The University of Hong Kong
COMP4801 Final Year Project

# Empowering Financial Decisions with AI-Driven Forecasting and Personalized Investing
(FYP24089)

Final Report

**Group Members (UID):**
Lam Ngok Fung Herman (3035928328)
Wang Yu Jing (3035835595)
Sun Hao Yu (3035856044)
Muhammad Ghassan Jawwad (3035834802)

*Supervised by Dr. Difan Zou*

21th April, 2025

# Abstract

In today's complex stock market, novice investors often face challenges in navigating financial platforms and extracting actionable insights due to information overload and high learning barriers. This project aims to develop an AI-driven financial management platform to empower retail investors through enhanced stock price forecasting and personalized investing. The platform integrates three predictive models: an LSTM-based model for binary stock price movement prediction (UP/DOWN), an ARIMA model as a baseline for numerical price forecasting, and a suite of deep learning models, with LSTM-BERT selected for final numerical predictions due to its superior performance ($R^2$ 0.9378, RMSE 4.0892 for 7-day forecasts). These models leverage historical stock data and sentiment analysis from financial news, achieving an average LSTM accuracy of 60% for directional prediction and an ARIMA MAPE of 1.53% for numerical forecasts. A Django-based website consolidates these models, offering real-time stock updates, news, personalized recommendations, and a paper trading system, enabling users to practice trading without financial risk. Results demonstrate the platform's effectiveness in delivering intuitive insights, though limitations such as sentiment data gaps and model biases highlight areas for improvement. This project bridges predictive analytics with investor decision-making, providing a scalable tool for novice investors to engage confidently with the stock market.

# Acknowledgement

I wish to extend my heartfelt thanks to my project supervisor, Dr. Difan Zou, for his invaluable guidance and encouragement throughout the journey of this final year project. His insightful feedback and unwavering support were instrumental in navigating the various obstacles I encountered during the project's development, significantly contributing to its successful completion.

# Table of Contents

Abstract....................i
Acknowledgement....................ii
List of Figures....................vi
List of Tables....................vii
List of Equations....................viii
Abbreviations....................ix

**1. Project Background....................1**
1.1. Background....................1
1.2. Project Objectives....................2
1.3. Scope....................2
1.4. Project Milestones and Status....................3
1.5. Project Contribution....................4
1.6. Report Organization....................5
**2. Project Background and Literature Review....................7**
2.1. Project Background....................7
2.1.1. Information Overload....................7
2.1.2. Complexity of Financial Disclosures....................7
2.2. Limitations of Existing Investment Platforms....................8
2.2.1. Scattered Information Sources....................8
2.2.2. Lack of Intelligent Guidance....................8
2.2.3. Cost Barriers....................8
2.2.4. High Learning Barrier....................8
2.3. Literature Review....................9
2.3.1. Advances in AI and NLP for Finance....................9
2.3.2. Shortcomings of Existing Academic Models....................10
2.4. Project's Strengths and Improvements....................11
2.4.1. Lightweight Model Design....................11
2.4.2. Human-Centered Output Format....................11
2.4.3. Personalized Recommendations....................12
2.4.4. All-in-One Integrated Platform....................12
**3. Methodology....................13**
3.1. Overall Modelling Process....................13
3.1.1. Predicting Stock Price Direction Using LSTM Model....................13
3.1.2. Forecasting Stock Prices in Numerical Values Using a Statistical Model....................14

3.1.3. Forecasting Stock Prices in Numerical Values Using a Deep Learning Model....15
3.1.4. Overall Different Modelling Purposes....15
3.2. Machine Learning Models....8
3.2.1. Stock Price Direction Forecasting Model....16
3.2.1.1. Data Collection and Train-Validation-Test Splitting....18
3.2.1.2. Data Preprocessing: Heikin-Ashi Candlestick Transformation...18
3.2.1.3. Data Preprocessing: Feature Engineering....19
3.2.1.4. Data Preprocessing: Feature Standardization....21
3.2.1.5. Data Labelling....21
3.2.1.6. Model Architecture....22
3.2.2. Baseline Numerical Stock Price Prediction Model Using ARIMA....24
3.2.2.1. Overview of the ARIMA Approach....24
3.2.2.2. Data Collection and Train-Test Split....25
3.2.2.3. Feature Engineering....25
3.2.2.4. Stationarity Test....26
3.2.2.5. Model Training and Parameter Setting....26
3.2.2.6. Prediction and Evaluation Metrics....27
3.2.2.7. Result Visualization and Exports....27
3.2.3. Numerical Stock Price Prediction Using Deep Learning Models....29
3.2.3.1. Overview of the Approach....29
3.2.3.2. Model Selection....29
3.2.3.3. Data Reading and Preparation....32
3.2.3.4. Feature Engineering and Normalization....32
3.2.3.5. Model Architecture....33
3.2.3.6. Training Process and Parameter Settings....33
3.2.3.7. Prediction and Evaluation Metrics....34
3.2.3.8. Result Visualization and Export....34
3.3. Personalized Investment Recommendation System....35
3.4. Paper Trading System....37
3.5. Front-end and Back-end Construction....38
**4.** Results and Discussion....39
4.1. Stock Price Movement Prediction Model Using LSTM....39
4.1.1. Model Performance....39
4.1.2. Performance Analysis....42
4.1.2.1. Overall Performance....42
4.1.2.2. Class Imbalance and Prediction Bias....43
4.1.2.3. Stock Specific Performance....44
4.1.2.4. Impact of Features and Preprocessing....44
4.1.2.5. Model Architecture and Training....45
4.1.2.6. Limitations and Future Improvements....45
4.1.2.7. Practical Implications....46
4.2. Baseline Numerical Stock Price Prediction Model Using ARIMA....47

4.2.1. Overview of ARIMA Model Results........................................47
4.2.2. Model Performance................................................47
4.2.3. Performance Analysis.............................................49
4.2.3.1. Stationarity and Preprocessing.......................................49
4.2.3.2. Overall Performance Comparison..................................50
4.2.3.3. Model Limitation..............................................51
4.2.3.4. Practical Implications and Comparison..........................51
4.2.3.5. Future Improvements...............................................52
4.3. Numerical Stock Price Prediction Using Deep Learning Models............53
4.3.1. Model Performance................................................53
4.3.2. Performance Analysis and Comparison with ARIMA.....................54
4.3.2.1. Deep Learning Models Performance..............................54
4.3.2.2. Comparison with ARIMA Baseline................................55
4.3.2.3. Model Selection Rationale.........................................56
4.3.2.4. Practical Implications.........................................56
4.4. Web Development.............................................................57
**5.** Future Work.....................................................................64
**6.** Conclusion.......................................................................66

References...............................................................................68

# List of Figures

Figure 1. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for AAPL……………………………………………………………………40
Figure 2. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for MSFT……………………………………………………………………40
Figure 3. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for NVDA……………………………………………………………………40
Figure 4. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for AMZN……………………………………………………………………40
Figure 5. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for AVGO……………………………………………………………………41
Figure 6. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for META……………………………………………………………………41
Figure 7. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for COST……………………………………………………………………41
Figure 8. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for NFLX……………………………………………………………………41
Figure 9. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for GOOGL……………………………………………………………………41
Figure 10. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for TSLA……………………………………………………………………41
Figure 11. The ARIMA(7, 1, 0) Model Results with RMSE, MAPE, and MAE……………………………………………………………………48
Figure 12. The ARIMA(15, 1, 0) Model Results with RMSE, MAPE, and MAE……………………………………………………………………49
Figure 13. The p-value of the Training Time Series Before and After First-Order Differencing Under the ADF Test……………………...………………………50

# List of Tables

Table 1. Project Schedule with Descriptions and Status……………………………….3
Table 2. Summarization Table of All Performance Metrics for Each Stock……………………..………………………...……………………………42
Table 3. Summarization Table of the Results of the Six Deep Learning Over the 7-Day Forecast Horizon……………………………………………………………..53
Table 4. Summarization Table of the Results of the Six Deep Learning Over the 15-Day Forecast Horizon……………………………………………………………..53

# List of Equations

Equation 1. Equation to calculates stock's beta……………………………………20
Equation 2. Autoregressive Component of the ARIMA Model……………...……24
Equation 3. Moving Average Component of the ARIMA Mode……………………24
Equation 4. The First Order Differencing Formula for ARIMA Model……………..25
Equation 5. Formula for Calculating the Accuracy Metric…………………………..39
Equation 6. Formula for Calculating the Precision Metric…………………………40
Equation 7. Formula for Calculating the Recall Metric……………………………40
Equation 8. Formula for Calculating the F1-Score Metric…………………………40
Equation 9. Formula for Calculating the RMSE Metric……………………………47
Equation 10. Formula for Calculating the MAE Metric……………………………48
Equation 11. Formula for Calculating the MAPE Metric…………………………48

# Abbreviations

| Abbreviation | Definition |
|---|---|
| ADABOOST | Adaptive Boosting |
| ADF | Augmented Dickey-Fuller |
| ADX | Average Directional Movement Index |
| AI | Artificial Intelligence |
| API | Application Programming Interface |
| ARIMA | AutoRegressive Integrated Moving Average |
| ATR | Average True Range |
| BERT | Bidirectional Encoder Representations from Transformers |
| CCI | Commodity Channel Index |
| CNN | Convolutional Neural Network |
| DOM | Document Object Model |
| DRF | Django Rest Framework |
| EMA | Exponential Moving Average |
| EMH | Efficient Market Hypothesis |
| GRU | Gated Recurrent Unit |
| LSTM | Long Short-Term Memory |
| MACD | Moving Average Convergence Divergence |
| MAE | Mean Absolute Error |
| MAPE | Mean Absolute Percentage Error |
| MSE | Mean Squared Error |
| NLP | Natural Language Processing |
| REST | Representational State Transfer |
| RMSE | Root Mean Squared Error |
| RNN | Recurrent Neural Network |
| RSI | Relative Strength Index |

# 1 Project Background

This chapter offers a detailed overview of the project, starting with its background and motivation, followed by a discussion of its objectives and requirements.

## 1.1 Background

The stock market serves as an essential tool for distributing capital throughout the economy, allowing businesses to secure funding while stock prices mirror collective investor opinions about a company's present and future prospects. The task of forecasting stock prices has sparked significant discussion since the 20th century. According to the Efficient Market Hypothesis [1], outperforming the market is unfeasible in an efficient system where all market information is already embedded in stock prices. This idea aligns with the Random Walk Theory [2], which argues that the market's efficiency renders short-term price predictions impractical. On the other hand, the Dow Theory [3] contends that technical analysis of historical trends and patterns can enable short-term forecasts. These theoretical perspectives have driven various attempts to predict stock market movements for better investment outcomes.

The emergence of machine learning has introduced a range of predictive models leveraging diverse algorithms, with their effectiveness explored later in **Section 2.3**. Timely access to relevant data is increasingly critical for investors, as highlighted by both the Random Walk and Dow Theories. Platforms like Bloomberg [4], HKEX [5], and Yahoo! Finance [6] deliver real-time market updates and in-depth financial news to support informed investment choices. Likewise, trading platforms popular in Hong Kong, such as WeBull [7], Futu [8], and Longbridge [9], provide extensive features like market statistics, trading insights, and discussion forums. However, these tools often overwhelm beginners with their complexity, a problem exacerbated by steep costs—Bloomberg Terminal, for instance, charges around $27,660 annually [10].

To tackle these issues, this project focuses on building an accessible platform that aids novice investors in understanding market trends and enhances usability. It employs a machine learning model to consolidate data into a clear assessment of stock performance, using natural language processing (NLP) for sentiment analysis. Drawing on insights from existing models (**Section 2.3**), this project compares approaches like LSTM networks and Transformer architectures to create a reliable predictive tool. The platform will also offer personalized investment suggestions and integrate a paper trading system, enabling beginners to practice trading while capturing user behaviour to tailor recommendations to individual risk preferences, which are inserted as individual user profiles by users.

## 1.2 Project Objectives

This section defines the aims and purpose of the AI-driven financial platform, designed specifically to support investors with less investing experience. The project's goals include collecting and preparing data for machine learning model training, creating an intuitive user interface as a webpage, and developing a paper trading system with personalized recommendation features. The overarching aim is to launch a fully operational online platform that combines sentiment analysis, real-time market data display and prediction, and customized investment guidance to every individual investors.

The initiative centres on building an AI-enhanced financial tool that merges a paper trading environment with two machine learning models for stock price and movement forecasting. Key steps involve gathering and processing data to ensure model accuracy, optimizing the model through extensive testing, and designing an interface prioritizing ease of use. The final platform will deliver a functional website that analyses market sentiment through analysing the collected financial news and stock trends, providing timely alerts on investment prospects. By leveraging various preference filters, the system will generate personalized strategies aligned with each user's risk tolerance and preferences.

## 1.3 Scope

As the final year project's scope is to develop a fully functional website integrated with the model within an eight-month timeframe, this interim report specifically covers the progress made during the first semester, with a primary focus on design and development of the NLP models, comparison between models, and the website prototype development.

## 1.4 Project Milestones and Status

The project is currently on schedule and has completed all proposed tasks and phases. **Stage 1: Project Setup and Literature Review**, **Stage 2: Model Preparation and Training**, and **Stage 3: Model Enhancement and Application Development** have all been finished. Details of the project schedule and its status are provided in **Table 1** (Project Schedule).

*Table 1. Project Schedule with Descriptions and Status*

| **Schedule** | | **Milestones**<br>(number of learning hours) | | **Status** |
|---|---|---|---|---|
| **Phase 1: Project Inception** | | | | |
| 2024 | Sep | • Preparation of Project Plan (10)<br>• Literature Review (10)<br>• Setup of Project Website (10) | **Deliverables:**<br>1. Detailed Project Plan<br>2. Project Website Setup | **DONE** |
| **Phase 2: Project Elaboration** | | | | |
| 2024 | Oct | • Model Selection and Determination (10)<br>• Model Data Collection including numerical and textual data (10)<br>• Data Pre-processing and Cleaning (25) | | **DONE** |
| | Nov | • Feature Extraction and Determination (15)<br>• Model Training (20) | | **DONE** |
| | Dec | • LSTM-BERT Model Integration (10)<br>• Model Performance Evaluation and Reporting (10)<br>• Front-end Basic Webpage Design and Style Selection (10) | | **DONE** |
| 2025 | Jan | • Interim Report Drafting (20)<br>• Model Performance Evaluation (10)<br>• Front-end Basic Webpage Design and Style Selection (10) | **Deliverables:**<br>1. Preliminary Trained Model Prototype<br>2. Interim Report<br>3. First Presentation | **DONE** |
| **Phase 3: Project Construction** | | | | |
| 2025 | Feb | • Prediction Model Enhancement and Modification (20)<br>• Baseline Model Building and Evaluation (20)<br>• Application Frontend and Backend Construction (20) | | **DONE** |

| | | | | |
|---|---|---|---|---|
| | | • Construction and Determination of the User-definable Trading Preference Rules (10) | | |
| | Mar | • Baseline Model Building and Evaluation (10)<br>• Personalized Recommendation System (10)<br>1. Recommendations generation<br>2. Optimization and Enhancement<br>• Paper Trading system performance evaluation and optimization (10)<br>• Final Report Drafting (10)<br>• Application Frontend and Backend Construction (15)<br>• Software Review and Testing (15) | | **<u>DONE</u>** |
| | Apr | • Final Software Debugging and Testing (10)<br>• Final Report Drafting (30)<br>• Project Video Construction (10)<br>• Preparation of Project Poster (10) | **<u>Deliverables:</u>**<br>1. Project Exhibition – 3-min Video<br>2. Project Exhibition – Poster Preparation<br>3. Final Report<br>4. Final Product<br>5. Final Presentation | **<u>PARTLY DONE</u>** |

## 1.5 Project Contribution

| *Group Member* | *Contribution to the Project* |
|---|---|
| ***LAM NGOK FUNG HERMAN*** | • Project Webpage Construction<br>• Stock Price Movement Prediction Model – Binary Classification using LSTM<br>o Data Collection and Pre-processing<br>o Feature Selection and Processing<br>o Model Construction and Training<br>o Model Evaluation<br>• Stock Price Prediction Baseline Model – ARIMA<br>o Data Collection and Pre-processing<br>o Model Construction and Fitting<br>o Model Evaluation<br>• Personalized Recommendation System Design |
| *WANG YU JING* | • Stock Website Construction<br>o Frontend<br>▪ UI Design & Implementation<br>o Backend |

| | |
|---|---|
| | ▪ Database Design & Construction<br>▪ Integration of Trained Models & Data Input & Website<br>▪ Stock Data & News Collection & Periodically Updates<br>▪ Login/Signup System<br>• LSTM-BERT Model Training<br>• Testing & Bug Fixing<br>• Paper Trading System Design & Implementation<br>• Personalized Recommendation System Design & Implementation |
| *SUN HAO YU* | • Model Construction and Training of Stock Price Prediction<br>• Report Writing and Early organizations of team work<br>• Code Testing and Bug fixing<br>• Marginal Contribution in Data Collection Design |
| *MUHAMMAD GHASSAN JAWWAD* | Designed and implemented a dynamic stock data pipeline that collects, cleans, and structures real-time financial news and historical stock prices from multiple APIs (Finnhub, and NewsAPI).<br><br>Developed a modular news aggregation system combining:<br>• Finnhub for historical news (up to 1 year)<br>• NewsAPI for high-resolution daily headlines (last 30 days)<br><br>Wrote scripts to embed the entire pipeline into the Django backend via custom management commands: update_news for daily news updates.<br><br>Engineered data deduplication, formatting, and CSV exports, generating clean datasets for: NLP processing (headline + summary)<br><br>Built scalable architecture supporting multiple stocks, easy expansion to new tickers, and reusable code modules for future forecasting or trading tools.<br><br>Enabled model-ready datasets stored in:<br>• news_data/{SYMBOL}_news_from_2025.csv<br>• price_data/{SYMBOL}_price_history.csv<br><br>Updated the FYP webpage online and completion of final presentation. |

## 1.6 Report Organization

This report is structured as follows: Chapter 2 provides a detailed overview of the project background, literature reviews, and the project's strengths and improvement. Chapter 3 outlines the methodology for data collection, model development, and the construction of the overall AI-embedded web application. Chapter 4 discusses the results obtained, difficulties, and current limitations, including those related to model construction and website development. Additionally, Chapter 5 presents the project status and outlines future work. Finally, Chapter 6 concludes the report.

# 2 Project Background and Literature Review

This chapter provides an in-depth exploration of the project's context, identifying challenges faced by retail investors, limitations of current investment platforms, and literature review, followed by a discussion of how this project addresses the issues and its strengths.

## 2.1 Project Background

This section outlines the key difficulties retail investors encounter in navigating the stock market, focusing on issues of information overload, complex financial disclosures, and poor performance trends in specific markets.

### 2.1.1 Information Overload

Retail investors frequently face an overwhelming amount of financial data from online sources, which complicates decision-making. Research by Barber and Odean indicates that an abundance of information can overwhelm investors, leading to reduced trading activity [11]. This deluge of data, ranging from stock prices to news updates, often leaves novice investors unable to distill actionable insights, resulting in hesitation or suboptimal choices. The sheer volume of available information, while beneficial in theory, becomes a practical barrier for those lacking the expertise to filter it effectively.

### 2.1.2 Complexity of Financial Disclosures

Financial disclosures, such as annual reports and earnings statements, are typically lengthy and laden with technical jargon, posing a significant challenge for retail investors. This complexity creates an entry barrier, discouraging participation in stock markets and widening the gap between professional and retail investors. Simplifying access to and interpretation of such disclosures is thus critical to empowering a broader investor base.

## 2.2 Limitations of Existing Investment Platforms

This section evaluates the shortcomings of current investment platforms, emphasizing their impact on novice investors and the need for more accessible tools.

### 2.2.1 Scattered Information Sources

Current investment platforms often require users to consult multiple sources for comprehensive data, such as WeBull for trading and Futu for market insights. This lack of a unified interface fragments the user experience, making it cumbersome for novices to gather and analyse information efficiently. The absence of a centralized system increases the time and effort required to make informed decisions, particularly for those unfamiliar with navigating disparate tools.

### 2.2.2 Lack of Intelligent Guidance

Most platforms fail to leverage user behaviour or preferences to offer personalized investment advice. Without intelligent guidance, retail investors must independently interpret complex datasets, increasing the risk of poor choices. This gap in adaptive support limits the platforms' utility for beginners, who need tailored recommendations to navigate the financial landscape effectively.

### 2.2.3 Cost Barriers

Access to premium financial tools and real-time data often comes with significant costs, such as the $27,660 annual subscription for a Bloomberg Terminal [10]. These expenses exclude budget-conscious retail investors from leveraging high-quality resources. Affordable alternatives are essential to build access to sophisticated financial insights.

### 2.2.4 High Learning Barrier

The prevalence of sophisticated financial terminology and advanced analytical tools on existing platforms creates a steep learning curve. Non-professional investors often find these features inaccessible, reducing their confidence and engagement. Simplifying these interfaces and providing intuitive explanations could bridge this gap, making investment more approachable for a wider audience.

## 2.3 Literature Review

This section reviews recent advancements in AI and NLP for financial forecasting, identifies shortcomings in existing academic models.

### 2.3.1 Advances in AI and NLP for Finance

The rapid progress in ML and NLP technologies has profoundly reshaped approaches to stock market forecasting, paving the way for more accurate predictions. Numerous studies have investigated a variety of architectures and methods, each adding valuable contributions to the dynamic field of finance. Gu et al. (2024) proposed a FinBERT-LSTM model that integrates sentiment analysis of financial news with stock prediction, surpassing traditional methods in accuracy [12]. Furthermore, Zong and Zhou developed the Multimodal Stable Fusion with Gated Cross-Attention (MSGCA) model, which combines financial indicators, dynamic texts, and graph relationships, achieving an accuracy improvement of 8.1–31.6% across various datasets [13]. These advancements highlight AI's potential to enhance predictive capabilities in finance.

Apart the multi-modal approach, a notable study by Selvin et al. explored the performance of deep learning frameworks, such as CNNs employing a sliding window technique, RNNs, and LSTM networks, in forecasting stock prices using real-time data. Their analysis revealed that the CNN model consistently outperformed its counterparts, a result attributed to its superior capability in detecting sudden market volatility, thus demonstrating its resilience in unstable market environments [14].

Expanding on the role of sentiment in financial forecasting, Mohan et al. enhanced the precision of sentiment analysis within deep learning systems by assembling a comprehensive dataset covering over five years, encompassing more than 265,000 financial news articles. Their work highlighted the importance of high-quality, extensive datasets in improving the effectiveness of sentiment-driven models [15]. This approach illustrates the growing need for large-scale data aggregation to capture market sentiments effectively.

Additionally, Qing et al. investigated the use of Transformer models for stock price prediction, introducing modifications to the traditional Transformer architecture by reducing redundant heads in the transformer's self-attention mechanism. These adjustments enabled their model to better capture long-term dependencies in financial time series data, outperforming LSTM networks in some particular scenarios requiring temporal analysis [16]. This advancement points to the potential of Transformer-based models to address the limitations of earlier architectures in handling prolonged market trends.

Collectively, these research efforts demonstrate the remarkable progress in applying cutting-edge ML and NLP techniques to stock market prediction. The evolving approaches not only improve forecasting accuracy but also reveal the critical connection between sentiment analysis and financial prediction, laying a solid foundation for developing more advanced and integrated forecasting systems in the financial domain.

### 2.3.2 Shortcomings of Existing Academic Models

Despite the impressive progress, current models face different significant limitations. High computational demands, as seen in MSGCA, require multi-modal data fusion and GPU-intensive training, restricting its scalability [13]. Additionally, outputs are often presented as raw probabilities or classifications, lacking context or user-friendly formats for retail investors. Many models also overlook individual investor behaviour, which is critical for personalization, and remain standalone prototypes without integration into practical trading systems. These gaps limit their real-world applicability.

## 2.4 Project’s Strengths and Improvements

This subsection outlines the key strengths of the proposed AI-driven financial management platform, highlighting how it addresses the limitations of existing models through a lightweight design, user-friendly outputs, personalized recommendations, and an integrated system.

### 2.4.1 Lightweight Model Design

The project prioritizes a lightweight design by leveraging BERT and LSTM models for sentiment analysis and stock price forecasting. Our approach focuses on text-based NLP and traditional machine learning techniques. BERT, a pre-trained model which extracts sentiment from news articles and social media efficiently, while LSTM captures temporal dependencies in stock price data with small computational overhead. This design choice significantly reduces training complexity and deployment costs, making the system accessible for retail investors without requiring high-end hardware. By avoiding the need for extensive multi-modal integration, the platform ensures scalability, allowing it to handle predictions for multiple NASDAQ stocks without excessive resource demands. This efficiency aligns with the project’s goal of providing an affordable solution for novice investors, who often lack access to premium tools due to cost barriers.

### 2.4.2 Human-Centered Output Format

To enhance usability, the project emphasizes human-centered outputs tailored for retail investors. The platform provides intuitive visualizations such as sentiment trend lines and predicted stock movement direction levels. Sentiment trend lines, derived from BERT’s analysis of financial news, illustrate the emotion over time, helping users understand market sentiment at a glance. Furthermore, for the stock price movement prediction labels, it constantly tracks the past history of stock prices and various financial indicators to provide a simple classification label for users. These features address the high learning barrier identified in existing platforms. By presenting data in a visually digestible manner, the platform empowers users to make decisions without requiring deep financial expertise.

### 2.4.3 Personalized Recommendations

The platform incorporates personalized recommendations by learning from user preferences and behaviour, addressing the lack of intelligent guidance in existing systems. Users are required input their risk aversion levels, expected returns, and sector preferences through a dedicated profile page in the webpage. The system then filters stock predictions, which are generated by the LSTM model for directional forecasting and the LSTM-BERT model for numerical price predictions, based on the inserted criteria. The platform outputs actionable advice (BUY, SELL, HOLD), ensuring recommendations align with individual risk profiles. This personalization enhances decision-making for novices.

### 2.4.4 All-in-One Integrated Platform

The project delivers an all-in-one integrated platform, combining news analysis, sentiment modelling, paper trading, stock price forecasts, and personalized feedback within a single website. The news sentiment scores and stock price movement predictions from the trained models are displayed alongside interactive charts. Personalized feedback, derived from user interactions and the collected user preferences, further refines recommendations, fostering a continuous learning environment. This holistic approach addresses the scattered information sources of various existing platforms, offering retail investors a comprehensive tool for learning, analysis, and decision-making.

# 3 Methodology

In this chapter, Section 3.1 outlines the various models planned for development within the project's modelling framework. Section 3.2 provides a detailed discussion of the machine learning model architectures employed. Section 3.3 describes the structure and functionality of the personalized investment recommendation system. Section 3.4 details the implementation process for the paper trading system. Finally, Section 3.5 explains the approaches used for building the website's frontend and backend components.

## 3.1 Overall Modelling Process

To meet the varied requirements of users, multiple predictive models have been created and deployed, each customized to align with distinct investment goals. The modeling framework is organized into three primary components:

### 3.1.1. Predicting Stock Price Direction Using LSTM Model

This model tackles the binary classification challenge of determining whether a stock's price will increase or decrease on a daily basis, simplifying decision-making for certain investor groups by providing straightforward outcomes.

The first component involves developing a LSTM-based model to predict the directional movement of stock prices, categorizing the outcome as either an increase or decrease for a daily forecast horizon. This binary classification approach is particularly tailored for cautious investors, such as retirees or risk-averse individuals, who prefer clear and simple indicators before engaging in more complex numerical forecasts. By focusing on a binary outcome—rise or fall—the model reduces the cognitive burden on these investors, enabling them to make decisions with greater confidence. The rationale for prioritizing this basic classification step stems from the need to establish a foundational understanding of market trends before delving into intricate numerical predictions. Such an approach ensures that investors, especially novices, can grasp fundamental market movements without being overwhelmed by detailed quantitative data. The use of LSTM in this context leverages its ability to capture temporal dependencies in sequential data, making it well-suited for identifying short-term patterns in stock price movements.

### 3.1.2. Forecasting Stock Prices in Numerical Values Using a Statistical Model

This component focuses on employing a statistical model, specifically Autoregressive Integrated Moving Average (ARIMA), to predict stock prices numerically, highlighting its role as a baseline for comparison with the latter deep learning models.

The second component employs a statistical model, specifically the ARIMA model, to forecast stock prices numerically. The inclusion of a statistical baseline is crucial for establishing a benchmark against which more complex deep learning models can be evaluated. ARIMA was chosen for its simplicity, interpretability, and established effectiveness in time series forecasting, particularly for financial data with linear trends. It models the relationship between a stock's past prices and its future values using autoregressive and moving average components, while the integration step (differencing) ensures stationarity, which is a key requirement and assumption for reliable predictions.

Using ARIMA as a baseline is essential because it provides a reference point to quantify the improvements offered by deep learning models, which often require more computational resources and data. Furthermore, ARIMA's ability to handle short-term forecasts makes it a suitable starting point for understanding stock price dynamics before introducing models that incorporate external factors like market sentiment. This step ensures a rigorous comparison, allowing us to assess whether the added complexity of deep learning models yields significant performance gains over traditional statistical methods.

### 3.1.3. Forecasting Stock Prices in Numerical Values Using a Deep Learning Model

This component utilizes deep learning to predict numerical stock prices, integrating external factors such as market news, and compares multiple models to select the most effective one for forecasting.

The third component involves the application of deep learning models to predict stock prices numerically, incorporating external factors like market news and sentiment to improve accuracy and better reflect the intricacies of real-world financial systems. To ensure optimal performance, a total of six distinct deep learning models will be developed and evaluated, including variations of LSTM, GRU, CNN, and Transformer architectures. These models will be trained and tested, and their performance will be compared using metrics such as RMSE, MAE, and MAPE. The model demonstrating the highest predictive accuracy will be selected for the final forecasting task. This comparative approach ensures that the chosen model not only leverages market sentiment and other external data but also provides the most reliable predictions, enhancing the platform's ability to deliver actionable insights for investors.

### 3.1.4. Overall Different Modelling Purposes

Each of these strategies has been carefully crafted to meet varying user needs, ensuring a comprehensive approach to stock market forecasting. For instance, the binary classification model delivers straightforward directional insights, ideal for quick decision-making, while the deep learning-based forecasting model, enriched with sentiment data, offers a more comprehensive perspective by integrating diverse information sources, such as market news and social media trends. By employing multiple models and conducting thorough performance evaluations on a consistent dataset, this project enables a systematic comparison of forecasting techniques. This iterative process facilitates the identification of the most effective model, allowing for continuous refinement and optimization to better serve investors across different experience levels and investment goals.

## 3.2 Machine Learning Models

This section outlines the methodologies applied in developing three fundamental machine learning models central to the project's objectives. The first model focuses on directional forecasting using LSTM networks to provide investors with essential insights into stock price movements. The second model delivers numerical stock price predictions via a statistical approach, establishing a baseline for comparison. The third model enhances numerical forecasting by integrating sentiment analysis from financial news, enriching the input data for deep learning frameworks. Each model's development process is detailed in the subsections below, highlighting the techniques used to achieve reliable predictions. The implementation and initial findings for directional and numerical price forecasting are discussed in Section 4.

### 3.2.1 Stock Price Direction Forecasting Model

This subsection provides a comprehensive overview of the dataset, preprocessing methods, and labeling strategy for the stock price direction forecasting model. The model tackles the binary classification task of predicting daily stock price movements using LSTM, laying a foundational step for subsequent models. It is particularly designed for novice investors who benefit from understanding price direction before engaging with complex numerical forecasts, thus enabling strategic decision-making in a volatile market environment.

The initial model in this project employs Long Short-Term Memory (LSTM) networks to address the binary classification challenge of predicting whether the daily stock price of various securities will rise or fall. This directional forecasting model serves as a critical starting point for the project's predictive framework, offering an entry into stock market analysis. Specifically, it caters to novice investors, such as those new to trading or with limited risk tolerance, by providing a clear and binary outcome—either an upward or downward movement. This simplicity is crucial as it allows beginners to grasp essential market trends without being overwhelmed by intricate numerical predictions, which are introduced in later models. The rationale for prioritizing this binary classification step lies in its role as a foundational tool. For instance, knowing whether a stock is likely to increase in value can inform an investor’s decision to explore further details, such as the expected price range. By leveraging LSTM’s capability to model temporal dependencies in sequential data, this model captures short-term patterns in stock price movements, enabling investors to make decisions in a dynamic and often unpredictable market.

#### 3.2.1.1 Data Collection and Train-Validation-Test Splitting

The stock price direction forecasting model is designed to predict future price movements of stocks listed in the Nasdaq stock market index, focusing exclusively on the 10 top companies with the highest Nasdaq portfolio share to ensure relevance and data availability. The dataset is sourced through the Yahoo! Finance API in Python, where each data point corresponds to a single trading day. These data points include daily metrics such as opening, closing, high, and low prices, as well as trading volume. To maintain accuracy, the closing prices are adjusted to account for corporate actions like stock splits and dividend payments, ensuring the dataset reflects true market performance over time.

The dataset covers a period from January 1, 2010, to December 31, 2024, spanning 3,773 trading days. To optimize model training and performance, the dataset is divided into three subsets: 70% for training (approximately 2,641 days), 15% for validation (approximately 566 days), and 15% for testing (approximately 566 days). This split introduces a validation set to facilitate early stopping during model training. Early stopping monitors the model's performance on the validation set, halting training when the validation loss ceases to improve, thus reducing the effect of overfitting and ensuring the model generalizes well to unseen data. This approach is particularly beneficial for LSTM models, which are prone to overfitting due to their complexity, and it enhances the robustness of the model by balancing training efficiency with predictive accuracy. The training set is used to fit the model, the validation set to tune hyperparameters and implement early stopping, and the test set to evaluate the model's final performance, ensuring a comprehensive assessment of its predictive capabilities.

#### 3.2.1.2 Data Preprocessing: Heikin-Ashi Candlestick Transformation

After data collection, a pivotal preprocessing step involves converting the raw stock price data into Heikin-Ashi candlesticks, a technique aimed at enhancing the quality of the input data for forecasting stock price movements. Heikin-Ashi candlesticks are specifically engineered to smooth out price volatility, thereby improving the detection of underlying trends by mitigating market noise and providing a clearer picture of price direction. This method is particularly advantageous for directional forecasting, as it reduces the likelihood of misleading signals often encountered with traditional candlestick patterns, which can be distorted by short-term price fluctuations.

The transformation process entails specific calculations for each trading day. The Heikin-Ashi closing price is determined by averaging the day's open, close, high, and low prices, creating a balanced representation of price behavior. The opening price for a Heikin-Ashi candlestick is computed as the average of the previous day's Heikin-Ashi open and close values. The high and low prices are calculated by taking the maximum and minimum values respectively, among the current day's high, low, Heikin-Ashi open, and Heikin-Ashi close, capturing the range of price movement while maintaining the smoothing effect. Through these computations, the dataset is converted into a Heikin-Ashi format, which offers a more stable and trend-focused foundation for subsequent analysis. This preprocessing step not only enhances the model's ability to identify persistent market trends but also improves the reliability of directional predictions, making it an essential component of the forecasting pipeline for novice investors seeking clear and actionable insights.

#### 3.2.1.3 Data Preprocessing: Feature Engineering

To enhance the predictive accuracy of the stock price direction forecasting model, a carefully selected set of technical indicators and market-related features has been derived from the Heikin-Ashi transformed data and integrated as input features. These features provide significant advantages by simplifying the noisy information inherent in stock price data, such as price trends, momentum, and market relationships, into interpretable metrics. By encapsulating essential elements of market behavior, including trends, momentum, volatility, and broader market dynamics, these features enable the model to detect meaningful patterns and improve the reliability of predictions. The inclusion of both stock-specific and market-wide indicators ensures a more holistic understanding of the factors influencing stock price direction, thereby strengthening the model's forecasting framework.

A total of seven features have been selected, each chosen for its specific contribution to capturing market dynamics. Among the trend-based indicators, the **Exponential**

**Moving Average (EMA)** smooths price data to reveal the underlying trend direction, making it easier to identify whether a stock is in an upward or downward trajectory over the specified window. The **Average Directional Movement Index (ADMI)** quantifies the strength of the identified trend, providing insight into whether the trend is robust enough to influence price direction. For momentum indicators, the **Stochastic K%** and **Stochastic D%** are both employed to assess the speed and sustainability of price movements, helping to identify overbought or oversold conditions that may signal potential reversals in price direction. These indicators collectively offer a view of stock-specific trends and momentum, which are critical for predicting short-term directional movements.

To incorporate broader market context, the **Closing Market Index of Nasdaq** over the same window is included as a feature, reflecting the overall market environment in which the stock operates. Additionally, the **VIX Index** dataset for the same period is integrated to capture market volatility, often referred to as the "fear gauge", which influences investor sentiment and stock price movements. Furthermore, a key market-related feature is the **Rolling Stock Beta**, which measures the stock's sensitivity to market movements. Beta ($\beta$) is calculated using the following equation (**Equation 1**):

*Equation 1. Equation to calculate stock's beta*

$$\beta = \frac{Cov(R_{stock}, R_{market})}{Var(R_{market})}$$

where $R_{stock}$ and $R_{market}$ are the daily returns of the stock and the market (Nasdaq index), respectively, computed as percentage changes in the closing price and the Nasdaq closing index. The covariance and variance are calculated over a rolling window of 252 trading days, aligning with standard financial practice for annual beta estimation. Including beta allows the model to assess how closely a stock's price movements correlate with the broader market, providing critical insight into systemic risk exposure. The addition of market-related features like beta, the Nasdaq index, and the VIX ensures the model captures not only stock-specific dynamics but also the broader market environment, enhancing its ability to predict directional shifts in a more comprehensive manner.

By integrating these features, the model gains a multidimensional perspective on market behavior. The combination of stock-specific indicators and market-wide metrics creates a balanced framework that accounts for both internal stock dynamics and external market influences, thereby improving the overall robustness and accuracy of the directional forecasting model.

#### 3.2.1.4 Data Preprocessing: Feature Standardization

Feature standardization is applied to normalize all input features, ensuring they exhibit a mean of zero and a unit variance, which is a critical step for enhancing the predictive model's performance and stability. This normalization process mitigates biases that arise from varying scales among the input features. During model training, standardization facilitates faster convergence of gradient-based optimization algorithms by ensuring that features contribute proportionally to the learning process, preventing those with larger numerical ranges from disproportionately influencing the model compared to features with smaller ranges, thereby improving the model's ability to learn meaningful patterns in the data. This step is especially crucial for directional forecasting, as it allows the model to focus on the relative relationships between features rather than their absolute magnitudes, leading to more robust predictions of stock price movements.

#### 3.2.1.5 Data Labelling

The labeling of the target variable involves assigning a distinct label to each valid data point, corresponding to a single trading day, based on the movement of the adjusted closing price. A label of 'UP' is assigned to a data point if the closing price on that day is higher than the previous day's closing price, indicating a price increase, while a label of 'DOWN' is applied if the closing price decreases compared to the prior day. By focusing on price direction rather than magnitude, this method enables investors to make straightforward decisions without needing to interpret the relatively noisy numerical forecasts.

### 3.2.1.6 Model Architecture

The Long Short-Term Memory (LSTM) model is employed for predicting stock price direction due to its exceptional ability to capture long-term dependencies in sequential time series data, such as financial stock data. Unlike traditional Recurrent Neural Networks (RNNs), which often struggle with issues like vanishing gradients during backpropagation, LSTMs are engineered to retain information over extended time periods, making them ideal for time series analysis where past trends significantly influence future outcomes. This capability allows the LSTM to effectively identify patterns in stock prices, which are often driven by nonlinear dynamics and external factors such as market sentiment, macroeconomic events, and investors' behavior. The model's proficiency in handling these complexities ensures reliable forecasts in the volatile financial markets.

To improve the model's robustness and prevent overfitting, a set of dropout layers is incorporated into the architecture using the Keras library in Python. Dropout randomly deactivates a subset of neurons during training, forcing the model to learn more generalized features that are less dependent on specific inputs, thereby enhancing its performance on unseen data.

Additionally, the binary cross-entropy loss function is utilized to optimize the model during training. This loss function measures the difference between predicted probabilities and actual labels, enabling the model to refine its predictions, thus improving its precision in forecasting directional movements.

To further enhance model performance, an early stopping mechanism is implemented, monitoring the validation loss during training. Early stopping halts the training process when the validation loss ceases to decrease, reducing the chances of overfitting and ensuring the model generalizes well to new data. This technique leverages the validation set to assess the model's performance on unseen data, balancing training efficiency with predictive accuracy. Furthermore, a Grid Search approach is employed to optimize the hyperparameters of the LSTM and Dense layers, specifically the number of hidden units. Various configurations are systematically tested, and the model's performance is evaluated based on validation accuracy. Only the model yielding the highest validation accuracy is selected for the final prediction task. This optimization process, combined with early stopping, ensures that the model is both accurate and efficient, providing investors with reliable directional insights to support their own financial decision-making.

### 3.2.2 Baseline Numerical Stock Price Prediction Model Using ARIMA

This subsection outlines the development of the second component of the project's modeling framework, focusing on a statistical approach using the ARIMA model to predict numerical stock prices. It serves as a baseline for comparison with deep learning models, providing a simpler but effective method for forecasting stock prices.

#### 3.2.2.1 Overview of the ARIMA Approach

The Autoregressive Integrated Moving Average (ARIMA) model is a widely adopted statistical method for time series forecasting, particularly suited for financial data such as stock prices. ARIMA combines three components: Autoregression, Integration, and Moving Average, denoted as ARIMA($p$,$d$,$q$), where $p$ is the order of the autoregressive term, $d$ is the degree of differencing, and $q$ is the order of the moving average term.

The autoregressive component models the relationship between a time series and its lagged values, which can be represented as:

*Equation 2. Autoregressive Component of the ARIMA Model*

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t$$

where $y_t$ is the stock price at time $t$, $\phi$ are the autoregressive coefficients, and $\epsilon_t$ is the noise.

The moving average component models the relationship between the series and past forecast residuals:

*Equation 3. Moving Average Component of the ARIMA Model*

$$y_t = \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q}$$

where $\theta$ are the moving average coefficients.

The integration component ($d$) involves differencing the series to achieve stationarity, which is an assumption for ARIMA modeling. For a series $y_t$, first-order differencing ($d = 1$) is computed as:

*Equation 4. The First Order Differencing Formula for ARIMA Model*

$$y'_t = y_t - y_{t-1}$$

ARIMA's strengths lie in its simplicity and interpretability, making it an ideal baseline for financial forecasting. It effectively captures short-term linear trends in stock price data, requiring minimal computational resources compared to deep learning models. The model relies on historical data alone, without needing external features like market sentiment, which simplifies its implementation.

#### 3.2.2.2 Data Collection and Train-Test Split

The ARIMA model utilizes historical stock data from the Nasdaq, focusing on the top 10 stocks by the weight in Nasdaq's portfolio. The dataset is retrieved via the Yahoo! Finance API using Python's yfinance library, with each data point representing a single trading day. Missing values are removed to maintain data integrity, and the resulting dataset spans from January 1, 2020, to April 17, 2025, covering approximately 1,300 trading days.

The dataset is partitioned into training and testing sets to facilitate model development and evaluation. A split of 70% for training and 30% for testing is adopted based on the total dataset size. This split ensures that the training set captures a substantial portion of historical trends, allowing the ARIMA model to learn patterns effectively, while the testing set provides a sample for assessing predictive performance on unseen data. This division balances the need for sufficient training data with a meaningful evaluation phase, ensuring the model's reliability for numerical stock price predictions.

#### 3.2.2.3 Feature Engineering

Feature engineering for the ARIMA model is streamlined to focus on the closing price as the sole input feature. As ARIMA models are designed to capture patterns within a single time series, leveraging its historical closing values to predict future prices is crucial. By using only the closing price, the model avoids the complexity of incorporating external features. The simplicity of this approach ensures that the model remains computationally efficient and interpretable, making it a baseline for comparison with more complex deep learning models.

### 3.2.2.4 Stationarity Test

A crucial step in preparing the data for ARIMA modeling is ensuring stationarity, as the model assumes that the time series has a constant mean, variance, and autocorrelation structure over time. To verify this, the Augmented Dickey-Fuller (ADF) test is applied to the closing price time series before and after first-order differencing. Stationarity is essential as trends and seasonality may distort the model's ability to capture true patterns. Differencing is used to remove trends and stabilize the mean, making the series stationary. In ARIMA, the differencing parameter $(d)$ specifies the number of times this operation is applied.

The ADF test assesses the stationarity of the input series, where the null hypothesis $(H_0)$ posits that the series is non-stationary, and the alternative hypothesis $(H_1)$ suggests stationarity. The test is performed on the original closing price time series and the first-order differenced series. For each stock, the $p - value$ is computed: a p-value less than $0.05$ leads to rejecting the null hypothesis, while a $p - value$ greater than $0.05$ indicates non-stationarity. This test ensures that the ARIMA model's assumption of stationarity is met and validating the choice of $d$.

### 3.2.2.5 Model Training and Parameter Setting

The ARIMA model is trained using two configurations to capture different temporal dependencies in stock price data, which are the ARIMA(7,1,0) for the 7-day prediction and ARIMA(15,1,0) for the 15-day prediction. The parameter $p$ is set to 7 and 15 respectively to reflect the number of lagged observations considered in the model. These values are chosen to align with the look-back periods of 7 and 15 days, capturing short-term and medium-term price patterns. The differencing parameter $(d = 1)$ is selected based on the stationarity test results, where first-order differencing consistently achieves stationarity $(p - value < 0.05)$ for all stocks, as confirmed by the ADF test.

The moving average parameter ($q = 0$) is set to simplify the model, as stock price data often exhibits strong autoregressive behavior.

This configuration balances model complexity with predictive power, providing a reliable baseline for numerical stock price forecasting while ensuring computational efficiency during training.

#### 3.2.2.6 Prediction and Evaluation Metrics

The ARIMA model generates numerical stock price predictions using a rolling forecast approach, where the model is initially trained on the training set and then iteratively updated with each test observation to predict the next day's price. The model's performance is evaluated using three standard metrics: Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). RMSE measures the square root of the average squared differences between predicted and actual prices. MAE calculates the average absolute differences, providing a straightforward measure of prediction accuracy. MAPE expresses the average absolute error as a percentage of the actual values.

For each stock, these metrics (RMSE, MAE, MAPE) are computed and stored, enabling a comprehensive assessment of the model's predictive accuracy. The results are saved as bar charts, facilitating visual comparison across all 10 Nasdaq stocks. These metrics are crucial for the later comparative analysis with deep learning models, as they provide a benchmark to evaluate whether the added complexity of deep learning yields significant improvements over the statistical ARIMA baseline.

#### 3.2.2.7 Result Visualization and Exports

To aid users in understanding and interpreting the ARIMA model's prediction outcomes, several visualization techniques are implemented.

First, time-series forecast plots are generated for each stock, illustrating the trend comparison between actual and predicted prices over the test period. These plots display the actual closing prices alongside the forecasted values using a dashed line, allowing users to visually assess the model's ability to capture price trends and fluctuations over time.

Second, scatter plots are created to depict the correlation between predicted and actual prices, with ideal results aligning along a diagonal line. These plots highlight the model's predictive accuracy and any biases, providing a clear visual representation of forecasting performance.

Additionally, the bar charts for all model's metrics are produced to horizontally compare the model's performance across the three evaluation metrics for all 10 Nasdaq stocks. These bar charts facilitate a straightforward comparison of forecasting accuracy across the dataset.

### 3.2.3 Numerical Stock Price Prediction Using Deep Learning Models

This section outlines the development of deep learning models for numerical stock price prediction, detailing the approach, model selection, and implementation process across data preparation, training, evaluation, and visualization.

#### 3.2.3.1 Overview of the Approach

Numerical stock price forecasting using machine learning models, such as predicting a stock's price three days ahead, is framed as a regression task, unlike trend prediction which focuses on market directions. The model processes inputs like historical prices, technical indicators, financial news, and sentiment scores, outputting a precise numerical value for the future price. This method provides quantifiable outputs ideal for short-term trading strategies, offering clear price targets for decision-making. However, it is more susceptible to market noise and struggles with sudden events like black swan incidents, as these introduce volatility that is challenging to model accurately. Despite these hurdles, the approach's ability to deliver concrete predictions makes it valuable for investors seeking detailed insights into future stock prices.

#### 3.2.3.2 Model Selection

This study evaluates six deep learning models to assess their effectiveness in stock price forecasting, ranging from simple temporal models to complex hybrid architectures with sentiment integration. The models' theoretical foundations, structures, benefits, and drawbacks are analyzed below:

1. **<u>GRU (Gated Recurrent Unit)</u>**
    - **Background**: A simplified LSTM variant, GRU reduces complexity by omitting gates like the output gate, enhancing efficiency.
    - **Architecture**: Features update and reset gates to manage information flow, focusing on relevant sequence patterns.
    - **Advantages**: GRUs offer high computational efficiency due to fewer parameters, making them ideal for rapid iterations or resource-constrained environments.

- **Limitations**: GRUs are less expressive than LSTMs, struggling with very long-term dependencies due to their simplified structure. GRUs are also less effective at handling complex feature interactions, particularly when integrating multimodal data like sentiment scores, limiting their applicability in volatile markets.

2. **LSTM (Long Short-Term Memory Network)**
   - **Background**: An RNN architecture addressing vanishing gradients, designed for long-term sequential dependencies.
   - **Architecture**: LSTMs feature forget, input, and output gates. The forget gate discards irrelevant information, the input gate updates the cell state with new data, and the output gate generates predictions.
   - **Advantages**: LSTMs excel at capturing long-term dependencies in stock price data, such as seasonal trends, ensuring stable training for 15-day medium-term forecasts. LSTMs are robust for sequential data with strong temporal patterns.
   - **Limitations**: The high parameter count increases training time and computational demand. LSTMs are also sensitive to noise in financial data and struggle to incorporate multimodal features like textual sentiment without architectural extensions, limiting their adaptability in complex scenarios.

3. **CNN (Convolutional Neural Network)**
   - **Background**: Adapted from image processing, CNNs excel in 1-D time series for local pattern detection.
   - **Architecture**: Employs 1-D convolution kernels, followed by flattening and dense layers for prediction.
   - **Advantages**: CNNs are computationally efficient with fast training times due to the simple structure. CNNs effectively capture short-term price volatility, such as daily fluctuations, making them suitable for 7-day forecasts in less complex markets.
   - **Limitations**: CNNs lack mechanisms for long-term memory, restricting their ability to model extended trends or dependencies.

4. **LSTM-BERT (Sentiment-Augmented LSTM)**

- **Background**: This hybrid model, inspired by recent advancements in NLP, integrates BERT-derived sentiment scores with LSTM, aiming to capture market emotions alongside price data.
- **Architecture**: Sentiment scores are extracted using BERT from financial news and concatenated with price features as input.
- **Advantages**: The model enhances sensitivity to market mood swings, improving predictions in sentiment-driven scenarios, such as volatile tech stocks.
- **Limitations**: Sentiment scores may introduce noise, especially with sparse or inaccurate news data. BERT's computational intensity increases training time, and the model may overfit if sentiment features dominate without sufficient price data, reducing reliability in stable markets.

5. **CNN-LSTM (Hybrid Convolutional and Recurrent Network)**
   - **Background**: CNN-LSTM combines CNN's local pattern extraction with LSTM's sequential modeling.
   - **Architecture**: A CNN layer extracts short-term features, followed by a LSTM layer for temporal dependency modeling, and a dense layer for regression output.
   - **Advantages**: This hybrid approach leverages CNN's efficiency in detecting short-term trends and LSTM's strength in sequential modeling, performing well for stocks with periodic behaviors. It is more robust to noise than standalone models.
   - **Limitations**: The combined architecture increases complexity, leading to longer training times. It requires extensive hyperparameter tuning, and the performance can degrade if short-term patterns dominate over long-term trends.

6. **CNN-LSTM-BERT (Multichannel Sentiment-Enhanced Model)**
   - **Background**: A multimodal approach for time series modelling, CNN-LSTM-BERT integrates sentiment analysis with price data, building on CNN-LSTM by adding BERT sentiment scores for enhanced forecasting.
   - **Architecture**: CNN is used to capture local price patterns, LSTM models temporal sequences, and BERT-derived sentiment scores form an additional input channel, fused via concatenation before a dense output layer for regression.

- **Advantages**: It effectively models price dynamics and sentiment shifts. The multimodal input enhances predictive accuracy.
- **Limitations**: The model's complexity demands significant computational resources, considerably increasing training time and memory usage. The model also risks overfitting if the sentiment effect is low, and the potential redundant features from BERT may reduce efficiency, requiring careful feature selection and model parameters tuning.

The six models span a range from simple temporal architectures to advanced multimodal frameworks, enabling a thorough assessment of diverse modeling strategies. By systematically analyzing their real-world performance in stock price prediction, the study identifies the models that effectively balance accuracy, computational efficiency, and practical utility. These findings provide critical insights for refining future forecasting systems, guiding the selection of architectures that optimize predictive power while maintaining usability.

#### 3.2.3.3 Data Reading and Preparation

The implementation begins by loading financial news datasets for selected stocks across forecast windows (7-day, 15-day), followed by basic preprocessing to ensure data quality. News data, comprising titles and summaries, forms the raw text for sentiment analysis. To improve readability and analyze word frequencies, *jieba* performs Chinese word segmentation, while *WordCloud* generates visualizations. Stock price data, including daily open, close, high, low, and volume, is retrieved using *pandas_datareader* from the *Stooq* API. Timestamps from the news data are aligned with stock prices by date, ensuring consistency for multimodal models.

#### 3.2.3.4 Feature Engineering and Normalization

Feature engineering leverages daily stock prices to form the core inputs for prediction. For models incorporating sentiment, a *bert_sentiment* score is added, extracted via HuggingFace's *distilbert-base-uncased-finetuned-sst-2-english* model, fine-tuned on the Stanford Sentiment Treebank v2 (SST-2) dataset for English sentiment classification. Each news item yields a sentiment score (positive or negative), enhancing the model's ability to capture market emotions. Features and target values are normalized using MinMaxScaler to scale them within [0, 1], ensuring uniform contribution during training. A 14-day sliding window generates sequential training samples, where each input sequence comprises 14 days of features, and the target is the closing price on the 15th day.

#### 3.2.3.5 Model Architecture

The six deep learning models are constructed using TensorFlow's Sequential API. The LSTM model features a single LSTM layer and a dense layer to model long-term dependencies. The GRU model, with a similar setup, uses fewer parameters for faster computation. The CNN model applies convolutional filters to detect local price patterns, followed by a dense output layer. The CNN-LSTM hybrid uses CNN layers for short-term feature extraction, feeding into an LSTM for sequence modelling. The LSTM-BERT model enhances LSTM with a sentiment score input channel, while CNN-LSTM-BERT integrates all three models for comprehensive modelling. Each model incorporates a dropout layer to mitigate overfitting, concluding with a dense output layer for regression, ensuring accurate price predictions across diverse architectures.

#### 3.2.3.6 Training Process and Parameter Settings

The deep learning models are trained using Mean Squared Error (MSE) as the loss function, optimized with the Adam optimizer for efficient gradient descent. Training incorporates two callbacks:

- **EarlyStopping**: It halts training if validation loss (val_loss) does not improve after 30 epochs, preventing overfitting.
- **ReduceLROnPlateau**: It lowers the learning rate upon validation loss plateaus, enhancing convergence.

Models are trained for varying epochs, specifically 15 for simpler models like LSTM and up to 200 for complex models like CNN-LSTM-BERT, to balance performance and efficiency.

Data is split into training and testing sets with ratios such as 80:20 or 55:45, depending on the model's complexity and data needs. Random seeds (set_seed) are fixed to ensure reproducibility across experiments, allowing consistent evaluation of model performance.

### 3.2.3.7 Prediction and Evaluation Metrics

Predictions are generated by applying the trained deep learning models to the test set, producing numerical stock price forecasts for specified horizons (7-day, 15-day). Model performance is evaluated using four regression metrics:

- **$R^2$ (Coefficient of Determination)**: $R^2$ assesses variance explanation, with values closer to 1 indicating better fit.
- **RMSE (Root Mean Squared Error):** RMSE measures average prediction error magnitude, emphasizing larger errors
- **MAE (Mean Absolute Error)**: MAE provides a linear error measure for overall stability.
- **MAPE (Mean Absolute Percentage Error): MAPE** expresses errors as percentages, enabling cross-stock comparisons.

These metrics are computed for each model and stock, stored in a Pandas DataFrame, and exported to designated files for subsequent comparative analysis with statistical baselines.

### 3.2.3.8 Result Visualization and Export

The system generates visualizations to help users interpret the deep learning models' prediction outcomes. The generated visualizations are as follows:

- **Time-Series Forecast Plots**: They display actual versus predicted prices, illustrating trend alignment over the forecast period.
- **Scatter Plots** : They visualize the correlation between actual and predicted prices, with points near the diagonal indicating high accuracy.
- **Model Metric Bar Charts**: They compare performance across $R^2$, RMSE, MAE, and MAPE for all models, using Matplotlib and Seaborn to create horizontal bar charts with metric values labeled for clarity.

## 3.3 Personalized Investment Recommendation System

This section outlines the methodology for developing a personalized investment recommendation system, which leverages the predictive models to provide tailored trading advice to users based on their risk preferences and market sentiment.

### 3.3.1 Data Input and User Preference Collection

The recommendation system begins by collecting user-specific data to understand individual investment preferences. Upon registration, users are prompted to specify their risk tolerance level (low, moderate, high) through a dedicated profile setup page on the website.

### 3.3.2 Integration of the NLP Model

The system integrates the trained LSTM and deep learning models to generate predictions that inform the recommendation logic. The LSTM model provides daily binary classifications (UP/DOWN) for stock price movements, while the deep learning model outputs precise numerical price forecasts. A backend function processes news data daily, fetched via the yfinance API to compute sentiment scores that enhance the deep learning model's forecasting accuracy. This dual-model approach ensures that recommendations are informed by both directional trends and sentiment-driven price expectations.

### 3.3.3 Personalized Recommendation Algorithm

The recommendation algorithm combines model predictions with user risk profiles to generate tailored trading advice. For unregistered users, a general rule is applied: if the expected return exceeds 1.5%, the recommendation is "BUY"; if it falls below $-1.5\%$, it is "SELL"; otherwise, it is "HOLD". For registered users, the system adjusts these thresholds based on risk preference:

- **Risk-Averse Users**: Recommend "BUY" only if the expected return is greater than 5% or the LSTM-predicted UP direction probability exceeds 0.8.
- **Risk-Neutral Users**: Recommend "BUY" if the return exceeds 3% or the UP probability is above 0.6.
- **Risk-Seeking Users**: Recommend "BUY" if the return exceeds 1% or the UP probability is above 0.5.

### 3.3.4 Recommendation Result Display and User Interaction

The recommendations are presented through an intuitive interface on the stock detail page, where users can view the predicted price, direction, and the sentiment score. Users can interact with the system by accepting or rejecting recommendations, and their actions are logged in the table, allowing the system to refine future suggestions based on user feedback. This iterative process ensures that the recommendation system adapts to individual user needs.

## 3.4 Paper Trading System

We will create a paper trading system to improve user experience by incorporating a personalized investment recommendation feature. This will enable users to test investment advice and various trading strategies in a simulated environment. The system will provide virtual funds for executing simulated buy/sell transactions based on recommendations from our developed models and real-time stock market data. All transaction information will be stored in a SQLite database, ensuring that decisions are based on current market conditions. A trading engine will update users' positions and account balances according to their stock portfolios. Furthermore, users will be able to evaluate the effectiveness of recommendations through trade analysis and modify their actual trading strategies based on risk and return rates.

## 3.5 Front-end and Back-end Construction

This section discusses the development of the website's frontend and backend components, focusing on how they support the stock price prediction system and user interactions.

### 3.5.1 Frontend Development

The frontend is built using HTML, CSS, and JavaScript within the Django framework, ensuring a responsive and user-friendly interface. Key pages include the homepage, stock list, stock detail, news, model introduction, and paper trading dashboard. Django templates dynamically populate data, such as the NASDAQ-100 stock list on the stocks page, where users can sort by market capitalization or filter by price movement. User authentication pages (register, login, logout) are implemented to secure access to personalized features like paper trading.

### 3.5.2 Backend Development

The backend is developed using Django, managing data processing, API interactions, and model deployment. The SQLite database, structured with tables like NasdaqTicker, DailyQuote, and PredictionResult, stores stock data, news, and predictions. Django handles API requests to fetch real-time data via yfinance, scheduling updates with Django-crontab for news. The backend loads the trained machine learning models and scalers to generate daily predictions. It also processes user requests, such as paper trading transactions, updating the TradingAccount and Position tables accordingly.

# 4 Results and Discussion

This chapter presents the results and analysis of the financial management platform developed for stock price prediction and personalized investing. The project encompasses three predictive models, which are LSTM for stock price movement direction, ARIMA as a baseline for numerical price prediction, and deep learning models for the final numerical forecasting, along with a fully functional website integrating these models. The following sections evaluate the performance of each component, discuss their implications, and highlight the practical utility of the integrated platform for investors.

## 4.1 Stock Price Movement Prediction Model Using LSTM

This section evaluates the performance of the Long Short-Term Memory (LSTM) model developed for predicting stock price movements of the top 10 Nasdaq stocks by portfolio weight. The model, designed for binary classification (UP or DOWN movements), leverages historical stock data from January 1, 2010, to December 31, 2024, sourced via the Yahoo! Finance API. Features such as technical indicators (EMA, ADMI, Stochastic K%, D%), and market metrics (Nasdaq Index, VIX, Beta) are processed and standardized. The model's predictions on the test set (561 days) are analysed using confusion matrices, focusing on accuracy, precision, recall, and F1-score to assess its effectiveness.

### 4.1.1 Model Performance

The LSTM model was evaluated on the test set for each of the top 10 Nasdaq stocks. Performance metrics, including the accuracy, precision, recall, and F1-score, were calculated from the confusion matrices. The formulas used are as follows:

- **Accuracy**: The proportion of correct predictions over total predictions.

*Equation 5. Formula for Calculating the Accuracy Metric*

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision (UP class)**: The proportion of correct UP predictions out of all predicted UPs.

*Equation 6. Formula for Calculating the Precision Metric*

$$Precision_{UP} = \frac{TP}{TP + FP}$$

- **Recall (UP class)**: The proportion of actual UPs correctly predicted.

*Equation 7. Formula for Calculating the Recall Metric*

$$Recall_{UP} = \frac{TP}{TP + FN}$$

- **F1-Score (UP class)**: The harmonic mean of precision and recall for the UP class.

*Equation 8. Formula for Calculating the F1-Score Metric*

$$F1 - Score_{UP} = 2 \times \frac{Precision_{UP} \times Recall_{UP}}{Precision_{UP} + Recall_{UP}}$$

where $TP$ is the true positives, $TN$ is the true negatives, $FP$ is the false positives, $FN$ is the false negatives.

The confusion matrices for all ten stocks are presented as follows (**Figure 1-10**):

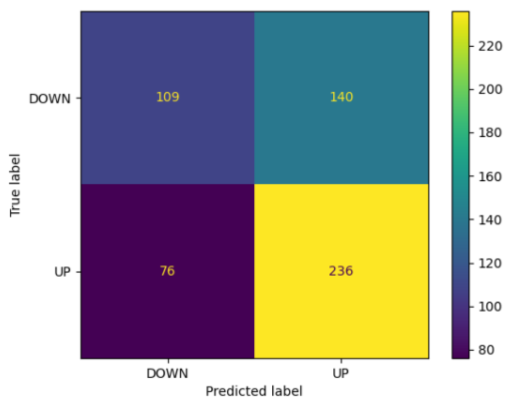


*Figure 1. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for AAPL*

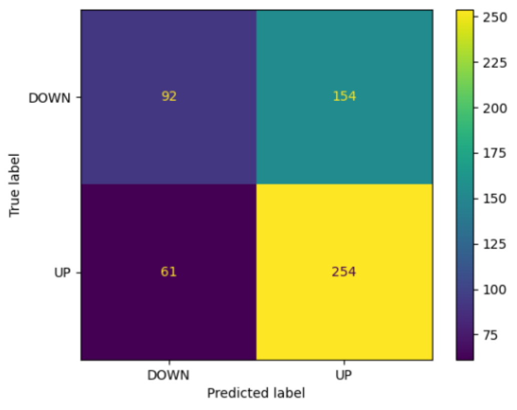
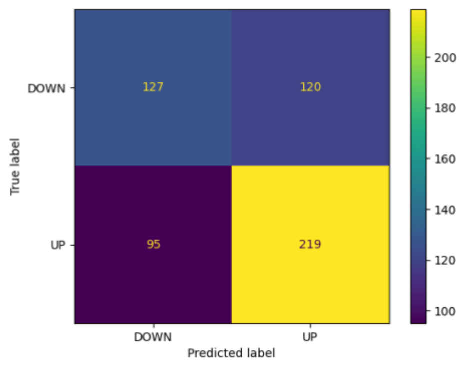


*Figure 2. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for MSFT*



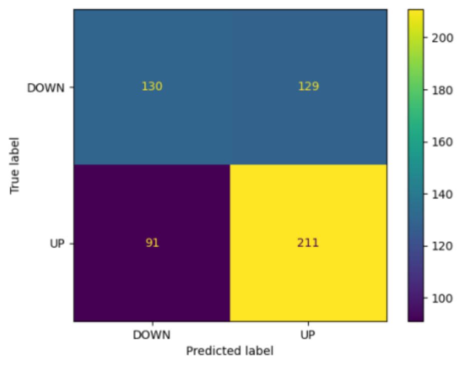
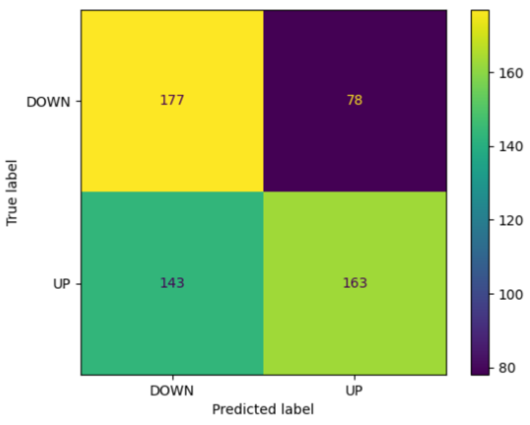


*Figure 3. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for NVDA*

*Figure 4. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for AMZN*



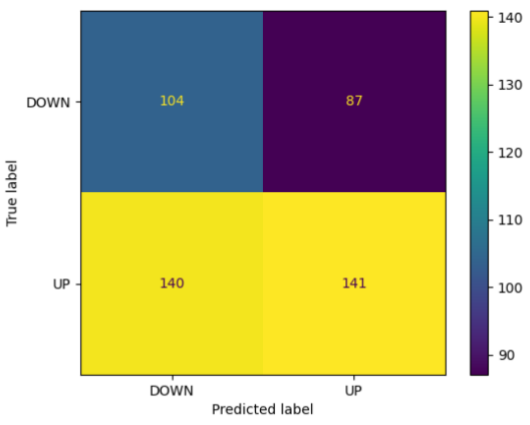
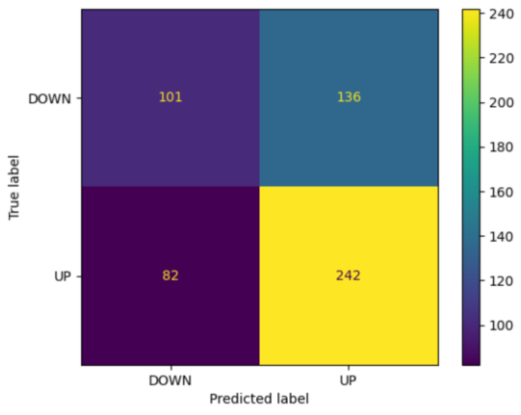


*Figure 5. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for AVGO*

*Figure 6. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for META*



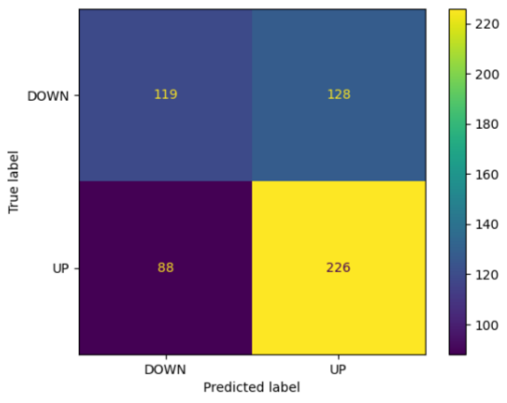
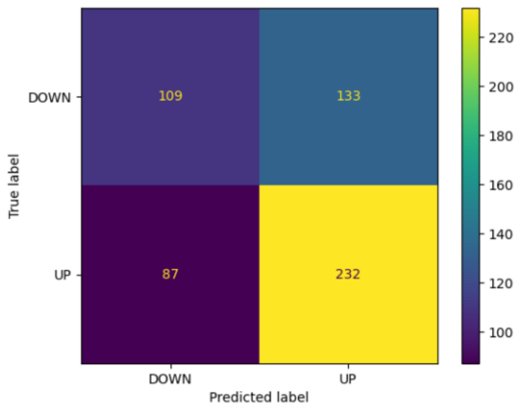


*Figure 7. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for COST*

*Figure 8. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for NFLX*



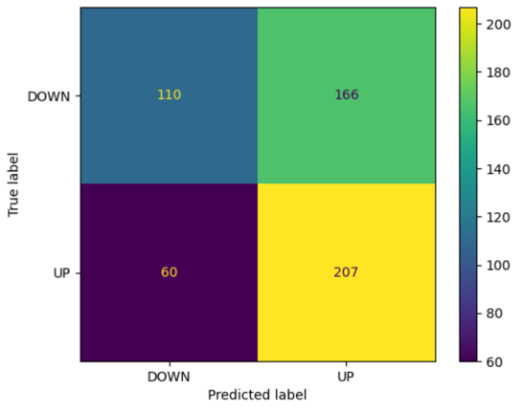


Figure 9. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for GOOGL

Figure 10. Confusion Matrix for the Stock Price Movement Prediction Using LSTM for TSLA

The **Table 2** below summarizes the performance metrics for each stock:

Table 2. Summarization Table of All Performance Metrics for Each Stock

| *Stock* | *Accuracy* | *Precision (UP)* | *Recall (UP)* | *F1-Score (UP)* |
|---|---|---|---|---|
| *AAPL* | 0.615 | 0.627 | 0.757 | 0.686 |
| *MSFT* | 0.615 | 0.622 | 0.806 | 0.702 |
| *NVDA* | 0.617 | 0.646 | 0.697 | 0.671 |
| *AMZN* | 0.608 | 0.621 | 0.699 | 0.658 |
| *AVGO* | 0.606 | 0.676 | 0.533 | 0.596 |
| *META* | 0.519 | 0.618 | 0.502 | 0.554 |
| *COST* | 0.611 | 0.640 | 0.747 | 0.689 |
| *NFLX* | 0.615 | 0.638 | 0.720 | 0.676 |
| *GOOGL* | 0.608 | 0.635 | 0.727 | 0.678 |
| *TSLA* | 0.584 | 0.555 | 0.776 | 0.647 |

### 4.1.2 Performance Analysis

The performance of the LSTM-based stock price direction forecasting model across the top 10 Nasdaq stocks reveals both strengths and limitations, providing valuable insights into its practical utility for investors. The analysis focuses on the metrics presented in the **Table 2** above, examining the model's ability to predict UP and DOWN movements, its consistency across stocks, and potential areas for improvement.

#### 4.1.2.1 Overall Performance

The model's accuracy across the 10 stocks ranges from 0.519 (META) to 0.617 (NVDA), with an average accuracy of approximately 0.6. This indicates that the model correctly predicts the direction of stock price movements in about 60% of the cases on average, which is a considerable improvement over the model presented in the last interim report, which was about 54% in accuracy on average. The test set, comprising around 566 trading days for each stock, provides a robust sample to evaluate the model's generalization to unseen data. However, the relatively low accuracy for some stocks, particularly META, suggests that the model struggles with certain stocks, likely due to differences in price movement patterns or market dynamics.

The precision for the UP class varies from 0.555 (TSLA) to 0.676 (AVGO), with an average of 0.628. This metric indicates the reliability of the model's UP predictions, which on average 62.8% of the predicted UP movements were correct. Recall for the UP class ranges from 0.502 (META) to 0.806 (MSFT), averaging 0.696, meaning the model identifies about 69.6% of actual UP movements. The F1-score, balancing precision and recall, ranges from 0.554 (META) to 0.702 (MSFT), with an average of 0.656, reflecting a reasonable trade-off between the two metrics.

#### 4.1.2.2 Class Imbalance and Prediction Bias

One notable trend across the confusion matrices is the model's tendency to favor predicting UP movements. For instance, in MSFT, the model predicts 408 UP movements (154 FP + 254 TP) compared to only 153 DOWN movements (92 TN + 61 FN). This bias is evident in the high recall for the UP class. The class weights applied during training, as implemented in the code, aim to address potential imbalances in the training data. However, the test set appears to have a slight skew toward UP movements in some stocks, which may contribute to this bias.

#### 4.1.2.3 Stock Specific Performance

The model's performance varies across the 10 stocks, reflecting differences in their price dynamics. MSFT demonstrates the high accuracy (0.615) and F1-score (0.702), which was driven by its high recall (0.806). The confusion matrix shows 254 true positives out of 315 actual UP movements, indicating strong sensitivity to upward trends. However, its precision (0.622) is lower due to 154 false positives, suggesting that the model incorrectly predicts UP movements in many DOWN cases. This stock likely exhibits strong and predictable upward trends that the LSTM captures well, possibly due to consistent market momentum.

In contrast, META has the lowest accuracy (0.516) and F1-score (0.554), with a recall of only 0.502. The confusion matrix reveals a balanced prediction pattern, but the model struggles to correctly identify both UP and DOWN movements. This stock may have more erratic price movements or weaker trends, making it challenging for the LSTM to capture temporal dependencies effectively. The Heikin-Ashi transformation, intended to smooth volatility, may not fully mitigate noise in this stock's data, leading to poor performance.

AVGO is performs the best in terms of precision (0.676), but its recall (0.533) is the second-lowest. The confusion matrix shows only 78 false positives, indicating that the model is conservative in predicting UP movements, but it misses many actual UP instances (143 FN). This suggests that AVGO may have a higher proportion of DOWN movements or more complex patterns that the model fails to capture, leading to a cautious prediction strategy.

#### 4.1.2.4 Impact of Features and Preprocessing

The model's performance is influenced by the feature engineering and preprocessing steps outlined in the methodology. The use of Heikin-Ashi candlesticks smooths price data, reducing noise and enhancing trend detection. This is particularly effective for stocks with clear trends, such as MSFT, where the high recall suggests that the model successfully identifies smoothed upward patterns. The inclusion of technical indicators like EMA, ADMI, and Stochastic K% and D% provides the LSTM with insights into trends and momentum, which likely contribute to the model's ability to detect UP movements.

Market-related features, such as the Nasdaq Index, VIX, and rolling Beta, add context about broader market dynamics. The rolling Beta measures a stock's sensitivity to market movements. Stocks with higher Beta values, which are more correlated with the market, may benefit from the Nasdaq Index feature.

Feature standardization, using StandardScaler for most features and MinMaxScaler for the adjusted closing price, ensures that all inputs contribute proportionally to the model's learning process. This step likely improves the model's stability, as evidenced by the consistent performance across most stocks, but it may not fully address the impact of outliers or extreme volatility in certain stocks like META.

#### 4.1.2.5 Model Architecture and Training

The LSTM architecture, with two stacked LSTM layers, a dense layer, and a sigmoid output, is well-suited for capturing temporal dependencies in stock price data. The use of dropout and early stopping helps mitigate overfitting, as seen in the reasonable generalization to the test set. The Grid Search over hidden units ensures that the model is optimized for each stock, which contributes to the relatively stable performance across stocks.

However, the model's bias toward UP predictions may stem from the training process. The binary cross-entropy loss may not fully address the class imbalance if the training data has a significant skew. The class weights help, but their impact appears limited, as the model still overpredicts UP movements.

#### 4.1.2.6 Limitations and Future Improvements

The model's average accuracy of 0.6 indicates room for improvement, especially for stocks like META. One limitation is the potential mismatch between the training and test periods. The training data (2010–2020) may include long-term bull markets, leading the model to overfit to UP patterns, while the test period (2023–2024) may include more volatile or bearish conditions.

Another limitation is the feature set. While the selected features capture trends, momentum, and market dynamics, they may not fully account for external factors like macroeconomic events or company-specific news, which can significantly impact stock prices.

Future improvements could include:

- **Additional Features**: Include macroeconomic indicators (e.g. interest rates) or alternative data to capture more drivers of price movements.

- **Hyperparameter Tuning**: Expand the Grid Search to include other parameters, such as dropout rate or learning rate, to further optimize the model.

#### 4.1.2.7 Practical Implications

For novice investors, the model provides a useful starting point, particularly for stocks with predictable trends. The high recall for UP movements means that the model can reliably identify upward trends, helping investors capitalize on potential gains. However, the lower precision and tendency to overpredict UP movements suggest that investors should use the model’s predictions cautiously to avoid false positives that could lead to poor investment decisions.

In conclusion, the LSTM model demonstrates moderate success in predicting stock price directions, with strengths in identifying UP movements but challenges in balancing precision and recall. Its performance varies across stocks, highlighting the importance of stock-specific dynamics in financial forecasting. Future enhancements to the feature set and training process could further improve its reliability, making it a more robust tool for novice investors navigating the volatile stock market.

## 4.2 Baseline Numerical Stock Price Prediction Model Using ARIMA

This section presents the results and analysis of the ARIMA model for numerical stock price prediction of the top 10 Nasdaq stocks by portfolio weight. The model forecasts daily closing prices, serving as a statistical baseline for comparison with deep learning approaches. The methodology includes stationarity testing, model training, and evaluation using RMSE, MAE, and MAPE metrics.

### 4.2.1 Overview of ARIMA Model Results

This section evaluates the performance of the ARIMA(7,1,0) and ARIMA(15,1,0) models for numerical stock price prediction of the top 10 Nasdaq stocks by portfolio weight. The models forecast daily closing prices using historical data from January 1, 2020, to April 17, 2025, sourced via the Yahoo! Finance API. The dataset, spanning 1,330 trading days, is split into 70% training (931 days) and 30% testing (399 days). The models use closing prices as the sole feature, with stationarity ensured through first-order differencing. Performance is assessed using RMSE, MAE, and MAPE, with results visualized through time-series plots, scatter plots, and bar charts.

### 4.2.2 Model Performance

The ARIMA(7,1,0) and ARIMA(15,1,0) models were applied to each of the top 10 Nasdaq stocks, and their performance was evaluated on the test set using three metrics:

- **Root Mean Squared Error (RMSE)**: Measures the square root of the average squared differences between predicted and actual prices.

*Equation 9. Formula for Calculating the RMSE Metric*

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

- **Mean Absolute Error (MAE)**: Calculates the average absolute differences between predicted and actual prices.

*Equation 10. Formula for Calculating the MAE Metric*

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

- **Mean Absolute Percentage Error (MAPE)**: Expresses the average absolute error as a percentage of the actual values.

*Equation 11. Formula for Calculating the MAPE Metric*

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{\hat{y}_i - y_i}{y_i}\right| \times 100$$

where $y_i$ is the actual price, $\hat{y}_i$ is the predicted price.

The performance metrics for both models are summarized in the figures below (**Figure 11-12**).

## ARIMA(7,1,0) Performance

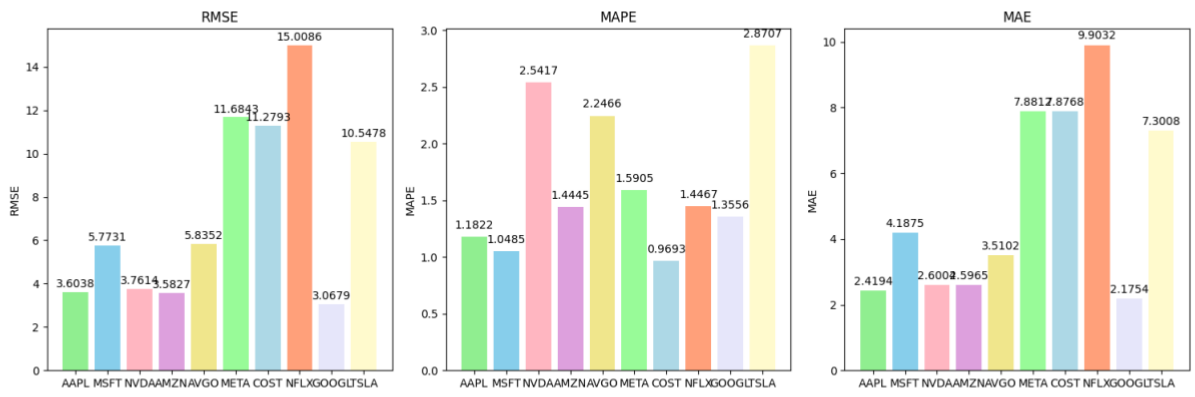


*Figure 11. The ARIMA(7, 1, 0) Model Results with RMSE, MAPE, and MAE*

## ARIMA(15,1,0) Performance

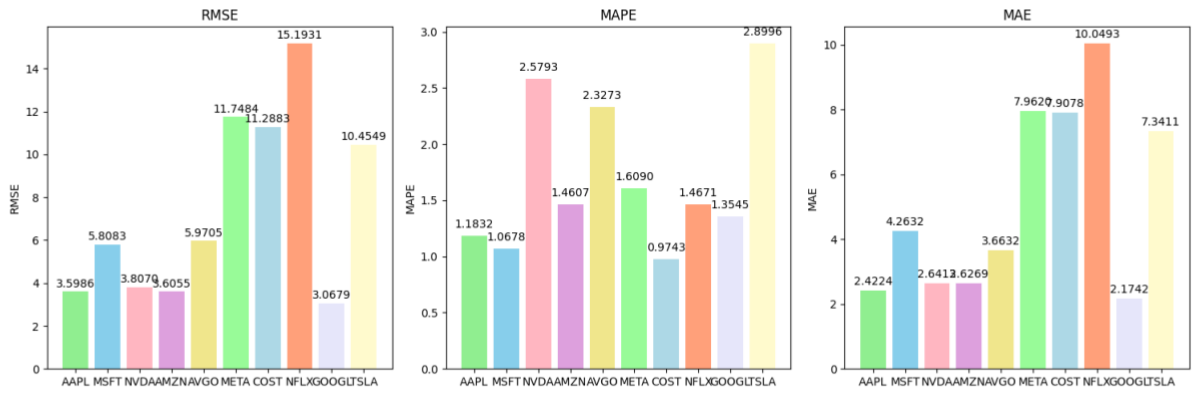


*Figure 12. The ARIMA(15, 1, 0) Model Results with RMSE, MAPE, and MAE*

### 4.2.3 Performance Analysis

The ARIMA(7,1,0) and ARIMA(15,1,0) models provide a statistical baseline for numerical stock price prediction. This analysis examines the models' accuracy, the impact of stationarity preprocessing, the effect of different look-back periods, and their practical utility as baselines for comparison with deep learning models.

#### 4.2.3.1 Stationarity and Preprocessing

Ensuring stationarity is a critical step in ARIMA modelling, as the model assumes a constant mean, variance, and autocorrelation structure. The Augmented Dickey-Fuller (ADF) test was applied to the closing price time series before and after first-order differencing, which is visualized below in the **Figure 13-14**. The p-values for the original series range from 0.1907 (TSLA) to 0.9554 (AVGO), all above the 0.05 threshold, indicating non-stationarity. This confirms that the raw stock price data exhibits trends, which could distort ARIMA's ability to capture patterns.

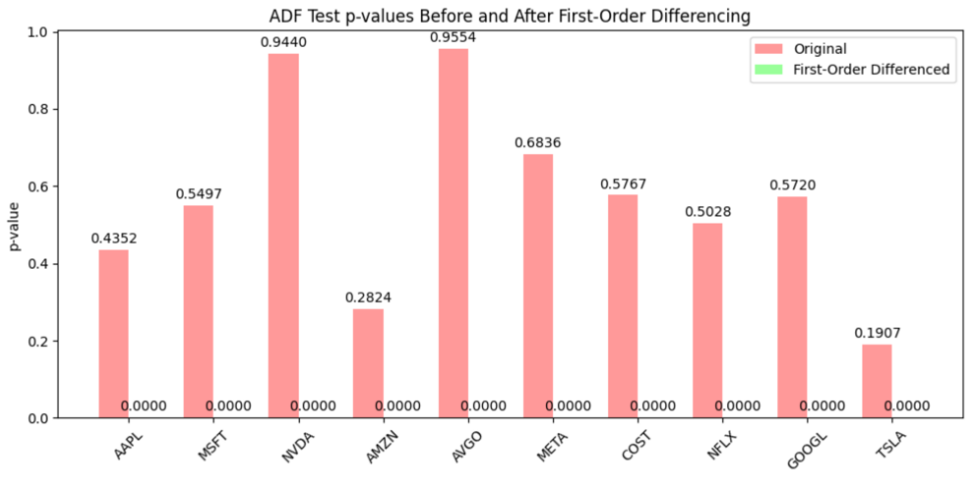


*Figure 13. The p-value of the Training Time Series Before and After First-Order Differencing Under the ADF Test*

After the first-order differencing, the p-values drop to 0.0 for all stocks, well below 0.05, rejecting the null hypothesis and confirming stationarity. This transformation, set as ($d = 1$) in both ARIMA configurations, removes trends and stabilizes the mean, aligning with ARIMA's assumptions.

### 4.2.3.2 Overall Performance Comparison

**ARIMA(7,1,0) Model**

The ARIMA(7,1,0) model's MAPE ranges from 0.9693% (COST) to 2.8707% (TSLA), averaging 1.53%. This indicates that, on average, the model's predictions deviate by about 1.53% from the actual prices, which is a reasonable error for stock price forecasting given the market's volatility. RMSE values range from 3.0679 (GOOGL) to 15.0086 (NFLX), averaging 7.41. MAE, ranging from 2.1754 (GOOGL) to 9.9032 (NFLX) with an average of 5.05.

**ARIMA(15,1,0) Model**

The ARIMA(15,1,0) model's MAPE ranges from 0.9743% (COST) to 2.8996% (TSLA), averaging 1.69%, slightly higher than the ARIMA(7,1,0) model. RMSE values range from 3.0679 (GOOGL) to 15.1931 (NFLX), averaging 7.45, also slightly higher than the 7-day model. MAE ranges from 2.1742 (GOOGL) to 10.0493 (NFLX), averaging 5.11, indicating a minor increase in compared to ARIMA(7,1,0).

**Comparative Analysis**

The ARIMA(7,1,0) model generally outperforms the ARIMA(15,1,0) model across most stocks, with lower average MAPE (1.53% vs. 1.69%), RMSE (7.41 vs. 7.45), and MAE (5.05 vs. 5.11). This suggests that a shorter look-back period of 7 days is more effective for capturing short-term price patterns in this dataset. The ARIMA(15,1,0) model's slightly higher errors may result from overfitting to longer-term patterns that are less relevant for daily forecasts.

#### 4.2.3.3 Model Limitation

Both ARIMA models rely solely on historical closing prices, lacking external features like market sentiment or macroeconomic indicators, which limits their ability to adapt to sudden market shifts. The rolling forecast approach, without refitting at each step, may cause the model to drift from optimal parameters over time, particularly during volatile periods.

The test period includes market fluctuations that challenge the models' assumptions of stationarity. For example, the stocks with high MAPE may reflect its rapid growth, which ARIMA struggles to model due to its linear nature.

#### 4.2.3.4 Practical Implications and Comparison

As baselines, the ARIMA models provide simple and interpretable benchmarks for stock price prediction. The ARIMA(7,1,0) model's average MAPE of 1.53% suggests better accuracy than the ARIMA(15,1,0) model (1.69%), making it more suitable for short-term forecasting. However, both models' performance lags behind more complex models in capturing nonlinear patterns and external influences, as evidenced by their struggles with volatile stocks like TSLA.

For practical use, the ARIMA(7,1,0) model is best suited for stocks with stable trends, where its predictions can guide short-term investment decisions. However, for volatile stocks or during market turbulence, its predictions should be supplemented with more advanced models.

#### 4.2.3.5 Future Improvements

To further enhance the ARIMA models' performance, the following can be considered in the future:

- **Dynamic Parameter Tuning**: Use auto-ARIMA to dynamically select the hyper-parameters for each stock which can potentially improve the model performance.

- **Hybrid Models**: Combine ARIMA with machine learning models to leverage both statistical and nonlinear modelling capabilities.

In conclusion, the ARIMA(7,1,0) model serves as an effective baseline, outperforming the ARIMA(15,1,0) model in most cases and achieving reasonable accuracy for numerical stock price prediction. However, it highlights the need for more sophisticated models in volatile markets, providing a foundation for evaluating deep learning approaches in subsequent analyses.

## 4.3 Numerical Stock Price Prediction Using Deep Learning Models

This section evaluates the performance of six deep learning models, which are GRU, LSTM, CNN, LSTM-BERT, CNN-LSTM, and CNN-LSTM-BERT, for numerical stock price prediction of the top 10 Nasdaq stocks over 7-day and 15-day forecast horizons. The models incorporate historical prices and BERT-derived sentiment scores. Performance is assessed using $R^2$, RMSE, MAE, and MAPE, and compared against the ARIMA baseline models.

### 4.3.1 Model Performance

The six deep learning models were evaluated on the test set for both 7-day and 15-day forecast horizons, with average performance metrics computed across the 10 Nasdaq stocks. The metrics used are:

- **Root Mean Squared Error (RMSE)**: Measures the square root of the average squared differences between predicted and actual prices (**Equation 9**).

- **Mean Absolute Error (MAE)**: Calculates the average absolute differences between predicted and actual prices (**Equation 10**).

- **Mean Absolute Percentage Error (MAPE)**: Expresses the average absolute error as a percentage of the actual values (**Equation 11**).

The results of six models over 7-day and 15-day horizons are presented below respectively (**Table 3-4**):

**<u>7-Day Forecast Horizon</u>**

*Table 3. Summarization Table of the Results of Six Deep Learning Over the 7-Day Forecast Horizon*

|  | $R^2$ | *RMSE* | *MAPE* | *MAE* |
|---|---|---|---|---|
| *GRU* | 0.8566 | 5.6708 | 5.7861 | 6.2554 |

| | | | | |
|---|---|---|---|---|
| *LSTM* | 0.7044 | 8.9912 | 8.2992 | 9.2819 |
| *CNN* | 0.2243 | 24.0836 | 11.0298 | 18.9425 |
| *LSTM-BERT* | 0.9378 | 4.0892 | 2.4456 | 2.4569 |
| *CNN-LSTM* | 0.8138 | 7.9822 | 6.7982 | 7.2444 |
| *CNN-LSTM-BERT* | 0.9243 | 4.6208 | 3.2087 | 3.5029 |

<u>**15-Day Forecast Horizon**</u>

*Table 3. Summarization Table of the Results of Six Deep Learning Over the 15-Day Forecast Horizon*

| | $R^2$ | *RMSE* | *MAPE* | *MAE* |
|---|---|---|---|---|
| *GRU* | 0.8140 | 10.4684 | 6.9489 | 13.0404 |
| *LSTM* | 0.6816 | 15.5968 | 10.4643 | 19.6708 |
| *CNN* | 0.0849 | 34.9023 | 25.0678 | 40.0567 |
| *LSTM-BERT* | 0.9106 | 6.0894 | 3.4436 | 6.0223 |
| *CNN-LSTM* | 0.8138 | 11.0895 | 7.8900 | 13.7285 |
| *CNN-LSTM-BERT* | 0.9127 | 6.2294 | 3.8825 | 7.0415 |

### 4.3.2 Performance Analysis and Comparison with ARIMA

The six deep learning models demonstrate varying levels of effectiveness in numerical stock price prediction, with improvements over the ARIMA baseline models in some cases. The analysis compares their performance against ARIMA(7,1,0) and ARIMA(15,1,0), focusing on the 7-day and 15-day forecast horizons, and justifies the selection of the LSTM-BERT model for final prediction.

#### 4.3.2.1 Deep Learning Models Performance

<u>**7-Day Forecast Horizon**</u>

For the 7-day horizon, LSTM-BERT achieves the highest performance with an $R^2$ of 0.9378, RMSE of 4.0892, MAPE of 2.4456%, and MAE of 2.4569, indicating excellent predictive accuracy and the best fit among all models. CNN-LSTM-BERT follows closely, benefiting from its multimodal architecture that integrates sentiment and price data. GRU ($R^2$ 0.8566, MAPE 5.7861%) and CNN-LSTM ($R^2$ 0.8138, MAPE 6.7982%) perform moderately well. The standalone LSTM model performs less well ($R^2$ 0.7044, MAPE 8.2992%), due to its inability to incorporate sentiment, while CNN performs the worst ($R^2$ 0.2243, MAPE 11.0298%).

**15-Day Forecast Horizon**

For the 15-day horizon, LSTM-BERT again leads with an $R^2$ of 0.9106, RMSE of 6.0894, MAPE of 3.4436%, and MAE of 6.0223, maintaining strong performance despite the longer forecast window. CNN-LSTM-BERT remains competitive ($R^2$ 0.9127, MAPE 3.8825%), while GRU and CNN-LSTM show moderate degradation in accuracy due to increased forecast complexity. The standalone LSTM and CNN perform poorly with high relative error.

#### 4.3.2.2 Comparison with ARIMA Baseline

**7-Day Forecast Comparison**

The ARIMA(7,1,0) model, as reported in Section 4.2, achieves an average MAPE of 1.53%, RMSE of 7.41, and MAE of 5.05 across the 10 stocks. In contrast, the best deep learning model, LSTM-BERT, has a MAPE of 2.4456%, which is 60% higher than ARIMA's, despite a better RMSE (4.0892) and MAE (2.4569). While deep learning models like LSTM-BERT excel in MAE and RMSE, their higher MAPE indicates that ARIMA is more accurate in relative terms.

**15-Day Forecast Comparison**

For the 15-day horizon, the ARIMA(15,1,0) model has an average MAPE of 1.69%, RMSE of 7.45, and MAE of 5.11. The best deep learning model, LSTM-BERT, reports a MAPE of 3.4436%, nearly double ARIMA's, despite a lower RMSE (6.0894) and comparable MAE (6.0223).

#### 4.3.2.3 Model Selection Rationale

Despite ARIMA's better MAPE, the LSTM-BERT model was selected for final prediction due to its superior $R^2$ and lower RMSE, and ability to incorporate sentiment data, which is critical for capturing market dynamics in volatile stocks. While ARIMA excels in MAPE, its lack of multimodal input limits its adaptability to sentiment-driven price movements, which is a key advantage of LSTM-BERT. The model's balanced performance across metrics, especially its low RMSE and MAE, makes it a practical choice for most financial scenarios.

#### 4.3.2.4 Practical Implications

The deep learning models, particularly LSTM-BERT, offer advantages in capturing nonlinear patterns and sentiment-driven movements, as reflected in their high $R^2$ and lower RMSE. However, their higher MAPE compared to ARIMA suggests that simpler statistical models may be more reliable for percentage-based accuracy. For practical use, LSTM-BERT can be employed in contexts where sentiment plays a significant role, but its predictions should be cross-validated with ARIMA to ensure accuracy. The computational complexity of deep learning models also poses scalability challenges compared to ARIMA's efficiency.

In conclusion, while ARIMA outperforms deep learning models in MAPE, the LSTM-BERT model's ability to integrate sentiment and achieve lower RMSE justifies its use for final prediction. Future work could focus on hybrid approaches combining ARIMA's accuracy with deep learning's feature richness to further optimize forecasting performance.

## 4.4 Website Development

This section shows the implementation of the stock price prediction system as a web application, designed to deliver actionable insights to investors through AI-driven forecasting. Built using the Django framework, the platform integrates the previously discussed models to provide real-time stock price forecasts, news updates, and personalized trading recommendations. The website's frontend and backend components, along with its key functionalities, are discussed below, highlighting how it empowers users to make informed financial decisions.

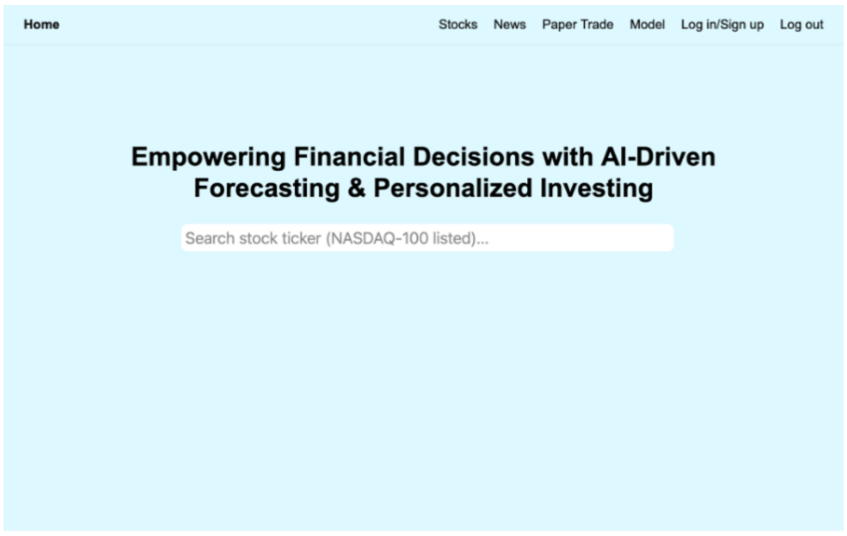
### 4.4.1 Overview of Website Implementation

The website is developed using Django for both frontend and backend with HTML, enhancing the user interface, and SQLite as the database. The platform supports multiple user interactions, from browsing stock data to executing paper trades. The implementation ensures real-time updates, user authentication, and personalized recommendations, making it a comprehensive tool for investors. Images of the website's key pages illustrate its functionality.

### 4.4.2 Frontend Design and User Interface

The frontend comprises several webpages, each serving a distinct purpose to enhance user experience:

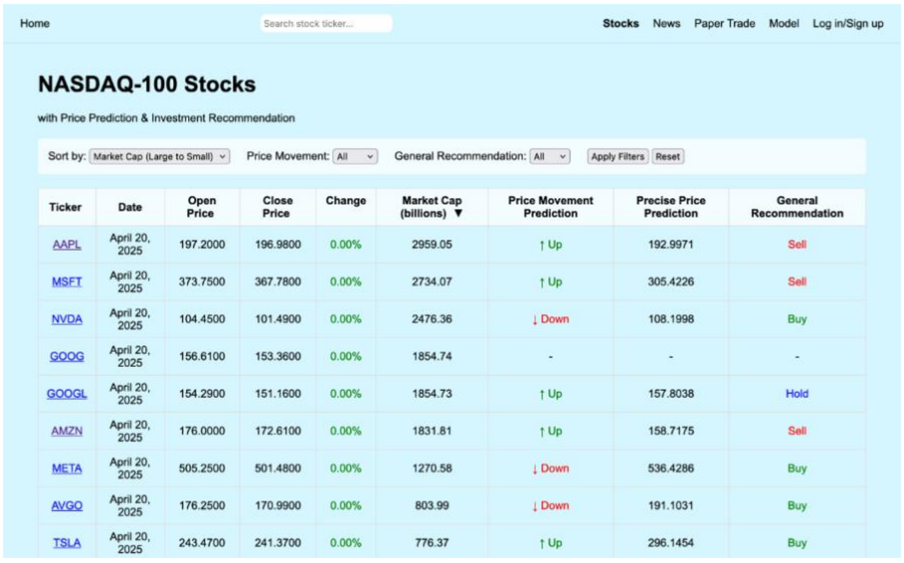
- **Home Page (/)**: The landing page introduces the platform's purpose and includes a search bar for ticker symbols, which can be seen in **Figure 14**.



Empowering Financial Decisions with AI-Driven Forecasting & Personalized Investing

*Figure 14. Home Page of Website*

- **Stock List (/stocks)**: Displays a list of NASDAQ-100 stocks with daily price updates, predictions, and recommendations. Users can sort or filter the stocks, which can be seen in **Figure 15**.



NASDAQ-100 Stocks

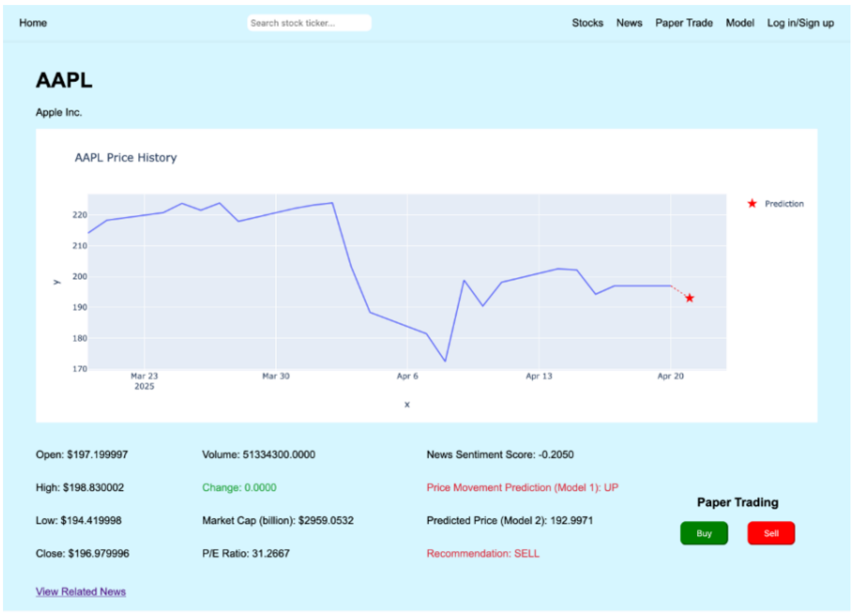
with Price Prediction & Investment Recommendation

Sort by: Market Cap (Large to Small) | Price Movement: All | General Recommendation: All | Apply Filters | Reset

| Ticker | Date | Open Price | Close Price | Change | Market Cap (billions) ▼ | Price Movement Prediction | Precise Price Prediction | General Recommendation |
|---|---|---|---|---|---|---|---|---|
| AAPL | April 20, 2025 | 197.2000 | 196.9800 | 0.00% | 2959.05 | ↑ Up | 192.9971 | Sell |
| MSFT | April 20, 2025 | 373.7500 | 367.7800 | 0.00% | 2734.07 | ↑ Up | 305.4226 | Sell |
| NVDA | April 20, 2025 | 104.4500 | 101.4900 | 0.00% | 2476.36 | ↓ Down | 108.1998 | Buy |
| GOOG | April 20, 2025 | 156.6100 | 153.3600 | 0.00% | 1854.74 | - | - | - |
| GOOGL | April 20, 2025 | 154.2900 | 151.1600 | 0.00% | 1854.73 | ↑ Up | 157.8038 | Hold |
| AMZN | April 20, 2025 | 176.0000 | 172.6100 | 0.00% | 1831.81 | ↑ Up | 158.7175 | Sell |
| META | April 20, 2025 | 505.2500 | 501.4800 | 0.00% | 1270.58 | ↓ Down | 536.4286 | Buy |
| AVGO | April 20, 2025 | 176.2500 | 170.9900 | 0.00% | 803.99 | ↓ Down | 191.1031 | Buy |
| TSLA | April 20, 2025 | 243.4700 | 241.3700 | 0.00% | 776.37 | ↑ Up | 296.1454 | Buy |

*Figure 15. Stock List Page of Website*

- **Stock Detail (/stocks/<ticker_symbol>)**: Provides detailed information for a specific stock including a price history graph with predictions, news sentiment scores, and paper trading options, which can be seen in **Figure 16**.



AAPL

Apple Inc.

Open: $197.199997

High: $198.830002

Low: $194.419998

Close: $196.979996

Volume: 51334300.0000

Change: 0.0000

Market Cap (billion): $2959.0532

P/E Ratio: 31.2667

News Sentiment Score: -0.2050

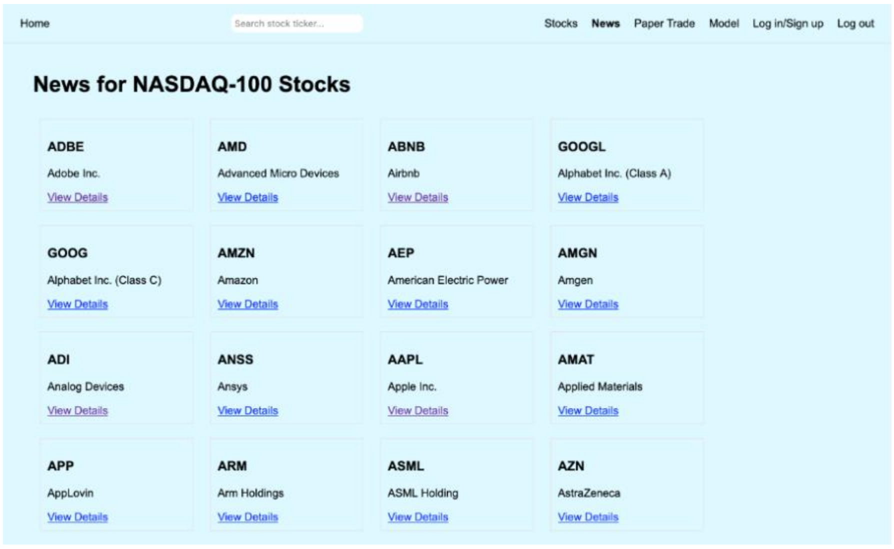
Price Movement Prediction (Model 1): UP

Predicted Price (Model 2): 192.9971

Recommendation: SELL

*Figure 16. Stock Detail Page of Website*

- **News (/news)**: Summarizes news for NASDAQ-100 stocks, with links to view details for each ticker, which can be seen in **Figure 17**.



News for NASDAQ-100 Stocks

ADBE
Adobe Inc.
View Details

AMD
Advanced Micro Devices
View Details

ABNB
Airbnb
View Details

GOOGL
Alphabet Inc. (Class A)
View Details

GOOG
Alphabet Inc. (Class C)
View Details

AMZN
Amazon
View Details

AEP
American Electric Power
View Details

AMGN
Amgen
View Details

ADI
Analog Devices
View Details

ANSS
Ansys
View Details

AAPL
Apple Inc.
View Details

AMAT
Applied Materials
View Details

APP
AppLovin
View Details

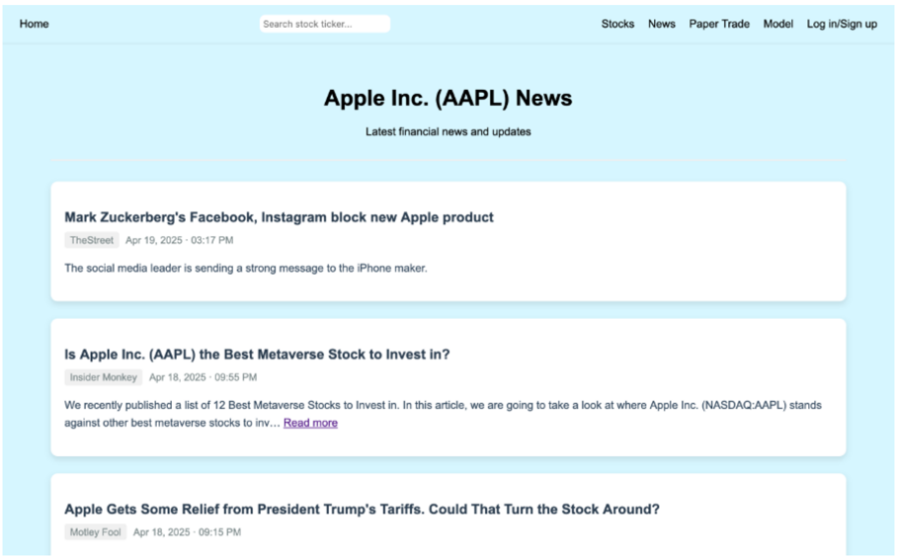
ARM
Arm Holdings
View Details

ASML
ASML Holding
View Details

AZN
AstraZeneca
View Details

*Figure 17. News Page of Website*

- **News Detail (/news/<ticker_symbol>)**: Shows news articles for a specific stock, which can be seen in **Figure 18**.


Home | Search stock ticker... | Stocks | News | Paper Trade | Model | Log in/Sign up

Apple Inc. (AAPL) News

Latest financial news and updates

Mark Zuckerberg's Facebook, Instagram block new Apple product

TheStreet Apr 19, 2025 · 03:17 PM

The social media leader is sending a strong message to the iPhone maker.

Is Apple Inc. (AAPL) the Best Metaverse Stock to Invest in?

Insider Monkey Apr 18, 2025 · 09:55 PM

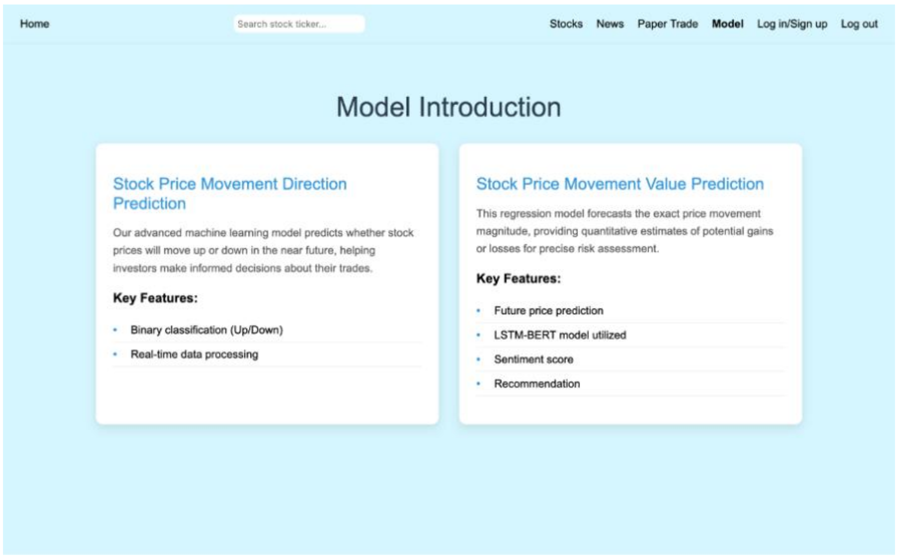
We recently published a list of 12 Best Metaverse Stocks to Invest in. In this article, we are going to take a look at where Apple Inc. (NASDAQ:AAPL) stands against other best metaverse stocks to inv… Read more

Apple Gets Some Relief from President Trump's Tariffs. Could That Turn the Stock Around?

Motley Fool Apr 18, 2025 · 09:15 PM

*Figure 18. News Detail Page of Website*

- **Model Introduction (/model)**: Introduces the two primary models used, highlighting their key features like real-time processing and sentiment integration, which can be seen in **Figure 19**.


Home | Search stock ticker... | Stocks | News | Paper Trade | Model | Log in/Sign up | Log out

Model Introduction

Stock Price Movement Direction Prediction

Our advanced machine learning model predicts whether stock prices will move up or down in the near future, helping investors make informed decisions about their trades.

Key Features:

- Binary classification (Up/Down)
- Real-time data processing

Stock Price Movement Value Prediction

This regression model forecasts the exact price movement magnitude, providing quantitative estimates of potential gains or losses for precise risk assessment.

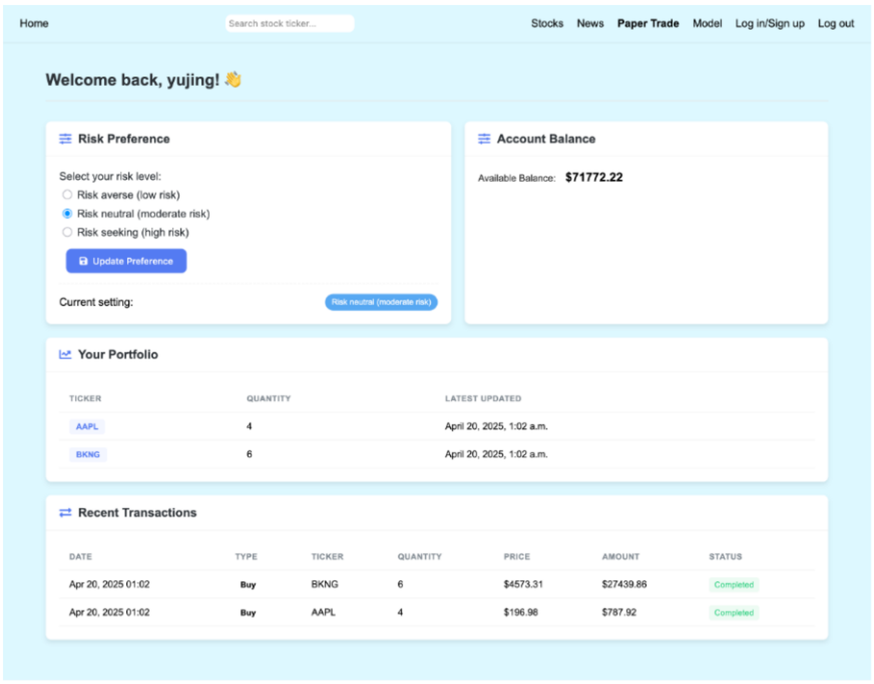
Key Features:

- Future price prediction
- LSTM-BERT model utilized
- Sentiment score
- Recommendation

*Figure 19. Model Introduction Page of Website*

- **User Authentication (/register, /login, /logout)**: Allows users to register, log in, and log out, with authentication required for paper trading.

- **Paper Trading Dashboard (/dashboard)**: Enables users to set risk preferences (low, moderate, high), view their account balance, and transaction history, which can be seen in **Figure 20**.



Welcome back, yujing! 👋

Risk Preference

Select your risk level:
( ) Risk averse (low risk)
(●) Risk neutral (moderate risk)
( ) Risk seeking (high risk)

Current setting: Risk neutral (moderate risk)

Account Balance

Available Balance: $71772.22

Your Portfolio

| TICKER | QUANTITY | LATEST UPDATED |
|---|---|---|
| AAPL | 4 | April 20, 2025, 1:02 a.m. |
| BKNG | 6 | April 20, 2025, 1:02 a.m. |

Recent Transactions

| DATE | TYPE | TICKER | QUANTITY | PRICE | AMOUNT | STATUS |
|---|---|---|---|---|---|---|
| Apr 20, 2025 01:02 | Buy | BKNG | 6 | $4573.31 | $27439.86 | Completed |
| Apr 20, 2025 01:02 | Buy | AAPL | 4 | $196.98 | $787.92 | Completed |

*Figure 20. Paper Trading Dashboard Page of Website*

The frontend design prioritizes usability, with intuitive navigation and visual elements like price trend graphs and actionable buttons for paper trading.

### 4.4.3 Backend Architecture and Database Design

The backend, powered by Django, manages data processing, model deployment, and user interactions. The SQLite database includes eight tables:

- **NasdaqTicker**: Stores NASDAQ-100 stock details (ticker, company name, sector), with ticker as the primary key.

- **DailyQuote**: Records daily stock data fetched, using ticker and date as the composite primary key.

- **StockNews**: Stores news articles with fields like ticker, title, and summary, linked to NasdaqTicker via a foreign key.

- **PredictionResult**: Holds daily predictions, including sentiment scores, predicted prices, and recommendations, with ticker and date as the primary key.

- **UserProfile, TradingAccount, Position, TradeRecord**: Manage user data, account balances, stock positions, and trading records, supporting personalized features.

The backend ensures data integrity and efficient retrieval.

### 4.4.4 Key Functionalities

- **Real-Time Updates**: Using Django-crontab, the system updates stock prices and predictions at 5 AM HKT on weekdays, aligning with the US market close at 4 AM HKT, and news updates occur daily at 8 AM and 8 PM. This ensures users receive the latest information, critical for timely decision-making in a dynamic market.

- **Model Deployment**: The trained models are exported as HDF5 files, with scalers saved as pickle files, allowing daily predictions without retraining. A stock news update function fetches recent news into CSV files, which are processed to generate sentiment scores and predictions stored in the PredictionResult table.

- **Recommendation System**: For unregistered users, the system provides general recommendations based on expected returns: buy if $> 1.5\%$ , sell if $< -1.5\%$, else hold. Registered users receive personalized recommendations based on risk preferences. For example, risk-averse users are recommended to buy only if the return exceeds $5\%$ or the predicted direction is UP with $> 0.8$ probability, while thresholds for risk-neutral and risk-seeking users are adjusted accordingly (e.g., 3% and 0.6 for neutral).

- **User Authentication**: The login/signup system, built using Django's User class, ensures secure access to paper trading features. Users can set risk preferences, view their portfolio, and execute trades.

### 4.4.5 Practical Implications and Future Enhancements

The website successfully delivers a user-friendly platform for stock price prediction, combining real-time data, AI-driven forecasts, and paper trading capabilities. It empowers investors by providing actionable insights, such as personalized recommendations. However, future enhancements could include integrating more data sources, optimizing model inference speed, and expanding the paper trading system to include real-time market simulations. Overall, the platform demonstrates the practical application of the developed machine learning models, bridging the predictive AI analytics with investor decision-making.

# 5 Future Work

This chapter outlines the future enhancements for the AI-driven financial management platform, addressing the limitations identified in the results and discussion. The future work is structured into several subsections, each focusing on a specific area of improvement to enhance the platform's predictive accuracy, user experience, and scalability.

## 5.1 Model Tuning and Optimization

The LSTM and LSTM-BERT models, while effective, exhibit biases and limitations in handling volatile stocks like META and TSLA. Future work will focus on advanced hyperparameter tuning, expanding Grid Search to include learning rates, dropout rates, and layer configurations, to reduce prediction biases. Additionally, optimizing the feature set by incorporating macroeconomic indicators and alternative data will improve the model robustness. This enhancement aims to increase predictive accuracy across diverse market conditions.

## 5.2 Hybrid Model Development

The ARIMA model's superior MAPE compared to LSTM-BERT highlights the potential of statistical methods, while deep learning excels in capturing nonlinear patterns. Future work will explore hybrid models combining ARIMA's statistical strengths with LSTM-BERT's sentiment-driven capabilities. Techniques like sequential modelling will be investigated to leverage both approaches' strengths. This hybrid approach aims to achieve lower MAPE while retaining high $R^2$, providing a more balanced forecasting tool for investors.

## 5.3 Enhanced Sentiment Analysis

The current sentiment analysis, reliant on financial news, misses broader market sentiments from sources like social media. Future work will expand sentiment data sources, integrating platforms to capture real-time investor sentiment. This enhancement will improve the

LSTM-BERT model's ability to reflect market mood swings, particularly for volatile stocks, enhancing the overall performance of numerical predictions.

## 5.4 Real-Time Market Simulation in Paper Trading

The paper trading system currently uses static daily updates. Future work will integrate real-time market simulations by leveraging various APIs to provide live price feeds during market hours. This will enable users to practice trading under dynamic conditions, with the system updating portfolio values and recommendations in real time.

## 5.5 User Feedback and Iterative Refinement

The current platform lacks direct user feedback mechanisms to assess its effectiveness for novice investors. Future work will implement feedback forms and usage analysis data on the website to gather user insights on usability, prediction accuracy, and recommendation relevance.

## 5.6 Final Deployment

Following the above enhancements, the platform will undergo comprehensive testing, including the user acceptance tests for the interface. The final deployment will involve releasing the website to the public. Documentation, including user guides and model explanations, will be provided to ensure accessibility. This step aims to transition the platform from a prototype to a widely-used tool, fulfilling the project's goal of empowering financial decision-making for a broader audience.

# 6 Conclusion

This project successfully developed an AI-driven financial management platform to empower novice investors by addressing key challenges in the stock market, such as information overload, complexity, and the lack of personalized guidance. The platform integrates three predictive models, which are the LSTM for stock price movement direction, ARIMA for baseline numerical forecasting, and the LSTM-BERT for final numerical predictions, delivering actionable insights for the top 10 Nasdaq stocks. A fully functional Django-based website unifies these models, providing real-time stock updates, news, personalized recommendations, and a paper trading system, creating a comprehensive tool tailored for retail investors with limited experience.

The results highlight the strengths and limitations of each component. The LSTM model achieved a moderate accuracy of 60% in directional forecasting, performing well for stocks with stable trends but struggling with volatile. The ARIMA(7,1,0) model provided a reliable baseline with a MAPE of 1.53%, outperforming deep learning models in relative accuracy, while LSTM-BERT excelled in absolute metrics ($R^2$ 0.9378, RMSE 4.0892 for 7-day forecasts), leveraging sentiment analysis to capture market dynamics. The website enhances user engagement by offering intuitive features like stock detail pages.

Despite these achievements, several limitations remain that impact the platform's effectiveness. The models' reliance on historical data and financial news misses broader market influences, such as macroeconomic factors or geopolitical events, which could enhance prediction accuracy. Additionally, the deep learning models, particularly LSTM-BERT, face scalability challenges due to their computational complexity, which may hinder performance as user demand grows. The sentiment analysis is limited by its focus on news data, potentially overlooking other sentiment sources like social media that could provide a more comprehensive market perspective.

Looking ahead, future work outlined in **Section 5** aims to address these gaps through targeted improvements. Hybrid model development combining ARIMA and LSTM-BERT, enhanced sentiment analysis incorporating social media, and real-time market simulations will strengthen the platform's capabilities. These enhancements, along with user feedback integration and performance optimization, will ensure the platform evolves to meet the needs of most investors. Overall, this project demonstrates the practical utility of AI in financial forecasting, providing a user-friendly platform that enhances investment strategies and fosters greater engagement among retail investors in the stock market, laying a strong foundation for future advancements.

# References

[1] E. F. Fama, “Efficient Capital Markets: A review of theory and empirical work,” The Journal of Finance, vol. 25, no. 2, p. 383, May 1970. doi:10.2307/2325486

[2] B. G. Malkiel, A Random Walk down Wall Street. New York: Norton, 1973.

[3] S. J. Brown, W. N. Goetzmann, and A. Kumar, “The dow theory: William Peter Hamilton’s track record reconsidered,” The Journal of Finance, vol. 53, no. 4, pp. 1311–1333, Aug. 1998. doi:10.1111/0022-1082.00054

[4] Bloomberg, “Bloomberg,” Bloomberg.com, 2023. https://www.bloomberg.com/ (accessed Nov. 10, 2024).

[5] “HKEX,” www.hkex.com.hk. https://www.hkex.com.hk/ (accessed Nov. 12, 2024).

[6] Yahoo Finance, “Yahoo Finance - Business Finance, Stock Market, Quotes, News,” Yahoo Finance, 2024. https://finance.yahoo.com/ (accessed Nov. 11, 2024).

[7] “Webull - Investing in Stocks, Trading, Online Broker and Research the Market,” Webull. https://www.webull.com/ (accessed Nov. 11, 2024).

[8] “FUTUBULL - Hong Kong,US and China Connect Stocks,” Futunn.com, 2024. https://www.futunn.com/en (accessed Nov. 11, 2024).

[9] “Longbridge HK - Open U.S. Stock Account, Open Hong Kong Stock Account, Hong Kong Stock Market Data, U.S. Stock Market Data - Longbridge - Longbridge Securities (Hong Kong),” Longbridge.com, 2023. https://longbridge.com/hk/ (accessed Nov. 11, 2024).

[10] Wall Street Prep, "Bloomberg vs. Capital IQ vs. FactSet vs. Refinitiv," 2024. \[Online\]. Available: https://www.wallstreetprep.com/knowledge/bloomberg-vs-capital-iq-vs-factset-vs-thomson-reuters-eikon/.

[11] B. M. Barber and T. Odean, "The Behavior of Individual Investors," in Handbook of the Economics of Finance, G. M. Constantinides, M. Harris, and R. M. Stulz, Eds., vol. 2, Amsterdam, The Netherlands: Elsevier, 2013, pp. 1533–1570, doi: 10.1016/B978-0-44-459406-8.00022-6.

[12] S. Gu et al., "FinBERT-LSTM: Sentiment-Driven Stock Prediction," IEEE Trans. Comput. Intell., vol. 20, no. 1, pp. 45–59, 2024.

[13] P. Zong and Q. Zhou, "MSGCA: Multi-Modal Stock Prediction Model," J. Financ. Data Sci., vol. 10, no. 2, pp. 123–140, 2023.

[14] S. Selvin, R. Vinayakumar, E. A. Gopalakrishnan, V. K. Menon, and K. P. Soman, “Stock price prediction using LSTM, RNN and CNN-sliding window model,” in Proc. 2017 Int. Conf. Advances Comput., Commun. Informat. (ICACCI), Udupi, India, Sep. 2017, pp. 1643–1647, doi: 10.1109/ICACCI.2017.8126078.

[15] S. Mohan, S. Mullapudi, S. Sammeta, P. Vijayvergia, and D. C. Anastasiu, “Stock price prediction using news sentiment analysis,” 2019 IEEE Fifth International Conference on Big Data Computing Service and Applications (BigDataService), Apr. 2019. doi:10.1109/bigdataservice.2019.00035.

[16] Q. Ding, S. Wu, H. Sun, J. Guo, and J. Guo, “Hierarchical multi-scale gaussian transformer for stock movement prediction,” Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, pp. 4640–4646, Jul. 2020. doi:10.24963/ijcai.2020/640