporate a way to integrate this bot with a broker using its API.

2.4 Scope

AI Driven Options Trading aims to develop an options trading software for both individual and institutional traders. The software will offer advanced trading features and analytics to help users trade options effectively.

Chapter 3

Software Requirement Specification

3.1 Introduction

Purpose

1. Reuse Objectives:
- Modular Design: Create a modular and well-documented code-base that allows for easy reuse of algorithmic components in different trading strategies or systems.
- Algorithm Libraries: Use libraries or frameworks for commonly used trading algorithms, risk management strategies, and data analysis tools that can be reused in future projects.
- Version Control: Implement version control systems to track and manage changes to algorithms and strategies for efficient reuse.
- Knowledge Sharing: Encourage knowledge sharing and collaboration among team members to ensure that expertise is transferred and available for reuse.

Document Conventions

This SRS Document follows standard conventions that are followed when writing a SRS. All of the references are cited and ensured that they are cited correctly

Intended Audience and Reading Suggestions

1. Developers: This SRS provides developers with a comprehensive understanding of the software requirements, functional specifications, and design constraints related to Algorithm-Based Options Trading.
2. Financial Analysts and Traders: These end-users of the Algorithm-Based Options Trading system can gain insights into how the software facilitates options trading.

Product Scope

The AI-Driven Options Trading Bot is software designed for automating options trading. It integrates advanced algorithms like ANN and RNN, combines the output of both models to provide a more accurate buy/sell signal. Users can configure trading parameters and the bot will automatically execute trades. The software caters to traders with varying levels of experience, from novices to professionals, and aims to reduce downside risk and volatility in existing long equity positions. This academic project scope focuses on research-driven development and adherence to academic guidelines.

3.2 Overall Description

Product Perspective

The project entails the development of an AI-Driven Options Trading platform, featuring two primary components: a sophisticated trading bot driven by artificial intelligence for automated and optimized options trading, and a broker integration with API to execute trades in the real market. The project's primary goal is to offer an advanced and comprehensive options trading solution, underpinned by AI and real-time data visualization, enhancing users' trading capabilities and decision-making processes.

Product Functions

- Automated Options Trading: Enables users to automate options trading using AI-driven algorithms.
- Strategy Optimization: Optimize trading strategies to enhance returns and risk management.
- Broker Integration: Provide real-time trade execution by integrating broker using its API.

User Classes and Characteristics

1. Traders:
- High-frequency traders and professionals.
- Casual traders with moderate experience.
- Beginners in options trading.

Operating Environment

Hardware Platform: The software is designed to run on standard x86-64 and ARM-based hardware, ensuring compatibility with a wide range of modern computing systems.
Operating System: The software is platform-free and can be deployed on multiple operating systems, including Windows 10 and later, macOS, and various Linux distributions

Software Components:
- Python 3.7 or later: For core application logic and algorithm implementations.
- Data analysis and visualization libraries: Used to process market data and plot buy/sell signals.

Design and Implementation Constraints

The software must integrate seamlessly with external financial data sources and brokers in compliance with academic guidelines, ensuring compatibility and interoperability within the academic context.

User Documentation

We will need to install Python and its required libraries. Then we need to install the required libraries for the AI Driven Options Trading project.
Once you have installed the necessary programs, you can proceed with setting up the trading bot. You will need to configure your API keys and settings by signing up on the required broker app.

Assumptions and Dependencies

1. The current stock market conditions may impact the profitability of the trading bot. A sustained bull market could result in increased profits, while a bear market may result in losses.
2. The availability and performance of third-party services like yFinance and TensorFlow could affect the development of the trading bot. Poor service performance may delay the development process or require the use of alternative services.
3. The quality and reliability of the stock market data used by the trading bot may impact its accuracy and performance. Factors such as data latency, inaccuracies, and data exclusions could influence the bot's behavior and performance.

3.3 External Interface Requirements

User Interfaces

The options trading bot will provide signals to buy/sell and execute trades automatically. It will also provide visual charts and plots to verify the models accuracy.

1. Trading Bot: The Python-based Algorithmic Trading Bot specializes in automated trading of put and call options. Focused on risk mitigation and increased profit probabilities, the bot intelligently buys and sells put and call options in response to market

conditions by identifying potential downward or upward trends.

2. Visual Charts: The Visual Charts feature in our project presents a dynamic visual representation of options market data by plotting various graphs. This graphical interface offers a comprehensive view of data consistency, models accuracy and Real vs Predicted values.

Hardware Interfaces

Logical Interface:
User Interface (UI): The UI will be the broker app that users can access through a web browser or mobile app.
Application Programming Interface (API): The API will provide a way for the trading bot to communicate with the real market.

Physical Interface: Trading bot will be the raw code interface which will be hidden behind the broker application.

Software Interfaces

Operating Systems: Windows 10/11, MacOS, Linux etc. are versatile and stable operating systems for hosting the web application server and providing a reliable environment for the trading bot.

Tools: Libraries:

- Python 3.7 or later
- NumPy
- Pandas
- TensorFlow
- yFinance
- matplot Libraries
- SkiKit Learn
- ta
- datetime

Communications Interfaces

RESTful API provides a reliable and scalable interface for communication between the Broker app and the backend.

3.4 System Features

The System contains of 2 main features:

1. Trading Bot
2. Visual Charts

3.4.1 Trading Bot

Description and Priority

The bot uses predictive models like Artificial Neural Network(ANN) and Recurrent Neural Network(RNN) mainly Long Short Term Memory(LSTM) for predicting and generating trade signals like buying and selling of call/put options.
Priority: High
Priority Component Ratings:

- Benefit: 8
- Penalty: 2
- Cost: 7
- Risk: 6

Stimulus/Response Sequences

User Actions: User only integrates his/her broker app with the bot using API and the trades are automatically executed by the bot.
System Responses: The bot executes trades buy analysing the historical data and predicting stock prices using the ANN and RNN models

Functional Requirements

REQ-1. Market Data Feed: The market data feed should provide data points like stock price, trading volume, open, high, low,impliedVolatility, option chain and historical price data.

REQ-2. Trading Strategies: The system should support multiple trading strategies such as intraday trading, etc.

REQ-3. Backtesting and Simulation: The trading bot should provide accurate results after backtesting and simulating trades.

3.4.2 Trade Execution

3.5 Other Nonfunctional Requirements

Performance Requirements

Data Loading Requirements: The bot should be able to load real-time data for options contracts in under 50 milliseconds.
Data Processing Requirements: The bot should be able to identify trading opportunities in under 50 milliseconds.
Trading Execution Requirements:

- The bot should be able to submit orders to a brokerage platform in under 50 milliseconds.
- The bot should be able to receive order confirmations from a brokerage platform in under 100 milliseconds.

Safety Requirements

Regular security audits must be conducted to identify and address vulnerabilities promptly. Mitigating the impact of potential DDoS attacks, ensuring the uninterrupted availability of services to users.

Security Requirements

The system maintains data integrity and consistency through validation mechanisms, constant monitoring, and robust backup procedures, ensuring accurate and reliable data. Additionally, it fortifies against malicious code by conducting regular vulnerability scans, utilizing Content Delivery Network (CDN) protection against DDoS attacks, and promptly addressing identified threats.

Software Quality Attributes

1. Adaptability
   Specific: The options trading bot should be able to adapt to changing market conditions. Quantifiable: The bot should be able to handle changes in market data, trading strategies.
2. Availability
   Specific: The options trading bot should be available 24/7 with minimal downtime.
3. Correctness
   Specific: The options trading bot should generate accurate trading signals and execute trades correctly. Quantifiable: The bot should achieve a trading accuracy of at least 85

percent.

4. Reliability
   Specific: The options trading bot should be reliable and prevent crashes or errors that could lead to financial losses for users. Quantifiable: The bot should have a low error rate.
5. Robustness
   Specific: The options trading bot should be able to handle unexpected events and market conditions without crashing or causing erroneous trades.

Business Rules

User Roles and Permissions

1. Admin: Has full access to all functionalities of the bot, including creating and modifying the bot, configuring trading strategies, and monitoring bot performance.
2. Standard User: Can create and manage their own trading accounts, select trading markets, and view their bot's performance and transaction history on the broker app.

User Authorization and Access Control

1. Strong authentication: All users must register with valid email addresses and strong passwords to ensure secure login on their chosen trading app.

Trading Strategy Management

1. Backtesting capabilities: Provide backtesting functionality to simulate trading strategies against historical market data, allowing users to evaluate their performance and optimize their settings.
2. Real-time monitoring: Continuously monitor the market and adapt trading strategies in real time based on changing market conditions.

Trade Execution and Risk Management

1. Secure trade execution: Securely connect to authorized trading platforms to execute trades based on the bot's recommendations, ensuring that orders are placed accurately and efficiently.

Performance Measurement and Reporting

1. Historical performance data: Store historical trading data to track the bot's performance over time, allowing users to assess its effectiveness and make informed decisions.
2. Customizable charting and analytics: Provide customizable charting and analytics tools to visualize trading data, identify trends, and evaluate the bot's performance against benchmarks.

3.6 Other Requirements

- Documentation standards: Adhere to industry-standard documentation practices to ensure clear, consistent, and up-to-date documentation for developers and users alike.
- Version control: Utilize a version control system to track changes, maintain code history, and facilitate collaboration among developers.
- Performance Optimization: Optimize the bot's algorithms and code to minimize latency, improve response times, and handle large datasets efficiently.
- Load Testing and Scalability: Perform load testing to simulate high traffic scenarios and ensure the bot can handle increased demand without compromising performance or stability.

Chapter 4

Project Scheduling and Planning

4.1 Project Scheduling

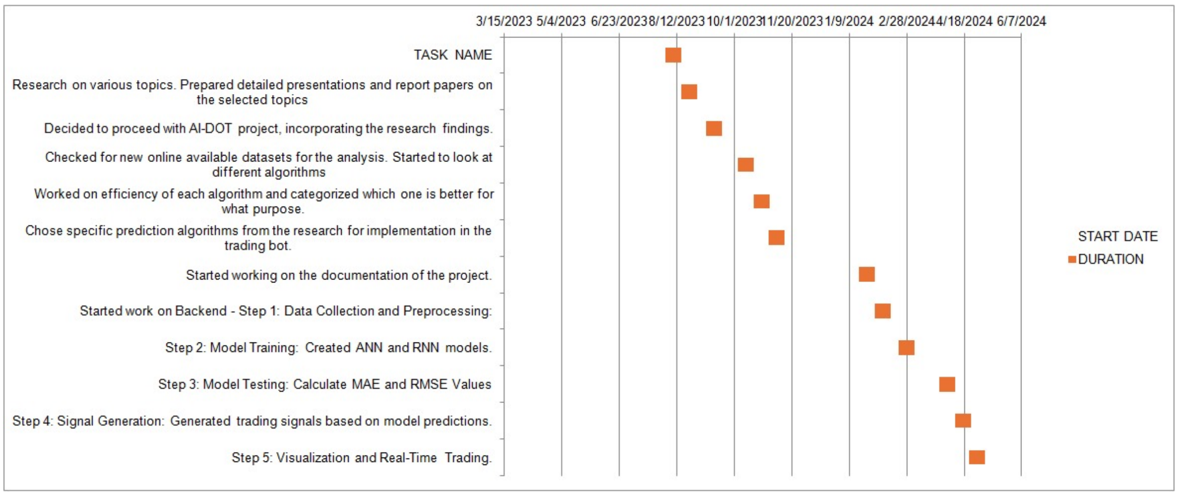


Figure 4.1: Project Scheduling

SR. No.	**Week No.**	**Project Work Done / Progress Achieved**	**Start Date**	**Deadline**
1	**1& 2**	Research on various topics. Prepared detailed presentations and report papers on the selected topics	**02/08/23**	**20/09/23**
2	**2 & 3**	Decided to proceed with AIDOT project, incorporating the research findings	**20/09/23**	**14/10/23**
3	**3 & 4**	Designed and developed a rough frontend interface. Enabled live options charts and graph trends for real-time data analysis.	**14/10/23**	**04/11/23**
4	**4 & 5**	Chose specific prediction algorithms from the research for implementation in the trading bot.	**04/11/23**	**18/11/23**
5	**5 & 6**	Started working on the documentation of the project	**18/11/23**	**02/12/23**

SR. No.	**Week No.**	**Project Work Done / Progress Achieved**	**Start Date**	**Deadline**
6	**7 & 8**	Step 1: Data Collection and Preprocessing Gathered historical stock data and options data	**17/01/24**	**01/02/24**
7	**8 & 9**	Step 2: Model Training: Created ANN and RNN models. Trained models on standardized data.	**01/02/24**	**13/02/24**
8	**9 & 10**	Step 3: Model Evaluation: Created Calculated evaluation metrics (MAE, RMSE).	**29/02/24**	**25/03/24**
9	**10 & 11**	Step 4: Signal Generation: Generated trading signals based on model predictions	**08/04/24**	**15/04/24**
10	**11 & 12**	Step 5: Visualization and Real-Time Trading: Plotted predicted entry/exit points on stock price chart. Implemented real-time trading functionality with broker's API.	**15/04/24**	**22/04/24**

Table 4.1: Project Scheduling

4.2 Planning

1. Set Clear Goals and Objectives: Establish specific and measurable goals for the project, such as achieving a certain rate of return, minimizing risk, or outperforming a benchmark index.
2. Algorithm Development: Specify the algorithms and models that the trading bot will use for decision-making and ensure that these algorithms are well-documented and thoroughly tested.
3. Data Acquisition and Integration: The sources of data required for the bot's operation, such as market data, news feeds, and economic indicators. Establish a plan for data acquisition and integration.
4. Backtesting and Simulation: Creating a plan for backtesting the trading strategies to evaluate their historical performance and simulate how they would have performed in past market conditions.
5. Testing and Quality Assurance: Establish a comprehensive testing plan to identify and fix any bugs or issues in the trading bot's code. Ensure that the bot is thoroughly tested before deployment.
6. Deployment Strategy: Develop a deployment plan that outlines how the trading bot will be launched into a production environment. Consider factors like monitoring, scaling, and failover mechanisms.

Chapter 5

Proposed System

5.1 Analysis/Framework/ Algorithm

Automated Trading: Develop a trading bot capable of executing options trades automatically based on predefined strategies.
Strategy Implementation: Incorporate strategies derived from research papers that aim to optimize trading performance, manage risk, and improve profitability.
Real-time Data: Integrate real-time market data and analytics into the bot to enhance decision-making capabilities.

5.2 Details of Hardware & Software

Hardware Requirements:

1. Computer: We will need a computer with sufficient processing power and memory to handle the complex computations.
2. Data Feeds: Access to high-quality, low-latency market data feeds is crucial. Redundant hardware can ensure uninterrupted operation.
3. Internet Connection: A high-speed, stable internet connection is vital for real-time data transmission and order execution.

Software Requirements:

1. Programming Languages: Python, QuantConnect API.
2. Trading APIs: user will specify which trading platform to integrate the bot.
3. Machine Learning Libraries: Libraries like yFinance, scikit-learn, TensorFlow and Keras are used for this purpose.
4. Backtesting Framework: Develop or use a backtesting framework to test our trading strategy against historical data.

5. Real-time Data Feeds: Integrate real-time market data feeds to make informed decisions.

5.3 Design Details

Our algorithm-based options trading system aims to provide traders with a powerful platform that utilizes advanced algorithms to make data-driven decisions and optimize trading strategies. This system consists of several interconnected modules, each serving a specific purpose:

1. Trading Algorithm: The core of the system are the Neural Networks. This model is designed to analyze market data, including stock prices, options tickers, trading volumes, and various technical indicators.
2. Data Feed Integration: To obtain real-time market data, our system integrates with yFinance. These data feeds include stock, options prices, impliedVolatility and other dataframes.
3. Trade Execution: The trade execution module is responsible for executing orders generated by the trading algorithm. It interfaces with various brokerage platforms and APIs to execute orders in real time.
4. User Interface: The user interface is brokers trading platform web application that allows traders to interact with the system. It provides access to real-time market data, trading insights, and performance analytics.
5. Back-Testing and Simulation: Traders can test their trading strategies using historical market data to assess their potential performance.

5.4 Methodology

1. Data Collection: Identify and select appropriate data sources, such as real-time market data, historical market data from yFinance. Importing options data including impliedVolatility, call and puts for a specified expiration date.
2. Data Preprocessing and plotting: Gathering the data from yFinance and Preprocessing the data to remove any inconsistency and null values in the data. Plotting the preprocessed data to verify if any outliers or noise is still present in the data. Computing moving averages (Short and Long) to identify trends.
3. Model Training: Splitting the data into training (X) and testing (Y) sets. Standardise the data using StandardScaler. Implementing Artificial Neural Network and Recurrent Neural Network (LSTM). Compiling models with adams optimiser, ReLU Activation and loss functions L1-L2 Regularisers to prevent overfitting and reduce generalisation errors. Utilizing callbacks such as Early Stopping and ReduceLROnPlateau to prevent overfitting. Learning rate is set to 0.0001.

4. Model Testing: Reshaping data for ANN and RNN model training. It is necessary to reshape the input data to match the expected input shape of the ANN and RNN model. RNN expect data in form of sequences. Handling NaN values to prevent any issues during training.
5. Model Validation: ANN Model's Epoch cycle is set to 600 as ANN model takes more cycles to converge. RNN Model's Epoch cycle is set to 200 RNN model takes less cycles to converge. Can be adjusted based on data and models complexity. Calculated MAE and RMSE values to check how well the model is predicting by finding the difference between actual values and predicted values. Lower the values the better prediction of models.
6. Plotting Learning Curve: Plotting Learning curve to check for Bias and Varience. Plotting Histograms of Actual and Predicted values to compare how well the model is predicting values graphically
7. Generating Trading Signals: Generating trading signals based on the predictions of the ANN and RNN models. Defining Option trading strategy(buying and selling of Call or Put) based on generated signals. Plotting the predicted entry and exit points on the stock price chart.
8. Backtesting and Optimization Analyze backtest results to identify potential weaknesses and refine trading strategies. Use historical data to simulate real-world scenarios and evaluate the body's ability to handle various market conditions.
9. Real Time Trading: Implementing function for live trading based on generated signals. Integrating with brokers/trading platforms API for real-time trade execution of orders.

5.5 Images References

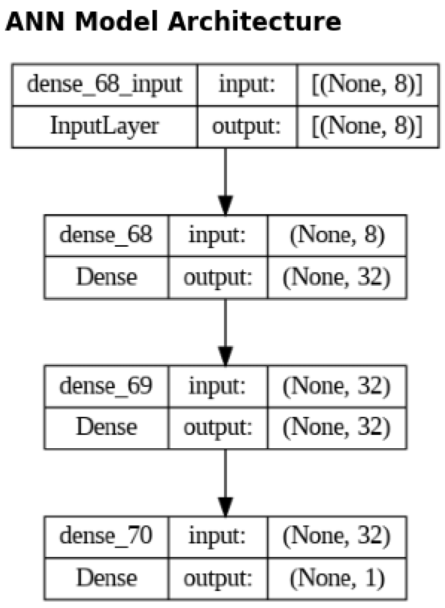

ANN Model Architecture
dense_68_input | input: | [(None, 8)]
InputLayer | output: | [(None, 8)]
dense_68 | input: | (None, 8)
Dense | output: | (None, 32)
dense_69 | input: | (None, 32)
Dense | output: | (None, 32)
dense_70 | input: | (None, 32)
Dense | output: | (None, 1)

Figure 5.1: ANN Model Architecture

5.6 Images References

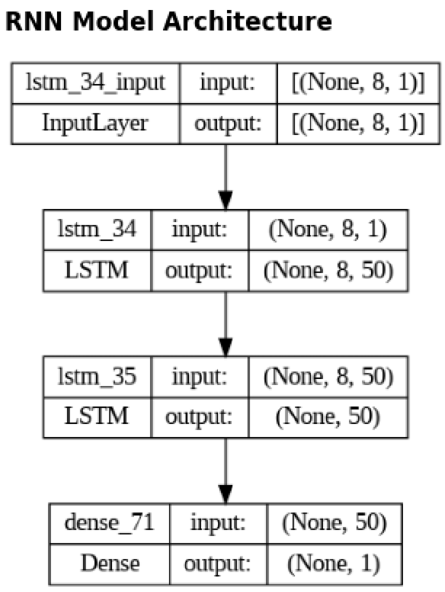

RNN Model Architecture
lstm_34_input | input: | [(None, 8, 1)]
InputLayer | output: | [(None, 8, 1)]
lstm_34 | input: | (None, 8, 1)
LSTM | output: | (None, 8, 50)
lstm_35 | input: | (None, 8, 50)
LSTM | output: | (None, 50)
dense_71 | input: | (None, 50)
Dense | output: | (None, 1)

Figure 5.2: RNN Model Architecture

Chapter 6

Implementation Details

6.1 Images References

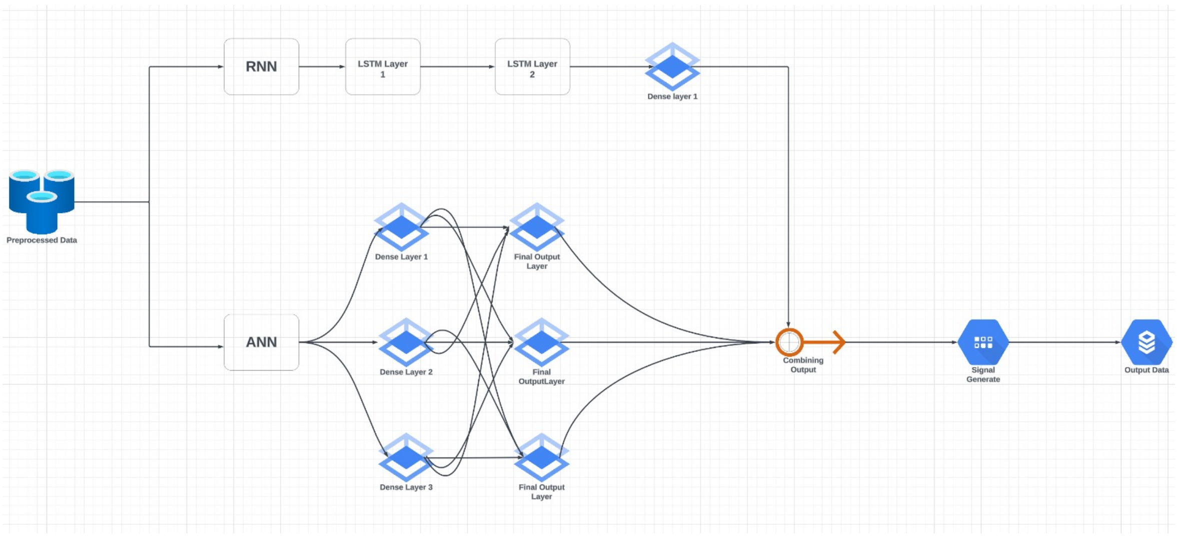


Figure 6.1: ANN+RNN Architecture

1. Data Collection: Historical data is obtained through the YFinance. At the same time, option data is imported, including options tickers like ImpliedVolatility, calls and puts for the selected expiration date. Finally, the collected data are processed to correct missing values using methods such as imputation or interpolation, thus ensuring the integrity of the data for subsequent analysis in the context of financial markets.
2. Data Preprocessing and plotting: The collected data is processed to remove potential inconsistencies and efficiently handle null values, ensuring data integrity. After pre-processing, the data is plotted to visually check for outliers or noise to help refine if necessary. In addition, both short and long moving averages are calculated to identify and analyze trends in a data set, providing valuable insight into the patterns behind stock data.

3. Model Training: Data is divided into training (X) and testing (Y) sets to evaluate model performance. StandardScaler is used to standardize the data, which improves model consistency. Both Artificial Neural Network (ANN) and LSTM (Recurrent Neural Network) architectures have been implemented. The models are assembled with the Adam optimizer, ReLU activation function and L1-L2 regularizers to avoid overfitting and reduce generalisation errors. Callbacks such as early stop and ReduceLROnPlateau are used to avoid overfitting. The learning rate is set to 0.0001 for optimal training stability.
4. Model Testing: Before training the ANN and RNN models, the data is reshaped to match the expected input format. In RNN models, data is divided into sequences to accommodate the sequential nature of RNNs. In addition, NaN values are handled to avoid interference during training, ensuring smooth convergence of models and performance evaluation.
5. Model Validation: In the ANN model, the epoch cycle is set to 600 to allow enough iterations for convergence due to the longer convergence time. On the other hand, the epoch cycle of the RNN model is set to 200 based on its faster convergence speed, although adjustments can be made based on the complexity of the data. The mean absolute error (MAE) and root mean square error (RMSE) values are calculated to evaluate the predictive accuracy of the model by comparing the actual and predicted values. Smaller values of MAE and RMSE indicate better forecasting performance of the models.
6. Plotting Learning Curve: A learning curve is plotted to estimate the bias and variance of the models. This helps visualize the model's performance on training and validation data from different epochs or iterations. Histograms of actual and predicted values are plotted to compare the models predictions on unseen data. This graphical comparison gives an idea of how well the model predicts the values in the entire data set.
7. Generating Trading Signals: Trading signals are generated using predictions from both ANN and RNN models. Based on these signals, a defined options trading strategy is implemented, which involves buying and selling call or put options.
8. Plotting Predicted Entry and Exit Points: Predicted entry and exit points generated by trading signals are plotted on the price -diagram of the stock.This visualization allows you to clearly present anticipated trading opportunities against the underlying stock price movement.

9. Backtesting and Optimization: Backtest results are thoroughly analyzed to identify potential weaknesses and areas for improvement in trading strategies. This analysis includes evaluating the performance of strategies under different market conditions and scenarios simulated using historical data. Historical data is used to simulate real-world scenarios, enabling a comprehensive assessment of the sustainability and adaptability of trading strategies to different market conditions.
10. Real Time Trading: A real-time trading function is implemented that uses the generated signals to make real-time trading decisions. This feature seamlessly integrates with brokers or trading platform APIs for instant order execution, ensuring timely entry and exit of positions based on trading signals generated by the model. It enables automated trading functions that respond quickly to market movements, facilitating the effective execution of trading strategies in real time.

Chapter 7

Testing

Sr No	Action	Input	Expected Output	Actual Output	Status
1	Preprocessing Functions - Data Filling, Volatility Calculation, Implied Volatility Handling and Moving Averages	Raw Stock and Options Data	Processed options data with required columns	Processed options data successfully	Passed
2	Function to plot and check outliers in data	Data with possible outliers	Plot graphs with no outliers	Plotted graphs with no outliers	Passed
3	Preprocessing, Splitting Data, Scaling Data	Data: DataFrame with features, Target, Test Size, Random State, Training, Validation, and Test datasets	Cleaned and organized data, Training and test datasets, Standardized feature values	Cleaned and organized data, Training and test datasets, Standardized feature values	Passed
4	ANN Model, RNN Model, Model Training, Model Evaluation, Model Summary	Input shape of features, Training datasets, Callbacks, Test datasets	Compiled, Trained ANN and RNN models, Summary of ANN and RNN models	Compiled, Trained ANN and RNN models, Summary of ANN and RNN models	Passed

Sr No	**Action**	**Input**	**Expected Output**	**Actual Output**	**Status**
5	Train ANN and RNN models using training data	Training Data, ANN and RNN Architecture	Trained ANN and RNN models	Trained ANN and RNN models	Passed
6	Test trained models using validation data	Validation Data	Model performance metrics on validation data like MAE and RMSE Values	Low MAE and RMSE values - 0.27 and 0.38 respectively	Passed
7	Evaluate ANN and RNN architecture performance	ANN and RNN architectures, training data	Training and validation loss curves	Smooth Training and validation loss curves depicting good generalization of model and no overfitting/underfitting	Passed
8	Plot histogram of actual and predicted values	Actual and predicted values	Histogram showing distribution of actual and predicted values	Very close and accurate comparison of actual and predicted values	Passed
9	Generate buy/sell signals based on model predictions	Trained models, features data	DataFrame with buy/sell signals based on ANN, RNN, MACD, and Bollinger Bands signals	Generated Buy/Sell Call and Put Signals	Passed
10	Plot signals on stock chart	Generated Signals, Selected Stock graph	Plotted Signals on stock data	Plotted Buy/Sell signals based on model predictions	Passed

Table 7.1: AI-DOT Test Case

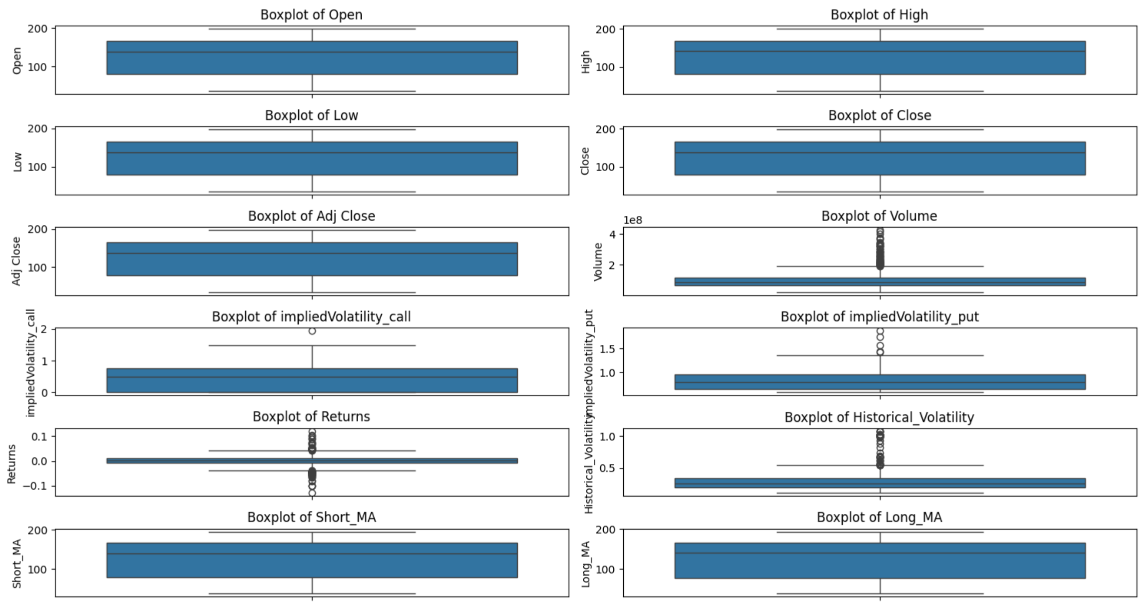


Figure 7.1: Box Plot to check for Noise/Outliers

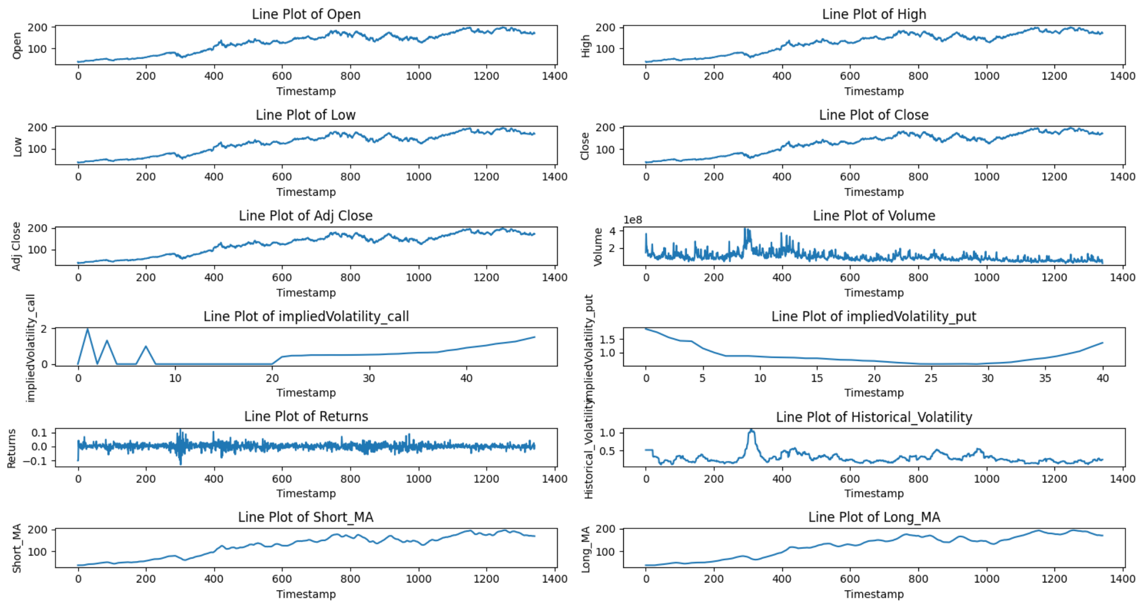
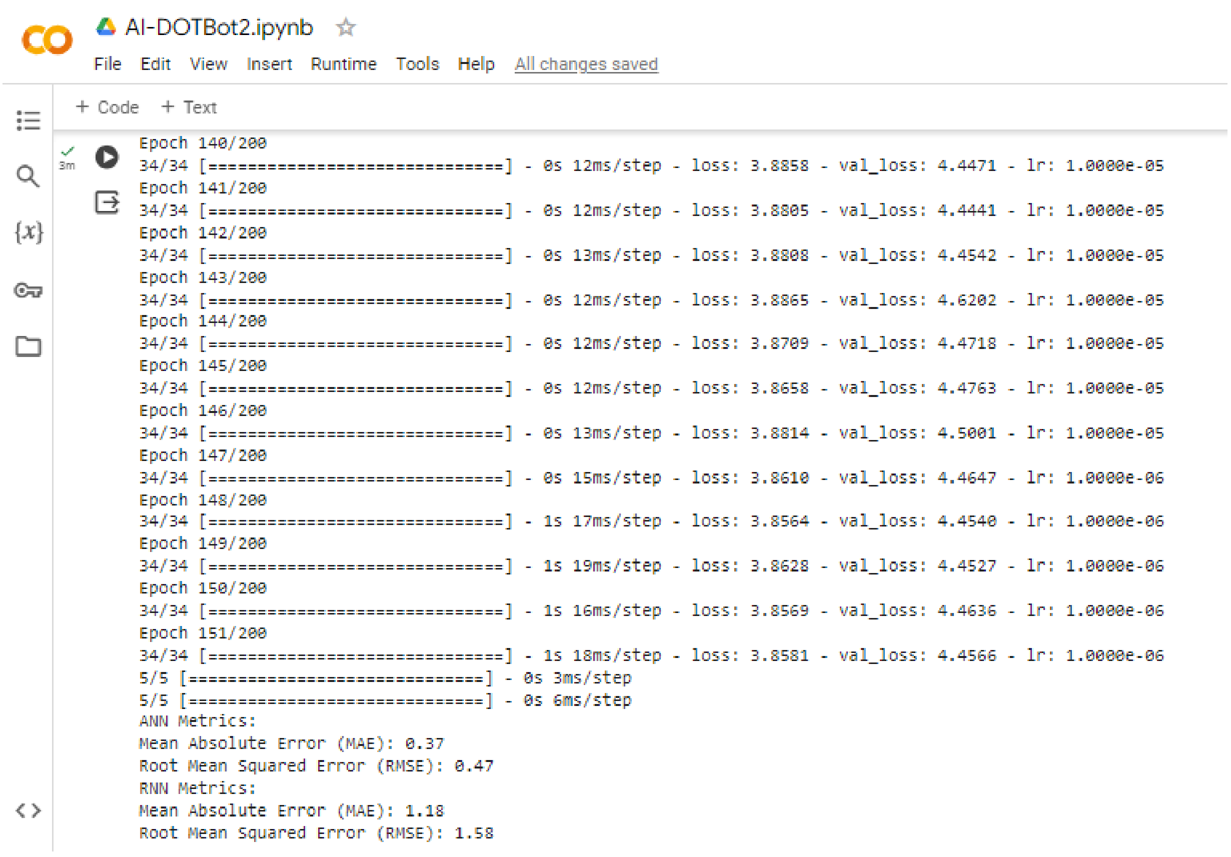


Figure 7.2: Line Plot to check for Noise/Outliers

Box plots are utilized to assess the spread and identify outliers within a dataset by visually displaying the distribution's quartiles, median, and any potential extreme values. On the other hand, line plots are employed to detect noise or irregularities in data patterns over time or ordered sequences, aiding in the identification of trends and fluctuations. Together, these plots offer complementary insights into the data, with box plots highlighting variability and extreme values, while line plots reveal temporal patterns and deviations, enabling a comprehensive assessment of data quality and integrity.



```
Epoch 140/200
34/34 [==============================] - 0s 12ms/step - loss: 3.8858 - val_loss: 4.4471 - lr: 1.0000e-05
Epoch 141/200
34/34 [==============================] - 0s 12ms/step - loss: 3.8805 - val_loss: 4.4441 - lr: 1.0000e-05
Epoch 142/200
34/34 [==============================] - 0s 13ms/step - loss: 3.8808 - val_loss: 4.4542 - lr: 1.0000e-05
Epoch 143/200
34/34 [==============================] - 0s 12ms/step - loss: 3.8865 - val_loss: 4.6202 - lr: 1.0000e-05
Epoch 144/200
34/34 [==============================] - 0s 12ms/step - loss: 3.8709 - val_loss: 4.4718 - lr: 1.0000e-05
Epoch 145/200
34/34 [==============================] - 0s 12ms/step - loss: 3.8658 - val_loss: 4.4763 - lr: 1.0000e-05
Epoch 146/200
34/34 [==============================] - 0s 13ms/step - loss: 3.8814 - val_loss: 4.5001 - lr: 1.0000e-05
Epoch 147/200
34/34 [==============================] - 0s 15ms/step - loss: 3.8610 - val_loss: 4.4647 - lr: 1.0000e-06
Epoch 148/200
34/34 [==============================] - 1s 17ms/step - loss: 3.8564 - val_loss: 4.4540 - lr: 1.0000e-06
Epoch 149/200
34/34 [==============================] - 1s 19ms/step - loss: 3.8628 - val_loss: 4.4527 - lr: 1.0000e-06
Epoch 150/200
34/34 [==============================] - 1s 16ms/step - loss: 3.8569 - val_loss: 4.4636 - lr: 1.0000e-06
Epoch 151/200
34/34 [==============================] - 1s 18ms/step - loss: 3.8581 - val_loss: 4.4566 - lr: 1.0000e-06
5/5 [==============================] - 0s 3ms/step
5/5 [==============================] - 0s 6ms/step
ANN Metrics:
Mean Absolute Error (MAE): 0.37
Root Mean Squared Error (RMSE): 0.47
RNN Metrics:
Mean Absolute Error (MAE): 1.18
Root Mean Squared Error (RMSE): 1.58
```

Figure 7.3: ANN and RNN Model's MAE and RMSE Values

The figure above demonstrates the training and validation performance of both the ANN and RNN models. Over 700 epochs, the ANN model achieves a decreasing loss trend, starting from 18146.8594 and converging to 0.8291, while the RNN model's loss decreases from 18159.8281 to 3.8581 over 200 epochs. Lower validation loss values for both models (18961.0996 for ANN and 18983.9434 for RNN) indicate effective generalization to unseen data. In terms of evaluation metrics, the ANN model yields an MAE of 0.37 and RMSE of 0.47, while the RNN model has an MAE of 1.18 and RMSE of 1.58. These values signify that the ANN model exhibits superior predictive accuracy and precision compared to the RNN model. Overall, the decreasing loss trends and relatively low validation loss values underscore the models' effective learning and generalization capabilities, with the ANN model outperforming the RNN model in predictive accuracy.

7.1 Images References

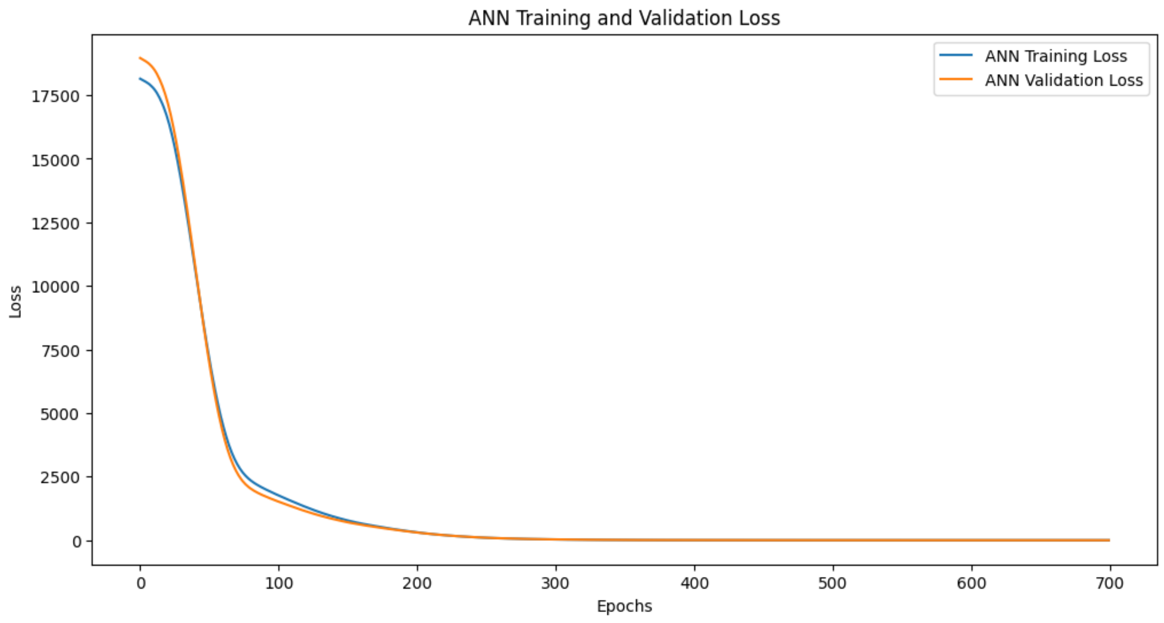


Figure 7.4: ANN Model Training and Validation Loss

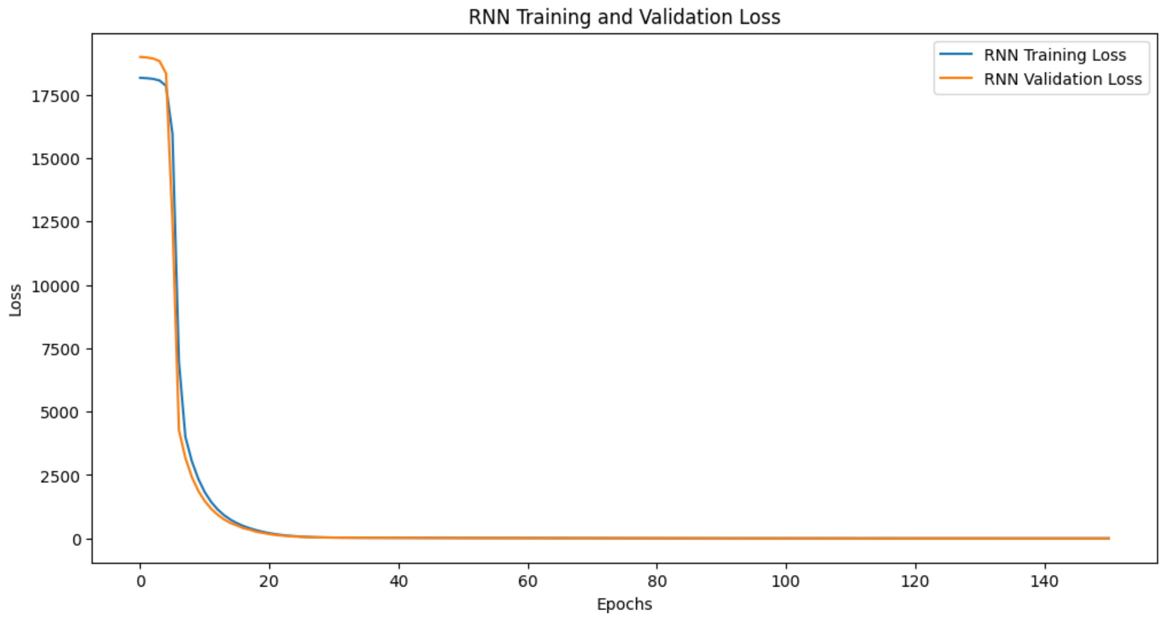


Figure 7.5: RNN Model Training and Validation Loss

This provides insights into how well the model is learning from the training data and how well it generalizes to unseen data. A decreasing training loss indicates that the model is improving its fit to the training data over epochs, while the validation loss gives an indication of how well the model is performing on data it has not seen before. Ideally, both training and validation losses should decrease over time, indicating that the model is learning effectively without over-fitting to the training data.

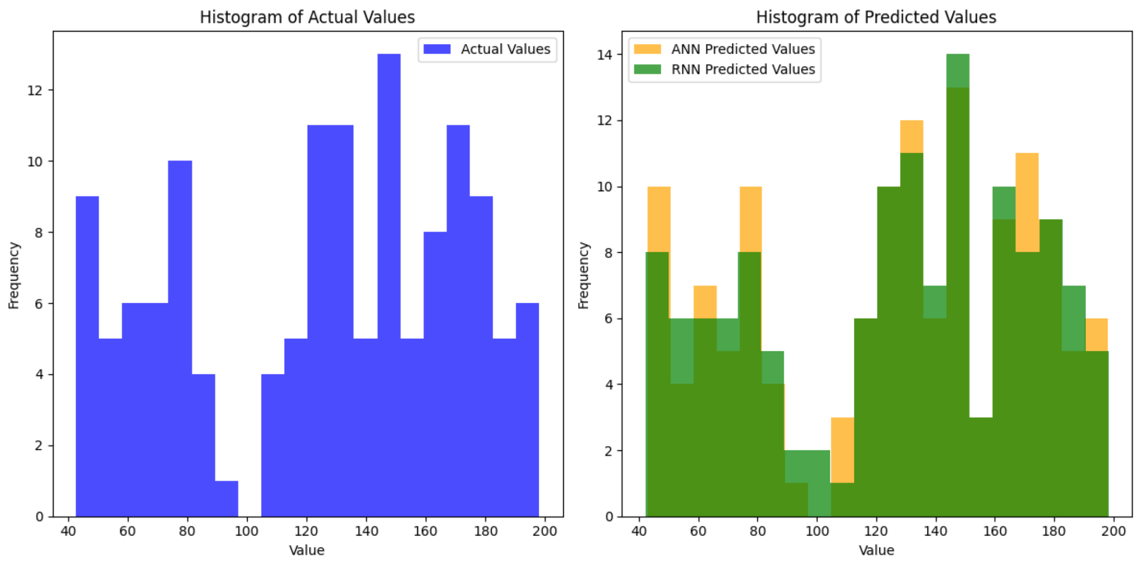


Figure 7.6: Histogram of Actual and Predicted Values

The histogram plotting of actual and predicted values provides a visual comparison between the distribution of the actual target values and the predicted values generated by the model. It helps to assess the accuracy and effectiveness of the model's predictions across the entire range of possible values. Ideally, the histograms of actual and predicted values should overlap closely, indicating that the model's predictions align well with the actual outcomes. Differences between the histograms may highlight areas where the model struggles to accurately predict certain values.

Chapter 8

Results and Analysis

8.1 Images References

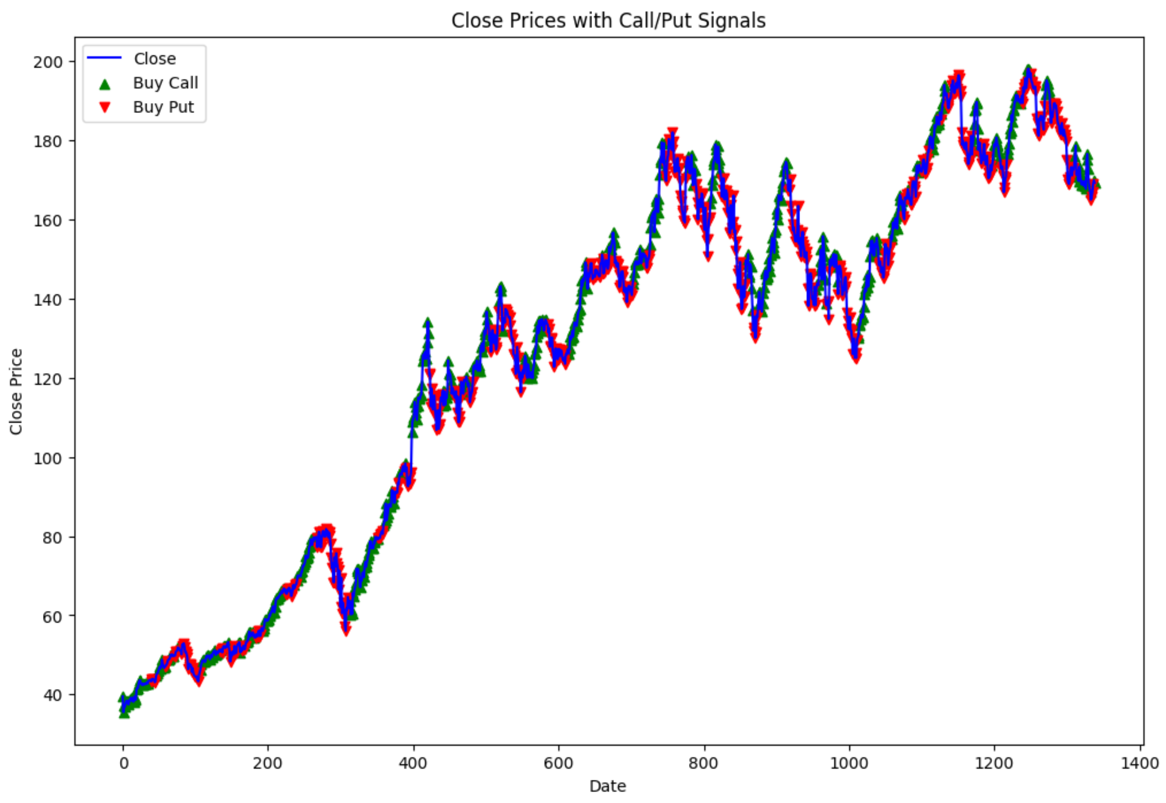


Figure 8.1: Call/Put Buy Signals

The graph displays closing prices of a financial instrument along with buy call (green) and buy put (red) signals based on the model's good generalization and predictions. Green triangles indicate suggested call option purchases, while red triangles represent opportunities to buy put options. These signals are derived from analysis involving technical indicators, market trends, and machine learning. Their alignment with price movements suggests the strategy captures market dynamics.

8.2 Images References

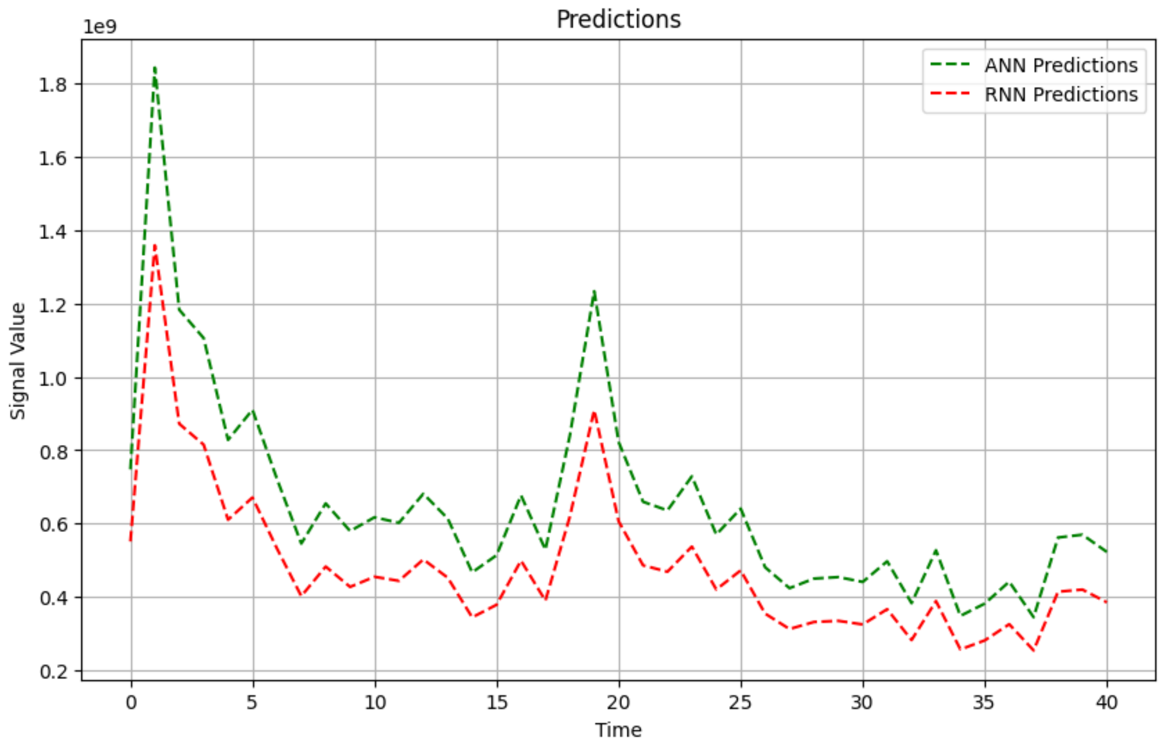


Figure 8.2: Predictions

The consistent alignment between the predictions of the Artificial Neural Network (ANN) and the Recurrent Neural Network (RNN) indicates a good performance across both models. Such resemblance in forecasts implies a good level of confidence in the models' ability to capture and interpret patterns within the data. Additionally, the fact that the ANN consistently yields higher predictions than the RNN on the y-axis signifies an optimistic stance regarding the anticipated outcomes. The positivity noted could stem from various factors, including the effectiveness of the chosen features, the generalization of the model's architecture, and the quality of the training data.

Chapter 9

Conclusion and Future Scope

This project focused on developing and evaluating trading signals using artificial neural networks (ANN) and recurrent neural networks (RNN) based on live and historical stock data and options data. By preprocessing the data to handle missing values and calculating additional features like historical volatility and implied volatility, we trained ANN and RNN models to predict future stock prices and generate trading signals. Through model evaluation and comparison, we assessed the performance of the trained models using metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). Leveraging the predictions from these models, we generated buy/sell signals for trading options, providing insights into the potential use of machine learning in enhancing trading strategies and decision-making processes in financial markets.

In future iterations, this project could incorporate live news sentiment analysis to enhance trading predictions. By integrating sentiment analysis of real-time news data, the system could dynamically adjust trading signals based on market sentiment, news events, and other external factors. This integration would enable the model to react to changing market conditions more effectively and potentially improve the accuracy of trading predictions.

Bibliography

[1] S. Pramod, B and S. Mallikarjuna, “Stock price prediction using lstm.,” *Applied sciences*, 2020.

[2] S. Ghosh, S. Kumar, A. Deshmukh, A. Kurve, and W. Dr.Rashmi, “Options trading using artificial neural network and algorithmic trading,” *International Journal of Next-Generation Computing Special issue, Vol. 13, No. 5. Research Gate*, 2022.

[3] D. Liu, Y. Liang, L. Zhang, P. Lung, and U. Rizwan, “Implied volatility forecast and option trading strategy.,” *International Review of Eco- nomics and Finance.*, 2020.

[4] J. Kavinnilaa, E. Hemalatha, M. S. Jacob, and R. Dhanalakshmi, “Stock price prediction based on lstm deep learning model.,” *IEEE*, 2021.

[5] A. Chatterjee, H. Bhowmick, and S. Jaydip, “Stock price prediction using time series, econometric, machine learning, and deep learning models.,” *IEEE*, 2021.

[6] H. N. Bhandari, B. Rimal, N. R. Pokhrel, R. Rimal, K. R. Dahal, and K. Rajendra, “Predicting stock market index using lstm.,” *IEEE*, 2022.

[7] Mr.P.Ramu, M. Priya, M. Sree, and Ms.S.Sankeerthana., “Stock price prediction using time series.,” *IRJET*, 2021.

[8] X. WENG, X. LIN, and S. ZHAO., “Stock price prediction based on lstm and bert.,” *IEEE*, 2022.

[9] M. Ge, S. Zhou, S. Luo, and B. Tian., “3d tensor-based deep learning models for predicting option price.,” *IEEE*, 2021.

[10] A. Lawi, H. Mesra, and S. Amir., “Implementation of long short-term memory and gated recurrent units on grouped time-series data to predict stock prices accurately.,” *IEEE*, 2021.

Appendix A

Appendices

A.1 Plagiarism Report