University of Hertfordshire UH

School of Physics, Engineering and Computer Science

MSc Data Science Project
7PAM2002-0206-2023

Department of Physics, Astronomy and Mathematics

Data Science FINAL PROJECT REPORT

Project Title:

Stock Selection and Comparative Analysis of Predictive Models for S&P 500 Future Price Forecasting

Student Name and SRN:

TOM THOMAS (22008590)

Supervisor: Dr Stephen Kane

Date Submitted: 02-05-2024

Word Count: 9660

DECLARATION STATEMENT

This report is submitted in partial fulfilment of the requirement for the degree of Master of Science in Data Science at the University of Hertfordshire.

I have read the guidance to students on academic integrity, misconduct and plagiarism information at **Assessment Offences and Academic Misconduct** and understand the University process of dealing with suspected cases of academic misconduct and the possible penalties, which could include failing the project module or course.

I certify that the work submitted is my own and that any material derived or quoted from published or unpublished work of other persons has been duly acknowledged. (Ref. UPR AS/C/6.1, section 7 and UPR AS/C/5, section 3.6). I have not used chatGPT, or any other generative AI tool, to write the report or code (other than where I have declared or referenced it).

I did not use human participants or undertake a survey in my MSc Project.

I hereby give permission for the report to be made available on module websites provided the source is acknowledged.

Student Name printed: **TOM THOMAS**

Student Name Signature:

[illegible]

Student SRN number: **22008590**

UNIVERSITY OF HERTFORDSHIRE

SCHOOL OF PHYSICS, ENGINEERING AND COMPUTER SCIENCE

ABSTRACT

This study evaluates the predictive capabilities of three types of deep learning models Gated Recurrent Unit, Bidirectional Long Short-Term Memory and Long Short-Term Memory which were tested in this study to see how well they could predict the future stock values of the top 5 companies in the Standard & Poor's 500 Index (S&P 500) index across different sectors. By carefully analysing the Long Short-Term Memory model using Root Mean Squared Error, Mean Absolute Error and prediction plots I found that it regularly produced robust forecasts and closely mirrored historical price movements, demonstrating higher accuracy. Financial forecasting in the unpredictable stock market can greatly benefit from the research's emphasis on Long Short-Term Memory models' ability to capture long-term dependency. According to the findings, combining Long Short-Term Memory with technical indicators like the Relative Strength Index and Moving Average Convergence Divergence could result in powerful tools for risk management and strategic investment planning. This work adds to the growing body of research supporting the use of deep learning models for stock price prediction, which in turn helps analysts and investors make better judgements.

ACKNOWLEDGEMENTS

I consider it a privilege to express my sincere gratitude and respect to all those who guided and inspired us in the successful completion of this project work.

I sincerely thank my project guide **Dr Stephen Kane**, Professor, for his consistent guidance and inspiration throughout the period for the completion of this project.

I would like to thank all the teaching and non-teaching staff of the University of Hertfordshire, for their valuable guidance and suggestions rendered during the project.

Finally, I thank my parents and all my friends for their help, encouragement and moral support given to us during the course of this work.

TOM THOMAS

Contents

1. Introduction **07**
1.1 Aim 08
1.2 Background 08
1.3 Objective 08

2. Literature Review **10**

3. Dataset **12**
3.1 Data Source 12
3.2 Data Collection 12
3.3 Usage in the Project 14
3.4 Justification for the Dataset 14
3.5 Exploratory Data Analysis (EDA) 14
3.5.1 Loading and Cleaning the Data: 14
3.5.2 Checking for Missing Data: 15
3.5.3 Removing Unnecessary Features: 15
3.5.4 Feature Engineering: 15
3.6 Ethical Considerations 15

4. Methodology **16**
4.1 Data Preprocessing 16
4.1.1 Data Cleaning: 16
4.1.2 Feature Engineering: 16
4.1.3Normalization: 16
4.2 Stock Selection 16
4.3 Building Deep Learning Models 19
4.3.1 LSTM Model 19
4.3.2 Bi-LSTM Model 20
4.3.3 GRU Model 21

5. Results **21**
5.1 META Stock Prediction with LSTM Model 21
5.2 TSLA Stock Prediction with LSTM Model 22
5.3 UNH Stock Prediction with LSTM Model 23
5.4 WMT Stock Prediction with LSTM model 24
5.5 XOM Stock Prediction with LSTM model 25
5.6 META Stock Prediction with Bi-LSTM model 26
5.7 TSLA Stock Prediction with Bi-LSTM model 27
5.8 UNH Stock Prediction with Bi-LSTM model 28
5.9 WMT Stock Prediction with Bi-LSTM model 29
5.10 XOM Stock Prediction with Bi-LSTM model 30
5.11 META Stock Prediction with GRU Model 31
5.12 TSLA Stock Prediction with GRU Model 32
5.13 UNH Stock Prediction with GRU Model 33
5.14 WMT Stock Prediction with GRU Model 34
5.15 XOM Stock Prediction with GRU Model 35

6. Analysis and discussion **36**
6.1 META Stock Prediction 36
6.2 TSLA Stock Prediction 38
6.3 UNH Stock Prediction 39
6.4 WMT Stock Prediction 40
6.5 XOM Stock Prediction 42
6.6 Summary 43

7. Conclusion **43**
7.1 Drawbacks and Future Work 43

8. References **44**

9. Appendices **45**

1. Introduction

Figuring out how much a stock will be worth in the future is very important in the financial industry. To handle a portfolio, evaluate risk, and choose where to invest, accurate projections are helpful. Market traders, experts, and investors are always trying to stay ahead of the movement. The S&P 500 index, which is made up of 500 of the biggest companies in a wide range of fields, is a very good signal of how the American stock market is doing. So, I chose to focus on predicting stock prices in this index.

Within my research, I look into how well deep learning models can predict the future values of S&P 500 index stocks. My three favourite models for handling time series data and spotting complex patterns are Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), and Bidirectional Long Short-Term Memory (Bi-LSTM). When trying to figure out how stock prices change, you need this exact information.

For variety's sake and to reduce association, I chose five S&P 500 companies from five different industry groups. By doing this, I can see what's going on in the market as a whole and not just focus on trends that are happening in the same places. I want to find the most accurate deep-learning model for predicting stock prices by looking at these companies.

This project compares these models' forecasting abilities for individual stocks in the S&P 500 index as its main goal. Through the analysis of a chosen sample of businesses from different industries, my goal is to identify which model provides the best accurate forecasts. Collecting historical stock price data, preparing it for deep learning models, training the models, and evaluating the models' accuracy with a variety of metrics are all part of this comparative analysis.

But my work isn't just about testing the models. I also want to understand what these predictions mean for investors and financial professionals. Knowing the strengths and limitations of these models can help improve stock price forecasting. Plus, the insights from my project could inspire more research in this field and contribute to a better understanding of how to predict stock prices.

The evolving landscape of financial markets, characterized by volatility and rapid changes, has necessitated the development of advanced predictive models to ensure better accuracy in stock price forecasting. Extensive research by Gao et al. (2021), Ji (2022), and Chaajer et al. (2021) has highlighted the effectiveness of deep learning techniques such as LSTM, GRU, and CNN in navigating the complexities of the stock market. These studies emphasize the integration of machine learning with traditional financial analysis to enhance predictive capabilities, addressing both linear and nonlinear dynamics of stock prices. Gao et al. (2021) talked about how dimension reduction methods like LASSO and PCA can help models work better, and Ji (2022) said that LSTM is the best way to find long-term dependencies that are needed for accurate market predictions. Chaajer et al. (2021) also showed how advanced algorithms like GRU could fix issues with older neural network models by preventing problems like gradient disappearance and providing accurate predictions. Together, these studies form the bedrock of the current project's methodology, aiming to harness these advanced deep learning models to predict the future values of S&P 500 index stocks, thereby providing valuable insights for market traders and investors.

My project is organised so that you can see the whole scope of my work. The context is established in the introduction, previous research in this field is reviewed in the literature review, and my data collection, processing, and model-building procedure is covered in depth in the methods section. My findings are presented and their implications are examined in the results and discussion sections. The conclusion brings everything together and makes some recommendations for possible directions for future study.

With this study, I hope to add something valuable to the ongoing discussion about financial forecasting and offer insights that can help investors and analysts make the most of deep-learning models when predicting stock prices.

1.1 Aim

To analyse historical S&P 500 stock data and evaluate predictive models for accurate stock price forecasting. Through dataset analysis, literature review, and model evaluation, the project seeks to identify effective approaches for informing investment decisions in financial markets.

1.2 Background

With prices impacted by a wide range of variables, including business results, political developments, economic trends, and investor sentiment at any one time, the stock market is a dynamic environment. Stock price predictions have long been made by analysts and investors using conventional techniques. Technical analysis examines price patterns and market movements, while fundamental analysis examines a company's earnings, financial statements, and other data to determine its value.

Despite having aided investors in navigating the market, these traditional methods have drawbacks, particularly in the fast-paced and unpredictably changing market of today. A lot of people in the finance industry are now looking for newer, more advanced solutions to deal with non-linear patterns and market volatility.

1.3 Objective

This study attempts to evaluate the predictive power of different deep learning models for S&P 500 index stocks. Finding the most accurate forecasting method among these approaches is my goal; I will be focusing on the GRU, Bi-LSTM, and LSTM models. Further, by improving their decision-making processes, these projections may benefit investors and financial professionals, as this research attempts to understand.

To achieve this objective, I will answer two key research questions:

I. **What is the most effective deep learning model, LSTM, Bi-LSTM, or GRU, for predicting the stock values of different businesses listed in the S&P 500?**

   The objective of this inquiry is to assess and contrast the precision of deep learning models when applied to a variety of companies listed in the S&P 500 index. The performance of these models will be evaluated using Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) as key metrics. Additionally, I will use a line graph to visually assess the accuracy of the predictions by concealing one month's known stock price data and predicting the next two months, including the hidden month's data. If the predicted values align with the actual prices for the hidden month, it suggests that the model's forecast for the following month is likely to be accurate. This combined approach, using both quantitative metrics and visual validation, provides a robust means to evaluate the models' ability to predict stock prices in the S&P 500, enhancing my understanding of their effectiveness in financial forecasting.

II. How can accurate stock price predictions from deep learning models help investors make better decisions about their investments?

This question explores the practical applications of stock price predictions. It aims to understand how these forecasts can guide investors in managing risk, building portfolios, and shaping their investment strategies.

These research questions and the overall objective will help guide my analysis of deep learning models and their implications in the financial sector. By answering these questions, I hope to contribute valuable insights to the field of stock price prediction and offer useful information for those involved in investment and financial planning.

2. Literature Review

This literature review explores recent advancements in stock market prediction, emphasizing the role of deep learning models like LSTM, Bi-LSTM, and GRU. These technologies have enhanced the accuracy of financial forecasts by effectively capturing complex patterns in volatile markets, illustrating their potential to transform financial analysis and strategy development.

The research conducted by Sunny et al.,(2020) presents a comprehensive investigation into the application of deep learning techniques for stock price prediction, focusing particularly on the utilization of Recurrent Neural Network (RNN) models such as LSTM and BI-LSTM. Their proposed framework offers a novel approach to forecasting stock trends, emphasizing the importance of proper hyper-parameter tuning for achieving high prediction accuracy. Through extensive simulations and analysis, the study reveals that both LSTM and BI-LSTM models exhibit promising performance in predicting future stock prices, with BI-LSTM demonstrating superior accuracy, particularly when trained with a higher number of epochs. The findings suggest that BI-LSTM's incorporation of backward propagation enhances its predictive capabilities, enabling it to mitigate underfitting issues encountered by the LSTM model. Moreover, the research highlights the significance of parameter adjustments, such as the number of hidden layers and units, in optimizing model performance. Notably, BI-LSTM models with reduced training time and lower RMSE emerge as promising candidates for practical stock market forecasting applications. This study underscores the growing influence of deep learning algorithms in developing accurate time-series prediction models, offering valuable insights for investors and stakeholders seeking to capitalize on predictive analytics in financial markets.

The study by Gao et al.,(2021) delves into the realm of stock market prediction, addressing the challenges posed by the complexities of modern financial markets. Traditionally, methods such as K-line diagrams have been employed for stock trend analysis, but these fail to capture the intricate interplay of factors influencing stock prices. In response, the authors propose a novel approach to optimize stock forecasting by integrating different technical indicators and financial data, while employing dimension reduction techniques such as Least Absolute Shrinkage and Selection Operator (LASSO) and Principal Component Analysis (PCA). The research conducts a comparative analysis of LSTM and GRU models under different parameters, shedding light on their efficacy in stock market prediction. Remarkably, the experiments reveal that both LSTM and GRU models exhibit efficient prediction capabilities, with no clear superiority of one over the other. This research contributes to the ongoing discourse on stock market prediction by providing valuable insights into the effectiveness of deep learning models and dimensionality reduction techniques in forecasting stock trends accurately.

The paper by Ji et al.,(2022) addresses the critical domain of stock price prediction, an area of profound interest and significance in both academic and practical contexts. With the proliferation of artificial intelligence (AI), the landscape of prediction algorithms in equity market analysis has expanded rapidly. This study investigates three prominent models SVM, LSTM, and AutoRegressive Integrated Moving Average (ARIMA) to forecast stock prices over varying time horizons. Empirical results are presented to demonstrate the efficacy and relevance of these models in capturing market dynamics. Through a thorough analysis of prediction techniques and model limitations, the paper not only highlights the challenges inherent in stock price forecasting but also provides insights into future research directions. It suggests that the integration of diverse indicators and the exploration of additional models such as Generalized Autoregressive Conditional Heteroskedasticity (GARCH) and multi-factor regression hold promise for enhancing investment strategies and maximizing returns. Ultimately, the study underscores the pivotal role of stock market research in driving economic growth and offers valuable implications for stakeholders navigating financial markets.

The paper authored by Chhajer et al., (2021) delves into the realm of stock market prediction, leveraging the power of machine learning and artificial intelligence (AI) to navigate the unpredictable terrain of financial markets. In an environment characterized by volatility and uncertainty, the utilization of advanced technologies for predictive analytics offers a promising avenue for investors seeking informed decision-making. This study provides a comprehensive overview of machine learning applications in the stock market, evaluating the strengths, weaknesses, opportunities, and threats associated with these methodologies. Specifically, the paper examines the efficacy of three prominent machine learning technologies artificial neural networks (ANN), SVM, and LSTM in predicting stock market trends. Through a critical analysis of practical experiments conducted with each model, the paper elucidates the transformative potential of machine learning in revolutionizing investment strategies. By presenting empirical evidence and insights into the capabilities of these models, the study underscores the imperative for embracing machine learning technologies and their profound impact on reshaping the landscape of investment dynamics. Sections dedicated to each neural network provide in-depth discussions, accompanied by summaries of practical experiments, thereby offering readers a comprehensive understanding of the applications and effectiveness of machine learning in stock market prediction. Overall, the paper advocates for the integration of machine learning into investment practices, highlighting its ability to enhance decision-making processes and optimize investment outcomes in the ever-evolving world of finance.

The paper authored by Venikar et al., (2022) proposes an innovative approach to stock market prediction by leveraging LSTM networks. Recognizing the complex and dynamic nature of stock market data, the study employs LSTM, a type of recurrent neural network known for its ability to capture long-term dependencies in sequential data. By utilizing past information, the model aims to accurately forecast stock prices, with a focus on the next 30 days. The paper highlights the challenges associated with stock market prediction, emphasizing the importance of machine learning techniques in navigating the volatile and unpredictable nature of financial markets. Furthermore, the study explores potential avenues for future research, including the incorporation of bi-directional LSTM and the development of real-world applications for the deployed model. In conclusion, the paper underscores the suitability of LSTM for stock price forecasting, offering insights into its effectiveness and potential for enhancing investment strategies. Through empirical validation and future scope discussions, the study contributes to advancing the field of stock market prediction, paving the way for more accurate and reliable forecasting models.

The literature review highlights significant advancements in the field of stock market prediction through deep learning models, particularly LSTM, Bi-LSTM, and GRU. These studies collectively underscore the importance of meticulous model configuration, including hyperparameter tuning and the integration of dimension reduction techniques, to enhance prediction accuracy. Notably, Bi-LSTM models excel due to their ability to capture temporal relationships in both directions, which proves crucial for detailed trend analysis in volatile markets. Additionally, the exploration of GRU models offers a streamlined, yet effective alternative to LSTM, showing comparable performance without the architectural complexity. These insights demonstrate the potential of deep learning technologies to transform investment strategies and market analysis, paving the way for further research into more sophisticated models and their integration with traditional financial indicators to boost predictive performance.

3. Dataset

This section describes the dataset used in my project, focusing on stock price prediction for the top 5 companies within the S&P 500 index from different sectors. The dataset contains historical stock price data and sector information, collected from reputable sources. Here, I outline how the data was gathered, what it includes, why it was chosen, and any ethical considerations.

3.1 Data Source

The dataset used in this project consists of two primary sources:

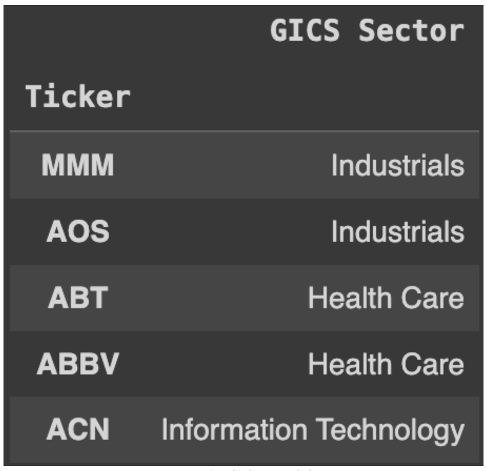
1. **Wikipedia:** The list of S&P 500 companies and their respective sectors was obtained from Wikipedia, specifically from the page listing S&P 500 companies. This source provided the symbols of the companies, which were used to identify the stocks for which historical data would be collected.

2. **Yahoo Finance:** Historical stock price data was collected from Yahoo Finance, covering daily adjusted closing prices, trading volumes, and other metrics. Yahoo Finance is a widely recognized platform for financial data, ensuring the reliability and accuracy of the information.

3.2 Data Collection

The purpose of data collection was to identify the top five stocks from each sector within the S&P 500 index, analyse the trends in stock prices, and develop predictive models. The contents include:

1. **Symbols and Sectors:** The following is a sample list of S&P 500 firms (Figure 1), followed by their distinct symbols and GICS (Global Industry Classification Standard) sectors. This data helps the classification of companies based on their industry and enables the choice of a wide variety of stocks.
   - Ticker(Symbols)
   - GICS Sectors



	GICS Sector
Ticker	
MMM	Industrials
AOS	Industrials
ABT	Health Care
ABBV	Health Care
ACN	Information Technology

Figure 1, S&P 500 List

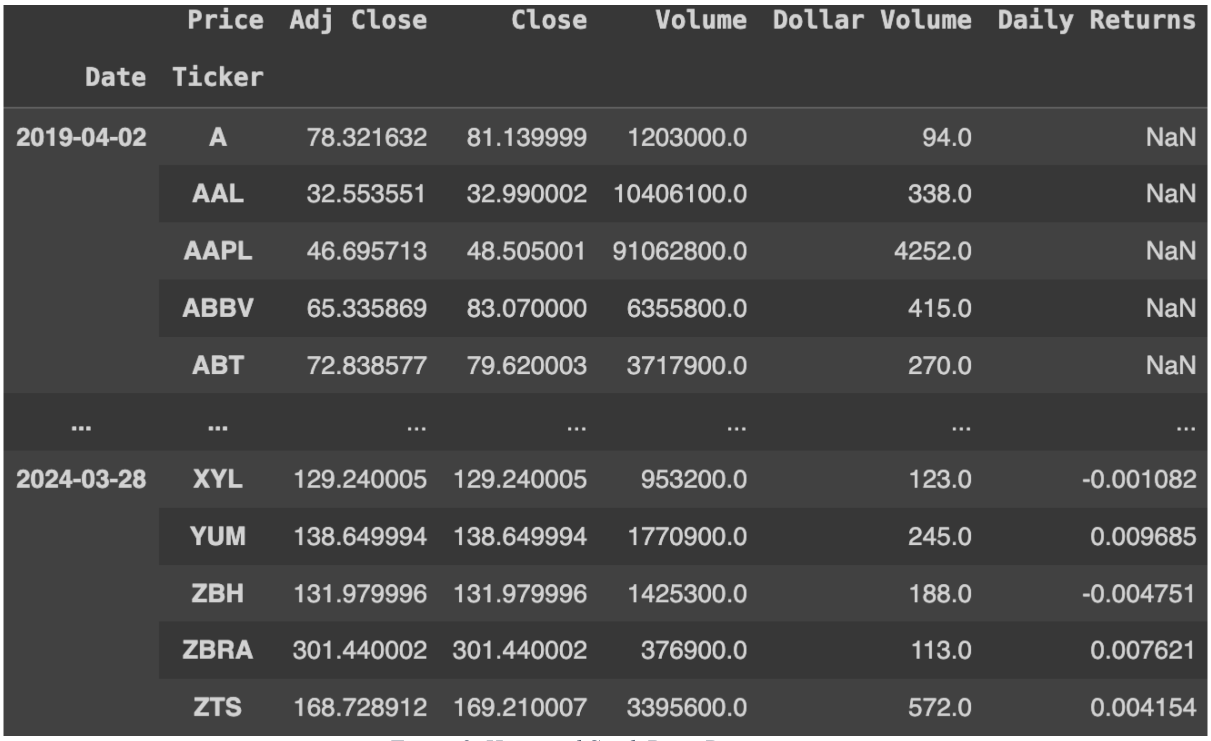
2. **Historical Stock Price Data**: The following is a list of daily data for all companies listed in Yahoo Finance (Figure 2), covering five years from early 2019 to early 2024. The data includes:

- **Adjusted Closing Price:**
  Based on dividends and stock splits, this feature shows the stock's closing price. As long as corporate acts that could change the price of a stock are taken into account, this important measure shows what the stock is worth. Adjusted closing price is often the main input for time-series predictions in deep learning models.

- **Volume:**
  Volume indicates the total number of shares traded for a particular stock on a given day. High volume can suggest increased interest or activity, while low volume may indicate reduced trading. This feature helps gauge market sentiment and liquidity.

- **Dollar Volume:**
  To determine dollar volume, the adjusted closing price is multiplied by the volume. It estimates the entire dollar value of transactions relating to a specified stock, thereby offering insight into the level of activity within the market. It can be utilised to determine the liquidity of a stock and identify periods of heavy trading.

- **Daily Returns:**
  Daily returns represent the percentage change in the adjusted closing price from one day to the next. This feature is useful for analyzing stock price volatility and detecting trends. In predictive modelling, daily returns can serve as a measure of a stock's performance over time.



	Price	Adj Close	Close	Volume	Dollar Volume	Daily Returns
Date	Ticker					
2019-04-02	A	78.321632	81.139999	1203000.0	94.0	NaN
	AAL	32.553551	32.990002	10406100.0	338.0	NaN
	AAPL	46.695713	48.505001	91062800.0	4252.0	NaN
	ABBV	65.335869	83.070000	6355800.0	415.0	NaN
	ABT	72.838577	79.620003	3717900.0	270.0	NaN
...	...	...	...	...	...	...
2024-03-28	XYL	129.240005	129.240005	953200.0	123.0	-0.001082
	YUM	138.649994	138.649994	1770900.0	245.0	0.009685
	ZBH	131.979996	131.979996	1425300.0	188.0	-0.004751
	ZBRA	301.440002	301.440002	376900.0	113.0	0.007621
	ZTS	168.728912	169.210007	3395600.0	572.0	0.004154

Figure 2, Historical Stock Price Data

3.3 Usage in the Project

These features are used in different ways throughout the project:

- **Model Input:**
  The adjusted closing price is the primary input for deep learning models. It forms the basis for time-series forecasting, allowing the models to learn patterns and trends in stock prices.

- **Feature Engineering:**
  Dollar volume is used to derive additional insights for analysis. For instance, it indicates overall market activity, aiding in ranking stocks and helping to identify top-performing stocks across different sectors.

- **Data Analysis:**
  Daily returns are employed to measure the correlation between selected stocks, guiding the selection process to choose stocks with the least correlation. This step helps to ensure diversity and reduce the impact of overlapping trends in the data.

- **Stock Selection and Diversity:**
  Tickers and GICS sector classifications are used to choose stocks from different industries, promoting diversity and minimizing correlations. This approach provides a broader perspective on the stock market, enhancing the robustness of the analysis.

3.4 Justification for the Dataset

The dataset was selected due to its ability to offer a comprehensive perspective of the S&P 500 index, encompassing a diverse array of industries. The five-year duration enables an in-depth study of stock price patterns and the creation of deep learning models for forecasting future prices. The diverse selection of stocks minimizes potential correlations and offers a broader perspective on market trends, supporting the research questions of the project.

3.5 Exploratory Data Analysis (EDA)

EDA is a very important step in getting to know the company and the features of the dataset that was used for this project. This part explains the EDA methods that were used on the stock price data to find patterns, check the accuracy of the data, and find insights that help with further research.

The first step in EDA was to inspect the raw data to identify any inconsistencies, missing values, or anomalies. This involved:

3.5.1 Loading and Cleaning the Data:

- I have read the S&P 500 company data from a Wikipedia page into a DataFrame (sp500), allowing you to extract the symbols and sector information.
- You replaced dots in the symbol column with hyphens (sp500['Symbol'].str.replace('.', '-')) to maintain consistency and avoid errors in data retrieval because financial databases use hyphens instead.

3.5.2 Checking for Missing Data:

- I examined the dataset for missing values and replaced them as needed. In cases where data was missing for adjusted closing prices, I used forward-fill to maintain continuity.

3.5.3 Removing Unnecessary Features:

- Columns such as 'High', 'Low', and 'Open' were removed, as they were not required for the analysis. This helped streamline the dataset and focus on key features.

3.5.4 Feature Engineering:

- I have calculated additional features, such as Dollar Volume and Daily Returns.

3.6 Ethical Considerations

Although the data used in this project is publicly available, ethical considerations are essential to maintain integrity and respect privacy. Key ethical issues include:

Data Privacy: The dataset contains only publicly available financial data. No personal or sensitive information is included, ensuring compliance with data privacy standards.

Data Attribution and Usage: The sources of the data, Wikipedia and Yahoo Finance, have been appropriately attributed. I ensured the use of data aligned with their terms of service and provided proper references.

Data Integrity: No data manipulation was done that could mislead or distort the results. The preprocessing and analysis steps were carried out transparently and accurately.

In summary, this dataset is suitable for analyzing stock price trends within the S&P 500 index. It provides a diverse range of companies, reliable historical data, and a comprehensive timeframe for building and testing deep learning models. Proper ethical considerations have been taken into account to ensure responsible use of the data.

4. Methodology

This section describes the practical work conducted for the project, with a specific focus on the use of deep learning models for predicting the stock prices of companies listed in the S&P 500 index. This comprehensive report includes the different stages involved in the process, including data pre-processing, feature engineering, the construction and training of deep learning models, and its subsequent evaluation. Here is a detailed account of the precise actions I undertook.

4.1 Data Pre-processing

Data pre-processing is a crucial step in preparing the dataset for deep learning models. It involves several tasks:

4.1.1 Data Cleaning:
I cleaned the data by removing any rows with missing or inconsistent values. Where missing values were found in the adjusted closing prices, I used forward-fill to maintain continuity. This step ensured that the data was accurate and complete for further analysis.

4.1.2 Feature Engineering:

- 'Dollar Volume' was calculated by multiplying the adjusted closing price by the trading volume, providing a measure of the total dollar value traded for each stock.
- 'Daily Returns' was derived by calculating the percentage change in the adjusted closing price from one day to the next. This feature helped capture stock price volatility and trends.

4.1.3 Normalization:
To ensure compatibility with deep learning models, I used MinMaxScaler to normalize the adjusted closing prices to a range between 0 and 1. This transformation is essential for training deep learning models effectively, as it keeps the data within a consistent range.

4.2 Stock Selection

To determine the optimal set of stocks for stock price prediction within the S&P 500, I began by ranking all companies based on their dollar volume a metric that reflects the total value of shares traded within a given period. By doing so, I could identify the top stock in each sector, providing a diverse selection across the S&P 500's various industries. The dollar volume ranking was derived from historical stock data and sector information.

The following list represents the top stocks selected from each sector, along with the associated Symbol:

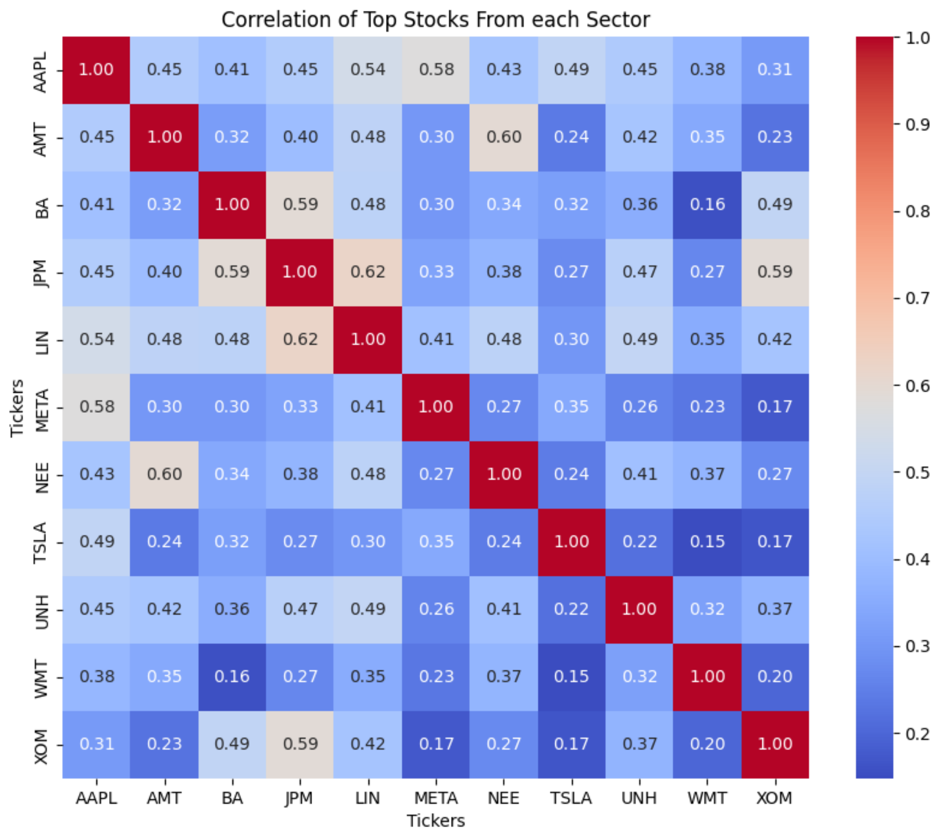
GICS Sector	Symbol	Company Name
Communication Services	META	Meta Platforms, Inc.
Consumer Discretionary	TSLA	Tesla, Inc.
Consumer Staples	WMT	Walmart Inc.
Energy	XOM	Exxon Mobil Corporation
Financials	JPM	JPMorgan Chase & Co.
Health Care	UNH	UnitedHealth Group Inc.
Industrials	BA	The Boeing Company

Information Technology	AAPL	Apple Inc.
Materials	LIN	Linde plc
Real Estate	AMT	American Tower Corp
Utilities	NEE	NextEra Energy, Inc.

Table 1.0

These top stocks, representing each of the 11 sectors in the S&P 500, were selected based on their high dollar volume. This method ensured a diverse range of companies from different industries, reducing potential correlation.

After selecting these 11 top stocks, I proceeded to find the five least related stocks among them. To do this, I created a heatmap that displayed the correlation between these top stocks based on their daily returns. The heatmap helps identify relationships and overlaps between the selected stocks. Here's the heatmap generated for this analysis:

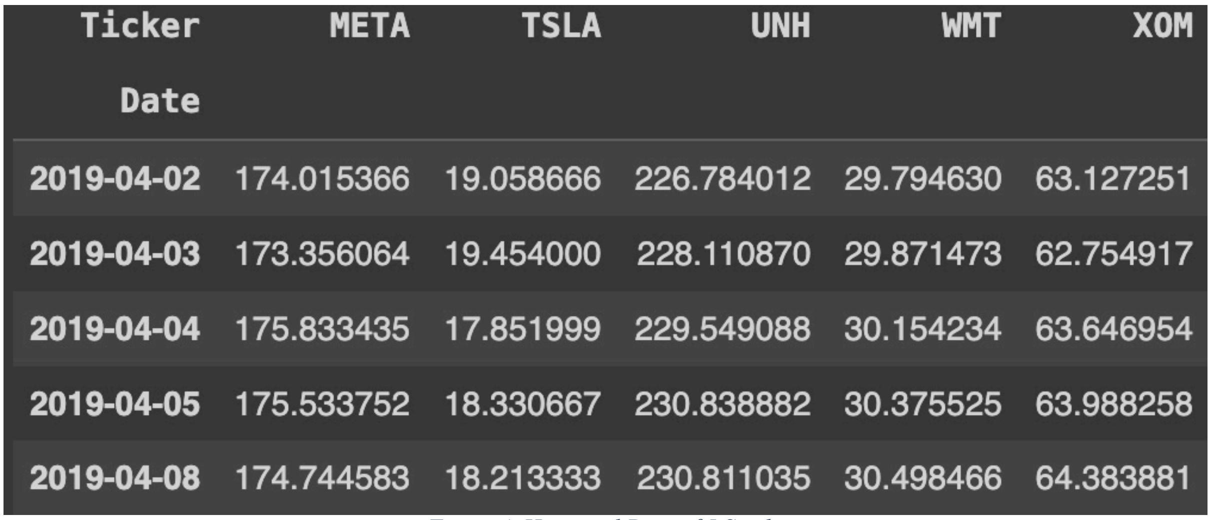


	AAPL	AMT	BA	JPM	LIN	META	NEE	TSLA	UNH	WMT	XOM
AAPL	1.00	0.45	0.41	0.45	0.54	0.58	0.43	0.49	0.45	0.38	0.31
AMT	0.45	1.00	0.32	0.40	0.48	0.30	0.60	0.24	0.42	0.35	0.23
BA	0.41	0.32	1.00	0.59	0.48	0.30	0.34	0.32	0.36	0.16	0.49
JPM	0.45	0.40	0.59	1.00	0.62	0.33	0.38	0.27	0.47	0.27	0.59
LIN	0.54	0.48	0.48	0.62	1.00	0.41	0.48	0.30	0.49	0.35	0.42
META	0.58	0.30	0.30	0.33	0.41	1.00	0.27	0.35	0.26	0.23	0.17
NEE	0.43	0.60	0.34	0.38	0.48	0.27	1.00	0.24	0.41	0.37	0.27
TSLA	0.49	0.24	0.32	0.27	0.30	0.35	0.24	1.00	0.22	0.15	0.17
UNH	0.45	0.42	0.36	0.47	0.49	0.26	0.41	0.22	1.00	0.32	0.37
WMT	0.38	0.35	0.16	0.27	0.35	0.23	0.37	0.15	0.32	1.00	0.20
XOM	0.31	0.23	0.49	0.59	0.42	0.17	0.27	0.17	0.37	0.20	1.00

Figure 3, Heatmap of selected stocks

Using this heatmap, I manually selected the five least related stocks from the set of 11 top performers. This manual selection process aimed to ensure a diverse and uncorrelated set of stocks for my analysis, contributing to a more reliable and robust stock price prediction model. By focusing on the least related stocks, I could avoid the risk of skewed results due to high correlations and better capture a broader range of market trends. This approach laid a solid foundation for building deep learning models and conducting further analysis to predict stock prices within the S&P 500 index.

Following this selection process, I plotted (Figure 5) the adjusted closing prices over time for the five least related stocks. This plot provides a visual representation of stock price trends for the selected companies, allowing me to assess their historical performance and identify patterns. Here's the table listing (Figure 4) the adjusted closing prices for these five stocks over a specific period:



Ticker	META	TSLA	UNH	WMT	XOM
Date					
2019-04-02	174.015366	19.058666	226.784012	29.794630	63.127251
2019-04-03	173.356064	19.454000	228.110870	29.871473	62.754917
2019-04-04	175.833435	17.851999	229.549088	30.154234	63.646954
2019-04-05	175.533752	18.330667	230.838882	30.375525	63.988258
2019-04-08	174.744583	18.213333	230.811035	30.498466	64.383881

Figure 4, Historical Data of 5 Stocks

The following plot visualizes the adjusted closing prices for these five stocks over time, showing their trends and fluctuations across the specified period:

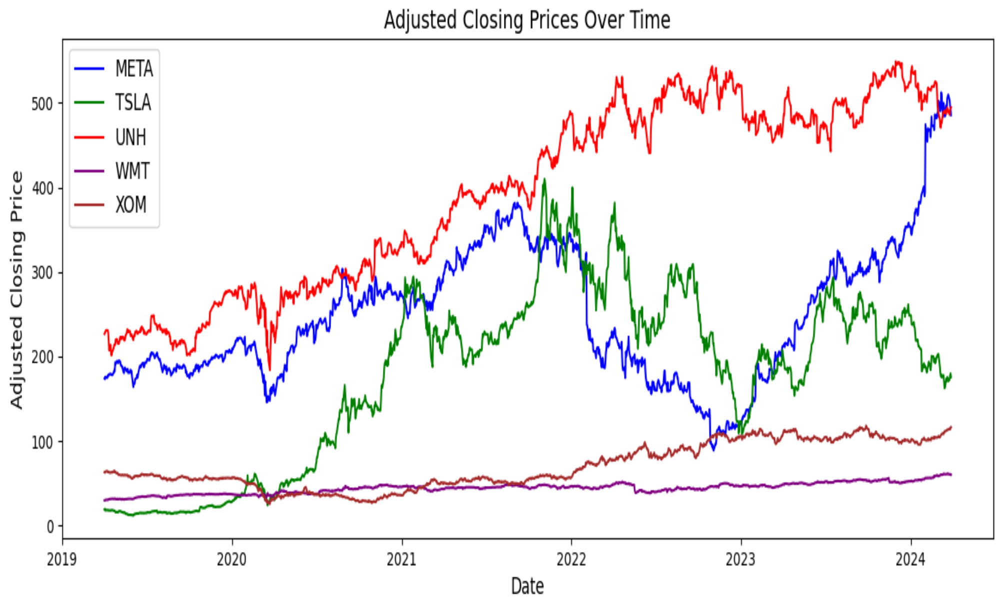


Figure 5, Adjusted Closing Prices Over Time

The plot of adjusted closing prices over time for the five least related stocks provides valuable insights into their price trends from early 2019 through early 2024. This visualization helps analyse the distinct price patterns and fluctuations among these selected stocks, allowing for an in-depth examination of how they have performed in the S&P 500 index.

In the plot, the x-axis represents the timeline, spanning from 2019 to 2024, while the y-axis represents the adjusted closing prices of the stocks. The lines on the plot show the trajectory of each stock over time, with different colours indicating different stocks for clarity.

Distinct Price Trends:
Each stock demonstrates a unique price pattern, reflecting varying degrees of growth, stability, or volatility. For example, UNH (red) shows a consistent upward trend, indicating steady growth, while TSLA (green) has a more erratic pattern, with significant peaks and troughs.

Long-Term Growth:
Stocks like META (blue) exhibit a generally upward trajectory over the observed period, suggesting substantial long-term growth. This trend may indicate strong market performance and investor confidence in these companies.

Stability and Volatility:
While some stocks, like WMT (purple) and XOM (brown), maintain relatively stable trends, others, such as TSLA, display high volatility, with sharp increases and decreases in price. This variation in behaviour is crucial for understanding the risks and opportunities in stock price prediction.

Comparative Analysis:
The plot allows for a comparative analysis between stocks, illustrating how their price trends align or differ. It helps determine the extent to which these stocks move together or separately, contributing to the broader understanding of correlations and diversity.

After examining the price trends and fluctuations among the five least related stocks, I refined my dataset to include only data till 2024-02-10. This filtering was done to focus my analysis on a specific timeframe and maintain consistency throughout my report.

I extracted this subset of data by applying a condition to include only those records where the 'Date' column was on or before 2024-02-10. The reason for filtering data up to this point was to use it as a benchmark for predicting future stock prices. By hiding the rest of the data after this date, I could test the accuracy of my deep learning models in predicting the stock prices, using the hidden data as a reference for verification.

The validation process using the hidden data helped determine if my deep learning models could accurately predict future stock prices within the S&P 500 index. This methodology provided a clear indication of the models' reliability, allowing me to fine-tune my approach and gain insights into the effectiveness of deep learning techniques in stock price prediction.

4.3 Building Deep Learning Models

After preparing the data, three deep-learning models were constructed: LSTM, Bi-LSTM, and GRU. This is the process via which I built each model:

4.3.1 LSTM Model

The LSTM model was constructed using the Sequential API from Keras, an architecture that facilitates the layer-by-layer construction of neural networks. The LSTM model contained the following components:

- **LSTM Layers:**
  Each of the three LSTM layers of the model is composed of 100 units. The length of the sequences generated by the Timeseries Generator is matched by the input shape parameter of the initial LSTM layer. The return sequences parameter is set to True for the first two LSTM layers, indicating that the output from these layers should retain the sequence structure. This design allows the information to flow through the layers, enabling the model to capture long-term dependencies in the time-series data.

- **Dropout Layers:**
  I added dropout layers with a 20% dropout rate after each LSTM layer to stop overfitting. Dropout randomly sets some of the input units to zero during training. This helps to make the model more consistent and makes it better at generalising.

- **Dense Output Layer:**
  The final output layer is a dense layer with one unit. This layer produces a single prediction for each sequence, which, in this case, is the predicted stock price. The linear activation function is used, as this is a regression problem where I am predicting continuous values.

After constructing the LSTM model, I compiled it with the Adam optimizer, a popular choice in deep learning due to its adaptive learning rate and efficient handling of gradients. The loss function used is mean squared error (MSE), appropriate for regression tasks like stock price prediction. MSE calculates the average of the squares of the errors, providing a measure of how much the predicted values deviate from the actual values.

4.3.2 Bi-LSTM Model

The architecture of the Bi-LSTM model has several components designed to leverage bidirectional data processing:

- **Bidirectional LSTM Layers:**
  In the model, there are three Bidirectional LSTM layers, and each one has 100 units. In bidirectional LSTMs, there are two LSTMs. One processes the series from beginning to end, and the other processes it from end to beginning. By using this method, the model can successfully capture patterns and relationships that go both ways, which makes it easier to understand the time-series data.

- **Dropout Layers:**
  There are dropout layers with a 20% dropout rate added after each Bidirectional LSTM layer to stop overfitting. Dropout helps to make the model more consistent by setting a random number of input units to zero during training. This makes it better at generalisation.

- **Dense Output Layer:**
  The final output layer is a dense layer with one unit, producing a single prediction for each sequence. This output represents the predicted stock price, as this is a regression problem where continuous values are predicted.

After building the Bi-LSTM model, I compiled it with the Adam optimizer, known for its efficiency and adaptive learning rate, which works well with deep learning tasks. The loss function used is MSE, a common choice for regression tasks as it measures the squared differences between predicted and actual values.

4.3.3 GRU Model

The GRU model is a variant of the recurrent neural network framework, resembling the LSTM model but featuring a more straightforward structure. The simplicity of GRU enhances its computational and training efficiency while preserving its capacity to interpret patterns in time-series data.

- **GRU Layers:**
  The model consists of three GRU layers, each with 100 units. The first two GRU layers have the return sequences parameter set to True, ensuring that the sequence structure is maintained throughout the network. This configuration allows the model to process sequential data and capture long-term dependencies. The third GRU layer has return sequences set to False, indicating that it outputs a single value, which is then passed to the dense output layer.

- **Dropout Layers:**
  To reduce overfitting, I incorporated dropout layers with a 20% dropout rate following each GRU layer. Dropout is a technique that randomly disables a subset of the units during training. This process promotes the model's ability to generalise more effectively when faced with new data.

- **Dense Output Layer:**
  The final output layer is a dense layer with one unit, representing the predicted stock price for each sequence. This output is the result of the model's interpretation of the time-series data.

Once the GRU model was built, I compiled it with the Adam optimizer, which is commonly used in deep learning due to its adaptive learning rate and efficient gradient descent. The loss function used was mean squared error (MSE), which is appropriate for regression tasks like stock price prediction.

5. Results

This section presents the results obtained from my stock price prediction project, focusing on key metrics and visualizations that evaluate the performance of the deep learning models. I used a combination of metrics to ensure a comprehensive assessment, providing insights into the models' accuracy, training time, and overall effectiveness.

5.1 META Stock Prediction with LSTM model

Training Time:
The time taken to train the LSTM model for META was 86.084 seconds. This measure shows how well the model uses computing power, which suggests that the training process went relatively quickly.

RMSE:
The RMSE for the LSTM model on META was 7.755. This relatively low value implies that the model's predictions are close to the actual stock prices, indicating high accuracy in the model's predictions.

MAE:
The MAE for the LSTM model on META was 4.647, further emphasizing the accuracy of the predictions. This metric measures the average absolute error, providing a straightforward indication of the model's reliability.

Visualization:
Visualizations are very important for judging how well the design works. Using the LSTM model, the following plot (Figure 6) shows the predicted stock prices for META. The green line shows the predicted stock prices and the red line shows the real stock prices:

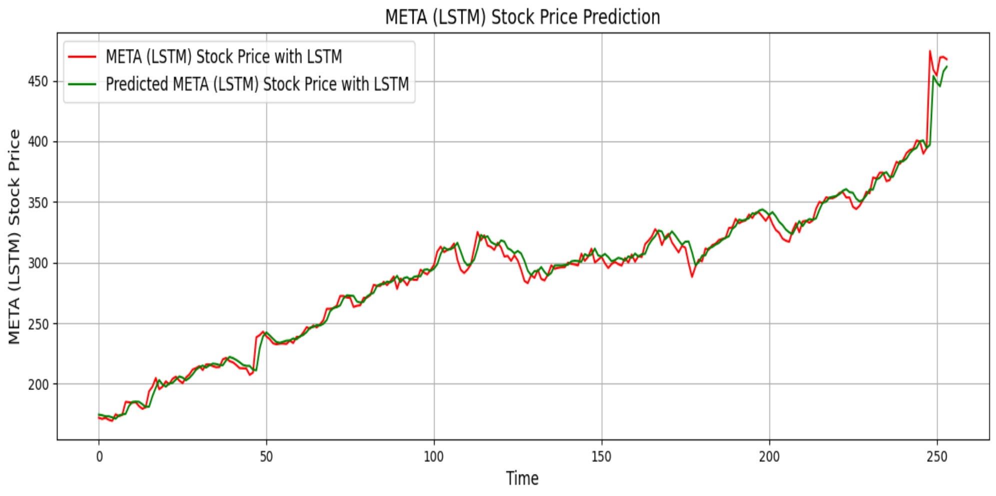


Figure 6, LSTM META Stocks Prediction

This plot demonstrates that the LSTM model closely follows the actual stock prices, suggesting high predictive accuracy. The proximity between the predicted and actual lines indicates that the model has successfully captured the underlying patterns in the stock price data.

5.2 TSLA Stock Prediction with LSTM model

Training Time:
The time taken to train the LSTM model for TSLA was approximately 89.239 seconds. This duration indicates the computational efficiency of the model, suggesting that the training process was relatively quick.

RMSE:
The RMSE for the LSTM model on TSLA was 7.996. This relatively low value implies that the model's predictions were reasonably close to the actual stock prices, indicating a good level of accuracy.

MAE:
The MAE for the LSTM model on TSLA was 6.045. This metric represents the average absolute difference between predicted and actual stock prices, suggesting that the model had a relatively low prediction error.

Visualization:
I used a plot (Figure 7) to show the difference between the predicted and real stock prices to see how well the model worked. Using the LSTM model, the following plot shows the predicted stock prices for TSLA. The green line shows the predicted prices and the red line shows the real prices.

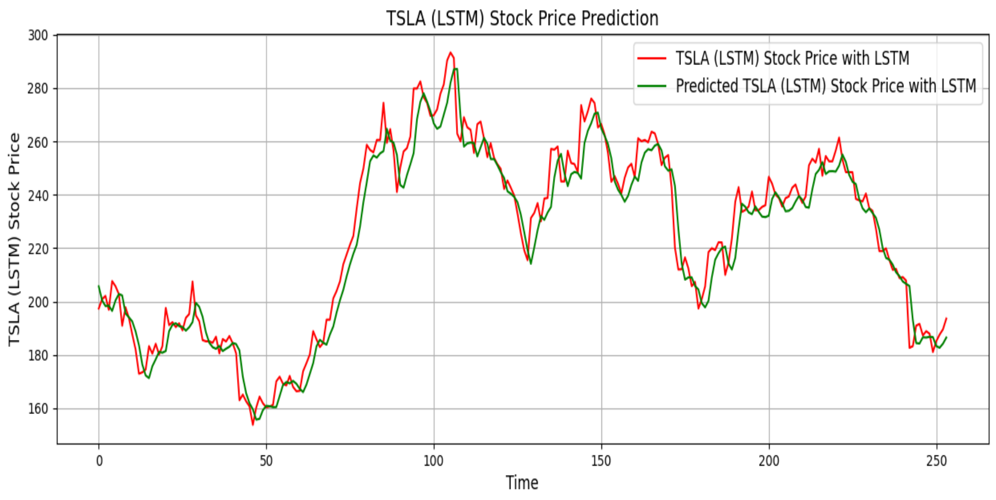


Figure 7, LSTM TSLA Stocks Prediction

The red and green lines are very close to each other, which shows that the LSTM model's predictions were very close to the real stock prices. This shows that the model is good at predicting stock price trends.

5.3 UNH Stock Prediction with LSTM model

Training Time:
This means that the LSTM model for UNH only took 78.192 seconds to train, which means that it makes good use of both time and computer resources.

RMSE:
The RMSE for the LSTM model on UNH was 7.617, demonstrating a relatively low prediction error. This value indicates that the model's predictions are reasonably close to the actual stock prices, suggesting high accuracy.

MAE:
The MAE for the LSTM model on UNH was 5.867, indicating that the model's average absolute prediction error was low, further reinforcing its accuracy.

Visualization:
To visualize the LSTM model's performance for UNH stock prediction, I used a plot (Figure 8) that compared the predicted and actual stock prices. In the following plot, the green line represents the predicted stock prices, while the red line represents the actual stock prices:

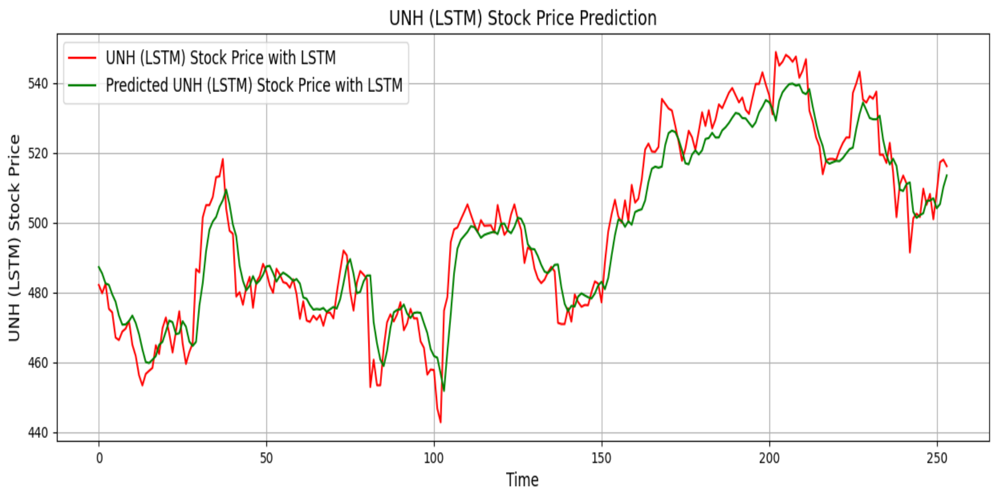


Figure 8, LSTM UNH Stocks Prediction

The close alignment between the red and green lines suggests that the LSTM model's predictions closely followed the actual stock prices, demonstrating the model's capability to predict stock price trends accurately.

5.4 WMT Stock Prediction with LSTM model

Training Time:
The training time for the LSTM model on WMT was approximately 72.029 seconds. This short duration suggests that the model is efficient and can be trained quickly, allowing for rapid predictions.

RMSE:
The RMSE for the LSTM model on WMT was 0.731, indicating a low level of prediction error. RMSE is a measure of the average squared differences between predicted and actual stock prices, and a lower value indicates higher accuracy.

MAE:
The MAE for the LSTM model on WMT was 0.574. This low value indicates that the average absolute prediction error was minimal, reinforcing the model's accuracy.

Visualization:
I made a plot (Figure 9) that shows the difference between predicted and real stock prices to see how well the LSTM model did at predicting WMT stock. The LSTM model was used to predict the stock prices for WMT. The green line shows the predicted prices and the red line shows the real prices.

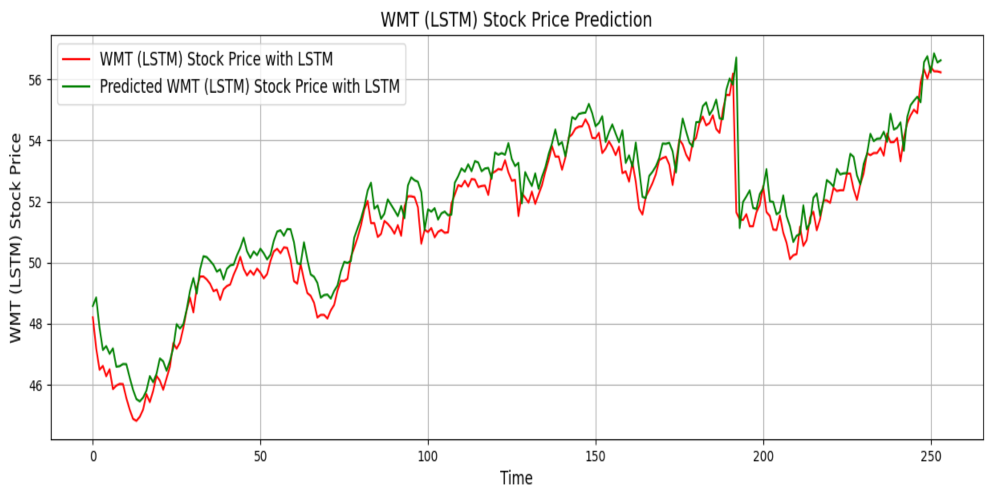


Figure 9, LSTM WMT Stocks Prediction

Since the red and green lines are very close to each other, it looks like the LSTM model's predictions were pretty accurate about the real stock prices. This shows that the model can correctly predict how stock prices will move.

5.5 XOM Stock Prediction with LSTM model

Training Time:
The LSTM model for XOM had a training time of 71.361 seconds. This quick training time indicates the model's computational efficiency and suggests it can be trained rapidly to make predictions.

RMSE:
The RMSE for the LSTM model on XOM was 1.863. This relatively low value indicates that the model's predictions were close to the actual stock prices, demonstrating high accuracy.

MAE:
The MAE for the LSTM model on XOM was 1.442, indicating that the average absolute error was low, reinforcing the model's accuracy and reliability.

Visualization:
I generated graphs(Figure 10) to visually represent the performance of the LSTM model by comparing the anticipated and real stock values for XOM. This graphic illustrates the projected stock prices for XOM utilising the LSTM model. The green line denotes the expected values, while the red line represents the actual prices.

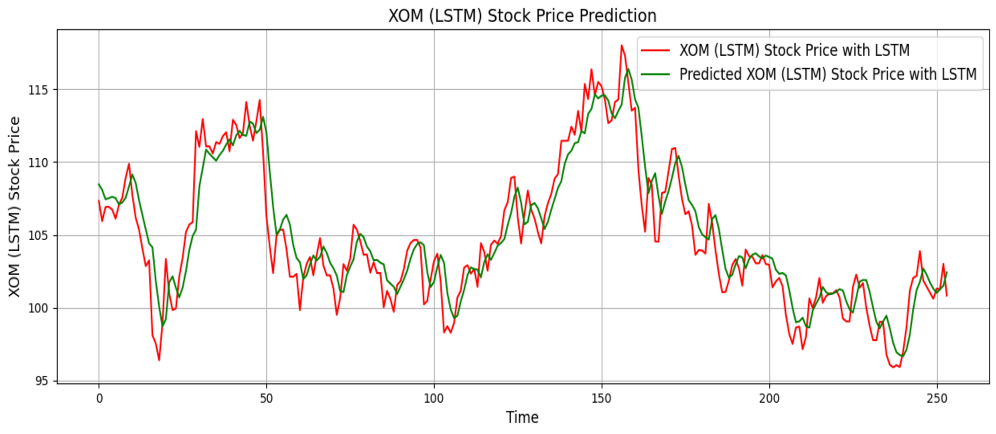


Figure 10, LSTM XOM Stocks Prediction

The strong correspondence observed between the red and green lines implies that the predictions generated by the LSTM model closely corresponded to the observed stock prices, hence indicating the model's proficiency in accurately forecasting stock price movements.

5.6 META Stock Prediction with Bi-LSTM model

Training Time:
The Bi-LSTM model took 141.432 seconds to train, indicating the time required to build and fine-tune this more complex model. This longer training time compared to LSTM models suggests increased computational demand.

RMSE:
The RMSE for the Bi-LSTM model on META was 9.171. This metric measures the average squared differences between predicted and actual stock prices, with lower values indicating higher accuracy. The RMSE here shows a moderate level of prediction error, suggesting room for improvement.

MAE:
The MAE for the Bi-LSTM model on META was 6.486. This value represents the average absolute error, indicating the typical deviation between predicted and actual stock prices. The MAE suggests a moderate level of accuracy in predictions.

Visualization:
To assess the efficacy of the Bi-LSTM model, I used a plot (Figure 11) to compare the projected and observed stock prices for META. The plot displays the anticipated stock prices as the green line and the actual stock prices as the red line.

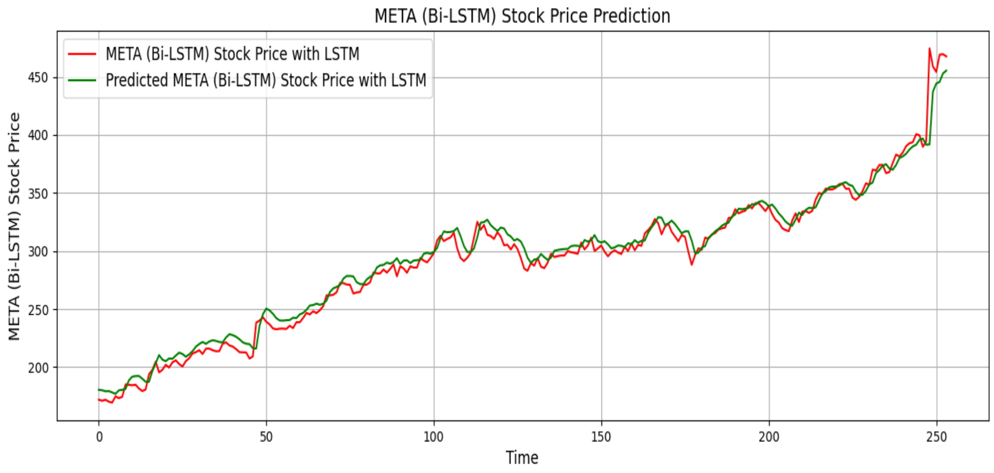


Figure 11, Bi-LSTM META Stocks Prediction

The strong correlation between the red and green lines suggests that the predictions made by the Bi-LSTM model were typically precise but with occasional variations. The differences indicate that the model may have accurately represented the overall pattern but had difficulties when dealing with smaller-scale fluctuations.

5.7 TSLA Stock Prediction with Bi-LSTM model

Training Time:
The Bi-LSTM model took 132.323 seconds to train. This longer training time reflects the added complexity of the bidirectional architecture, which can capture information from both past and future sequences.

RMSE:
The RMSE for the Bi-LSTM model on TSLA was 9.617. This metric, measuring the average squared differences between predicted and actual stock prices, indicates a moderate level of prediction error, suggesting that the model has some room for improvement in accuracy.

MAE:
The MAE for the Bi-LSTM model on TSLA was 7.735. This value, representing the average absolute error between predicted and actual stock prices, demonstrates a moderate deviation, indicating potential areas for enhancement in model accuracy.

Visualization:
To visualize the Bi-LSTM model's performance, I created a plot (Figure 12) to compare the predicted and actual stock prices for TSLA. The following plot shows the predicted stock prices for TSLA using the Bi-LSTM model, with the green line representing predicted prices and the red line showing actual prices:

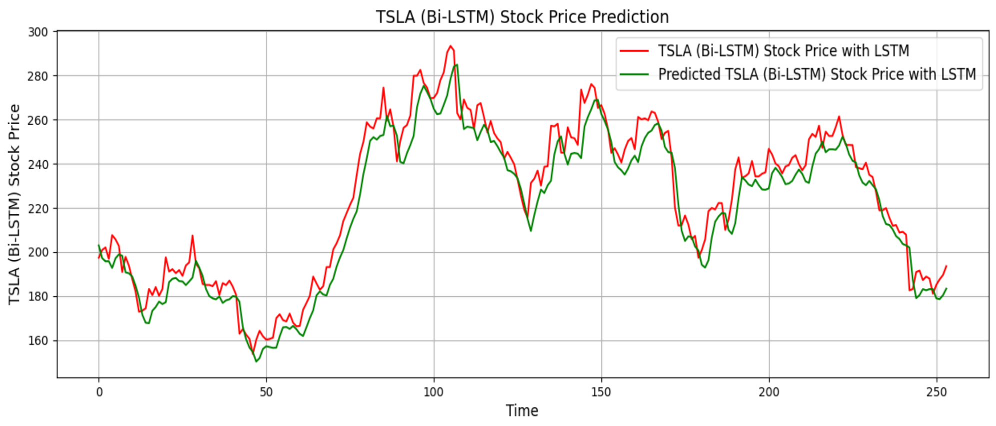


Figure 12, Bi-LSTM TSLA Stocks Prediction

The close alignment between the red and green lines suggests that the Bi-LSTM model's predictions generally align with the actual stock prices, indicating that the model has captured broader trends, though it may struggle with fine-scale variations.

5.8 UNH Stock Prediction with Bi-LSTM model

Training Time :
The training time for the Bi-LSTM model was approximately 134.959 seconds. This moderate duration reflects the computational complexity of the bidirectional architecture, which requires more resources and time compared to simpler models.

RMSE :
Bi-LSTM model on UNH had an RMSE of 10.302. The high prediction error level in this metric suggests that the model may not have been able to correctly predict how the stock's price would move.

MAE :
The MAE for the Bi-LSTM model on UNH was 8.284. This value represents the average absolute error, indicating that the model's predictions had significant deviations from the actual stock prices, suggesting a need for improvement in accuracy.

Visualization:
To see how well the Bi-LSTM model predicted UNH stock prices, I made a plot (Figure 13) that showed the difference between the predicted and real stock prices. Using the Bi-LSTM model, the following plot shows the predicted stock prices for UNH. The green line shows the predicted prices and the red line shows the real prices:

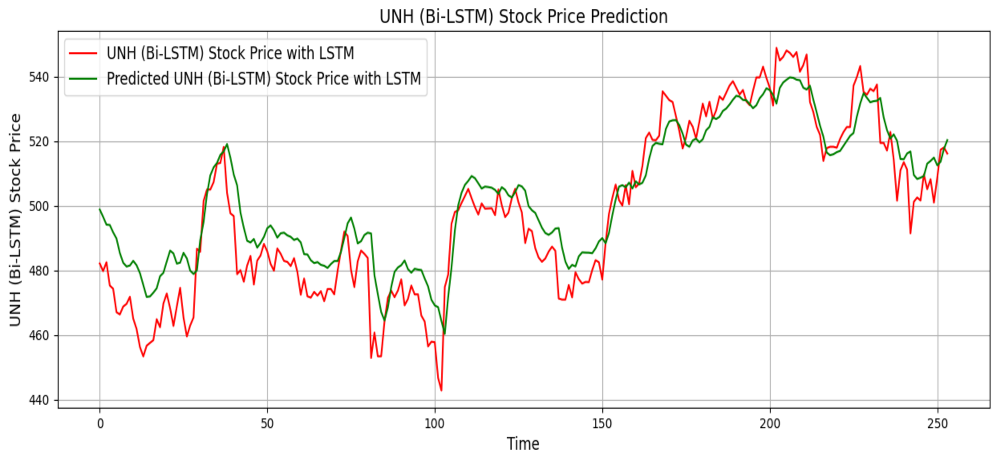


Figure 13, Bi-LSTM UNH Stocks Prediction

The gaps between the red and green lines suggest that the Bi-LSTM model struggled to closely align its predictions with the actual stock prices. This variance indicates that the model might have difficulty capturing certain patterns or fluctuations in the stock's price trends.

5.9 WMT Stock Prediction with Bi-LSTM model

Training Time:
The Bi-LSTM model for WMT required about 130.389 seconds for training. This slightly extended time reflects the additional computational load due to the bidirectional structure, processing sequences from both forward and backward perspectives.

RMSE:
The RMSE for the Bi-LSTM model on WMT was 0.536, suggesting that the model's predictions were generally close to the actual stock prices, indicating moderate accuracy.

MAE:
The MAE for the Bi-LSTM model on WMT was 0.399, a relatively low deviation that signifies the model's capacity to predict stock price movements with reasonable precision.

Visualization:
I created a plot (Figure 14) to illustrate the Bi-LSTM model's prediction accuracy for WMT stock prices. The plot below shows the predicted stock prices for WMT using the Bi-LSTM model, with the green line indicating predicted prices and the red line showing the actual prices:

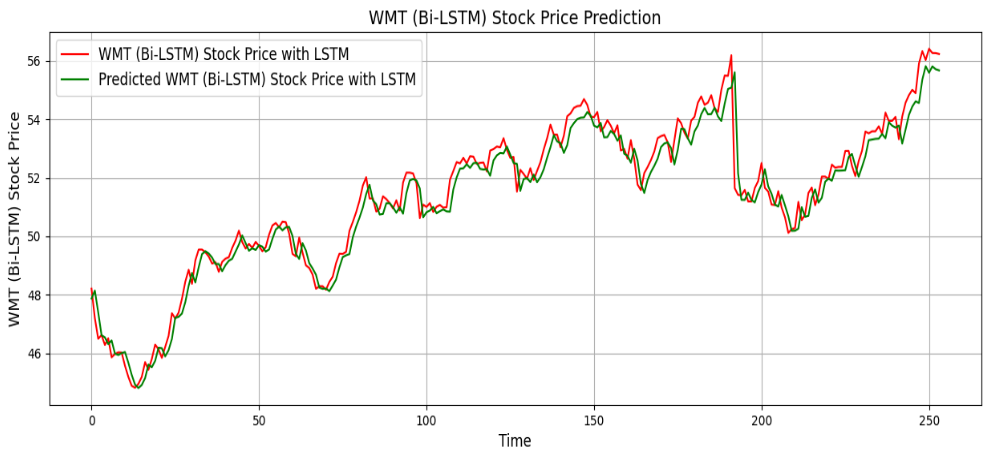


Figure 14, Bi-LSTM WMT Stocks Prediction

The close alignment between the red and green lines suggests that the Bi-LSTM model performed well in predicting WMT stock prices, showcasing the ability to capture trends and patterns accurately.

5.10 XOM Stock Prediction with Bi-LSTM model

Training Time:
About 138.445 seconds were needed to train the Bi-LSTM model for XOM. With bidirectional processing, which looks at sequence patterns in both forward and backward ways at the same time, this extra time is needed to account for the extra computational complexity.

RMSE:
The RMSE for the Bi-LSTM model on XOM was 1.829. This relatively low RMSE indicates that the model's predictions are closely aligned with the actual stock prices, suggesting effective pattern recognition and forecasting capabilities.

MAE:
The MAE for the Bi-LSTM model on XOM was 1.447. This small error between predicted and actual stock prices demonstrates that the Bi-LSTM model achieved a high degree of accuracy, with only slight deviations from the true stock price.

Visualization:
To see how well the Bi-LSTM model predicted XOM stock prices, I made plots that showed the difference between the predicted and real stock prices. Using the Bi-LSTM model, the following plot (Figure 15) shows the predicted stock prices for XOM. The green line shows the predicted prices and the red line shows the real prices:

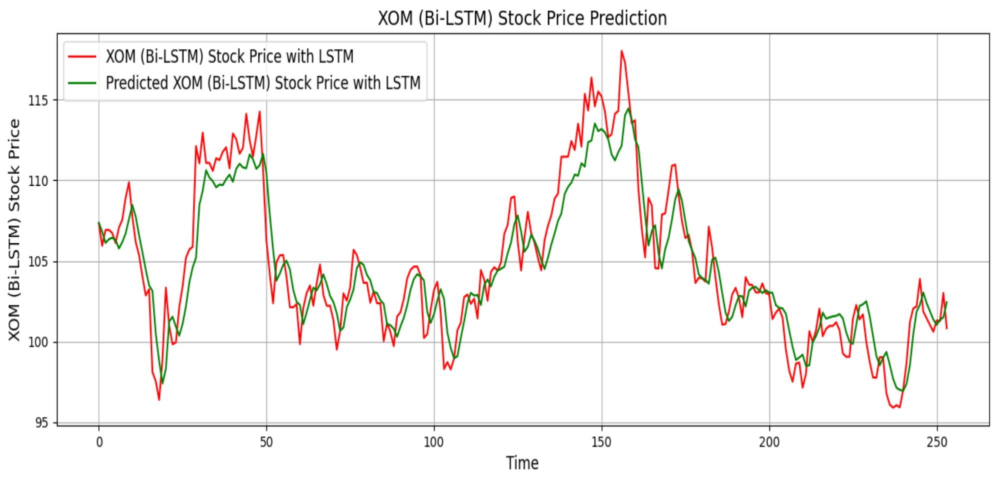


Figure 15, Bi-LSTM XOM Stocks Prediction

The red and green lines agreement shows that the Bi-LSTM model correctly predicted the price of XOM stock, which suggests that it is good at predicting stock price trends with some divergence at certain points.

5.11 META Stock Prediction with GRU Model

Training Time:
The GRU model for META took approximately 66.801 seconds to train, indicating a relatively short training time compared to more complex models like Bi-LSTM. This shorter training time suggests that the GRU architecture is computationally efficient while retaining high performance.

RMSE:
The RMSE for the GRU model on META was 9.359. This value indicates a moderate level of prediction accuracy, with a slight discrepancy between the predicted and actual stock prices. However, the GRU model was still able to capture significant trends and patterns in stock price movements.

MAE:
The MAE for the GRU model on META was 5.620. This metric shows that the GRU model achieved reasonable accuracy, with an average deviation of 5.620 between predicted and actual stock prices.

Visualization:
To better understand the GRU model's performance, I created a (Figure 16) plot to visualize the predicted and actual stock prices for META. The following plot compares predicted stock prices (in green) and actual stock prices (in red), indicating how closely the GRU model's predictions align with the real-world data:

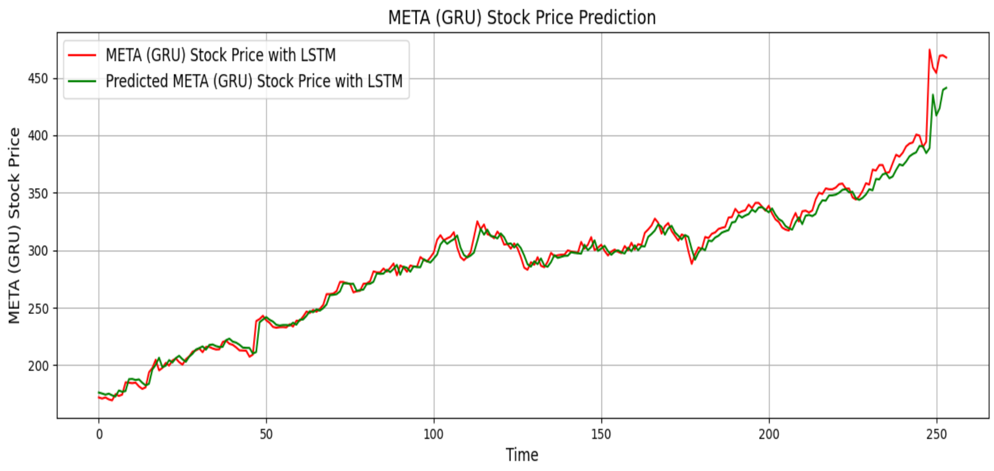


Figure 16, GRU META Stocks Prediction

The close alignment between the red and green lines suggests that the GRU model successfully predicted META stock prices. This outcome indicates that the model captured key patterns, demonstrating its capability to forecast stock price trends.

5.12 TSLA Stock Prediction with GRU Model

Training Time:
The GRU model for TSLA required approximately 68.185 seconds for training. This relatively short training time indicates that the model's architecture is less computationally intensive compared to other deep learning structures, allowing for efficient training without compromising accuracy.

RMSE:
The GRU model's RMSE on TSLA was 6.888. The model's ability to accurately predict stock prices is demonstrated by the low RMSE value, which also implies a high alignment between the model's predictions and real stock prices.

MAE:
The MAE for the GRU model on TSLA was 5.068. This relatively low MAE indicates a close match between predicted and actual stock prices, highlighting the model's accuracy and reliability.

Visualization:
The following figure (Figure 17) shows the projected and observed stock prices for TSLA to show how accurate the GRU model is in predicting stock prices. The red line shows the actual prices and the green line shows the expected values.

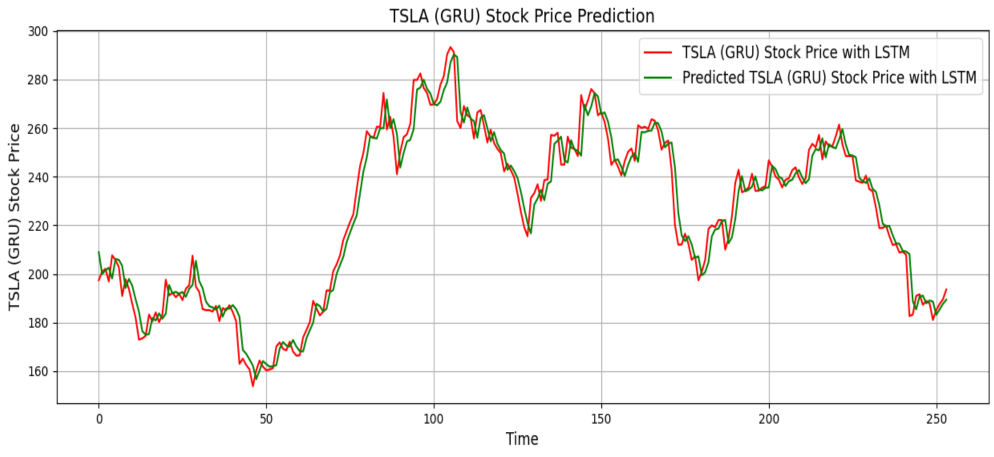


Figure 17, GRU TSLA Stocks Prediction

The robust correlation observed between the red and green lines serves as evidence of the GRU model's efficacy in predicting TSLA stock prices, hence validating its capacity to accurately model and predict stock price changes.

5.13 UNH Stock Prediction with GRU Model

Training Time:
The GRU model for UNH took approximately 67.610 seconds to train. This relatively short duration suggests efficient training, potentially due to the streamlined architecture of GRU compared to LSTM-based models.

RMSE:
The RMSE for the GRU model on UNH was 8.705, indicating the average magnitude of prediction error. This relatively moderate RMSE value suggests a reasonable level of prediction accuracy but also highlights some variance between predicted and actual stock prices.

MAE:
The MAE for the GRU model on UNH was 6.957. This value represents the average absolute difference between predicted and actual stock prices, indicating a moderate level of accuracy.

Visualization:
To visualize the GRU model's performance for UNH stock prediction, a plot (Figure 18) was created to compare predicted and actual stock prices. The following plot shows the predicted stock prices for UNH using the GRU model, with the green line representing predicted prices and the red line showing actual prices:

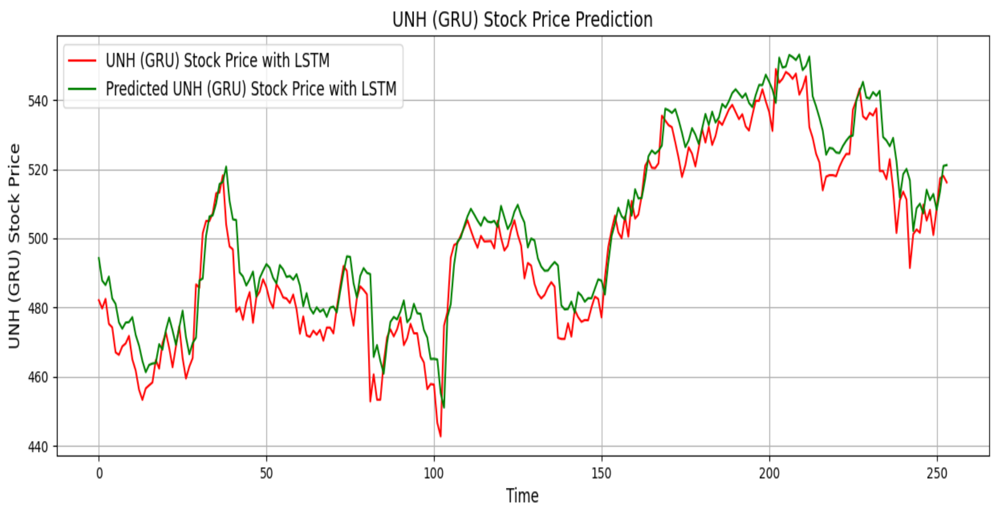


Figure 18, GRU UNH Stocks Prediction

The alignment between the red and green lines indicates that the GRU model effectively captured the overall trends in UNH stock price movements, with some divergence at certain points.

5.14 WMT Stock Prediction with GRU Model

Training Time:
The GRU model for WMT took approximately 70.345 seconds to train. This training duration reflects the efficiency of the GRU architecture, which is less computationally intense compared to more complex structures like Bi-LSTM.

RMSE:
The RMSE for the GRU model on WMT was 0.856. This relatively low RMSE suggests that the model's predictions are closely aligned with actual stock prices, indicating effective pattern recognition and accurate forecasting.

MAE:
The MAE for the GRU model on WMT was 0.731. This small difference between predicted and actual stock prices implies that the GRU model achieved a high level of accuracy in predicting stock price movements.

Visualization:
To visualize the performance of the GRU model for WMT stock prediction, I created a plot (Figure 19) to compare predicted and actual stock prices. The following plot shows the predicted stock prices for WMT using the GRU model, with the green line representing the predicted prices and the red line showing actual prices:

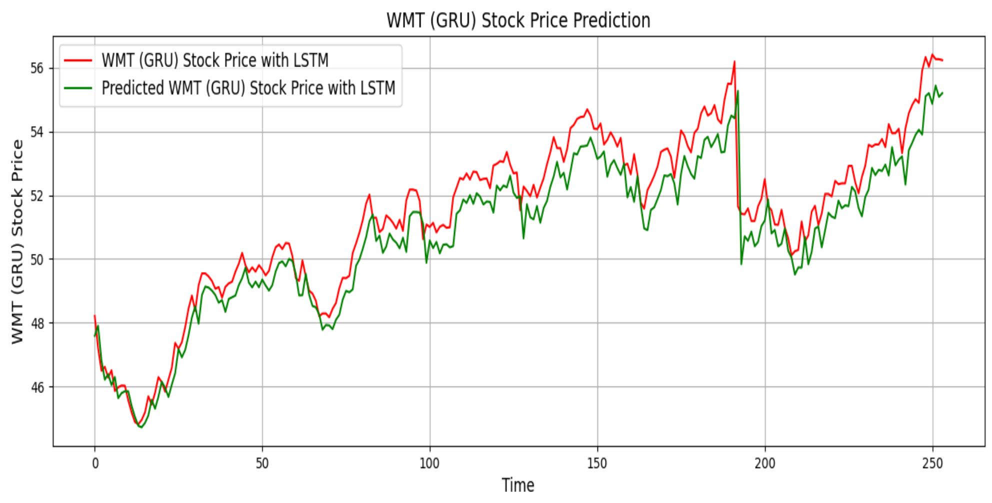


Figure 19, GRU WMT Stocks Prediction

The divergence between the red and green lines indicates that the GRU model did not predict WMT stock prices with high accuracy, suggesting some limitations in its ability to accurately track stock price trends.

5.15 XOM Stock Prediction with GRU Model

Training Time:
The GRU model for XOM took approximately 68.235 seconds to train. This relatively short training time demonstrates the computational efficiency of the GRU architecture, which tends to be faster due to its simpler structure compared to more complex models.

RMSE:
The RMSE for the GRU model on XOM was 3.633. This value suggests a moderate level of error between predicted and actual stock prices, indicating that while the model can capture stock price trends, there is still room for improvement in reducing prediction errors.

MAE:
The MAE for the GRU model on XOM was 3.216. This MAE value suggests a moderate level of accuracy, indicating that the GRU model's predictions were generally close to the actual stock prices, with some deviations.

Visualization:
To visualize the GRU model's performance for XOM stock prediction, I created plots to compare predicted and actual stock prices. The following plot shows the predicted stock prices for XOM using the GRU model, with the red line representing predicted prices and the green line showing actual prices:

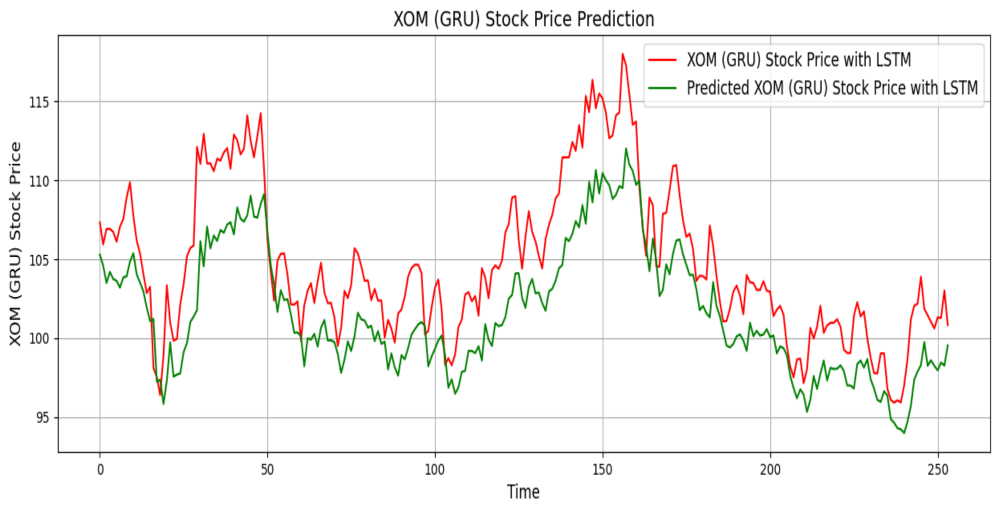


Figure 20, GRU XOM Stocks Prediction

The misalignment between the red and green lines indicates that the GRU model did not accurately predict XOM stock prices, demonstrating some challenges in tracking stock price trends effectively.

6. Analysis and discussion

In this section, I examine the performance of LSTM, Bi-LSTM, and GRU models in predicting the stock prices of META, TSLA, UNH, WMT, and XOM, comparing their ability to accurately forecast future market movements based on historical data.

6.1 META Stock Prediction

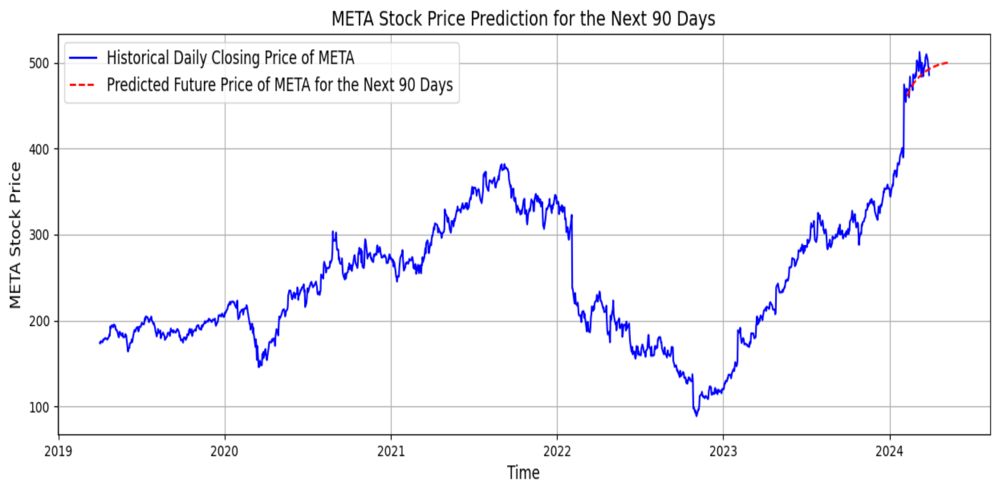


Figure 21, LSTM META Future Prediction

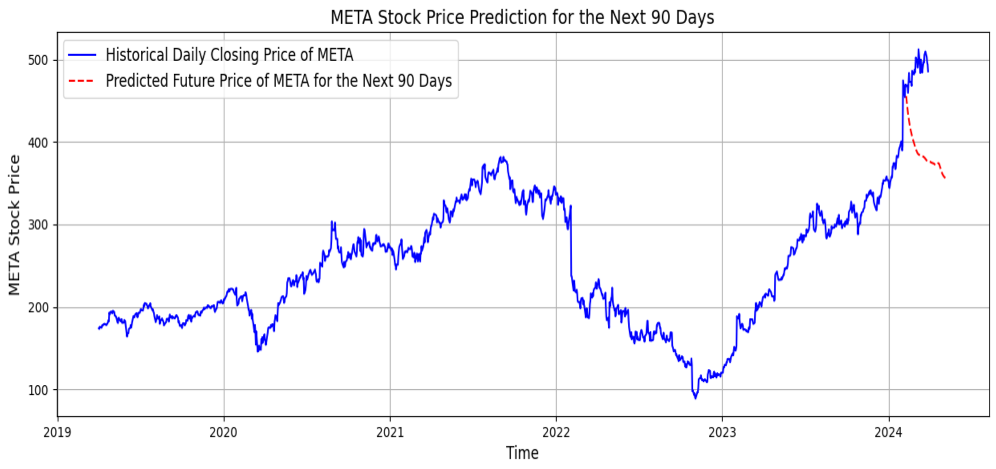


Figure 22, Bi-LSTM META Future Prediction

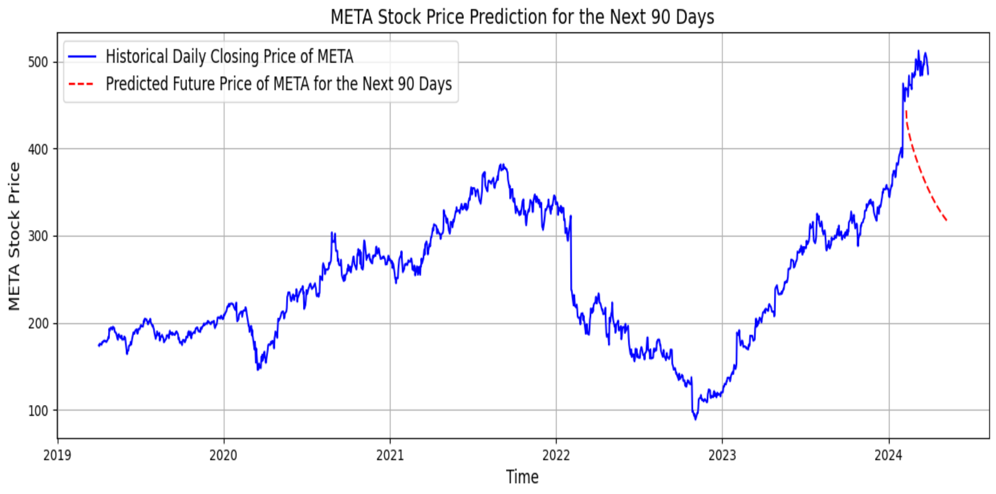


Figure 23, GRU META Future Prediction

In a comparison of predictive models on META stock prices LSTM, Bi-LSTM, and GRU, the LSTM model stands out for its precise alignment with historical data. The LSTM's predictions closely follow the actual stock prices, illustrating its ability to capture key market dynamics. Meanwhile, the Bi-LSTM and GRU models forecast varying degrees of decline, with the GRU indicating a sharp downturn. These differences underline the unique predictive capabilities of each model, with the LSTM offering the most consistent and reliable forecasts based on historical trends.

6.2 TSLA Stock Prediction

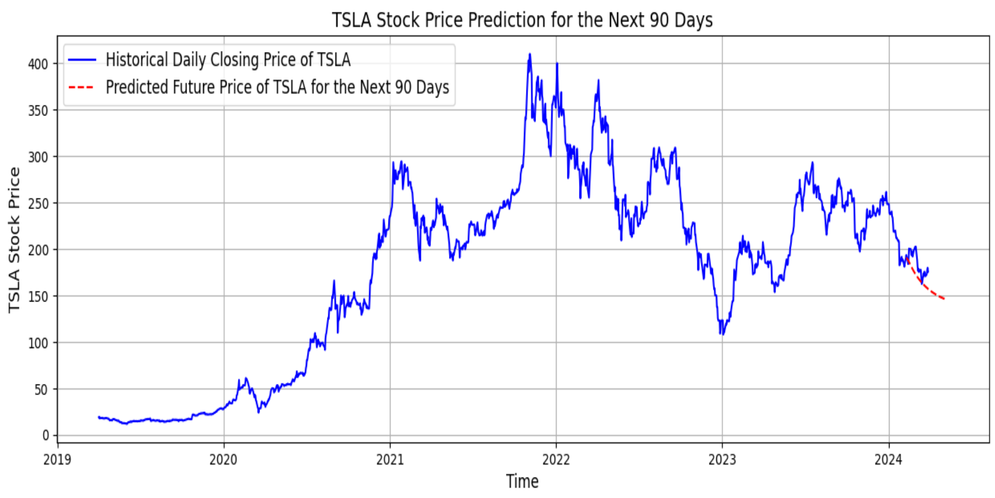


Figure 24, LSTM TSLA Future Prediction

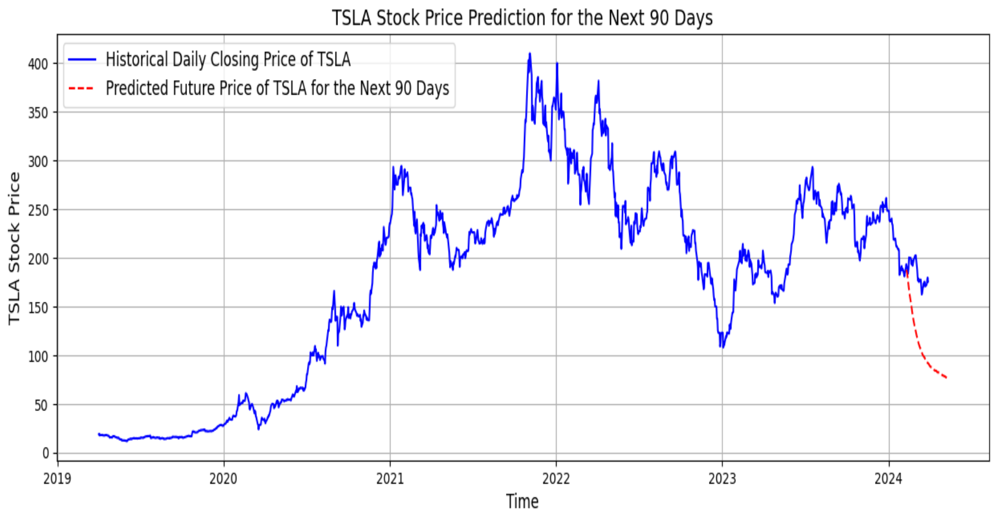


Figure 25, Bi-LSTM TSLA Future Prediction

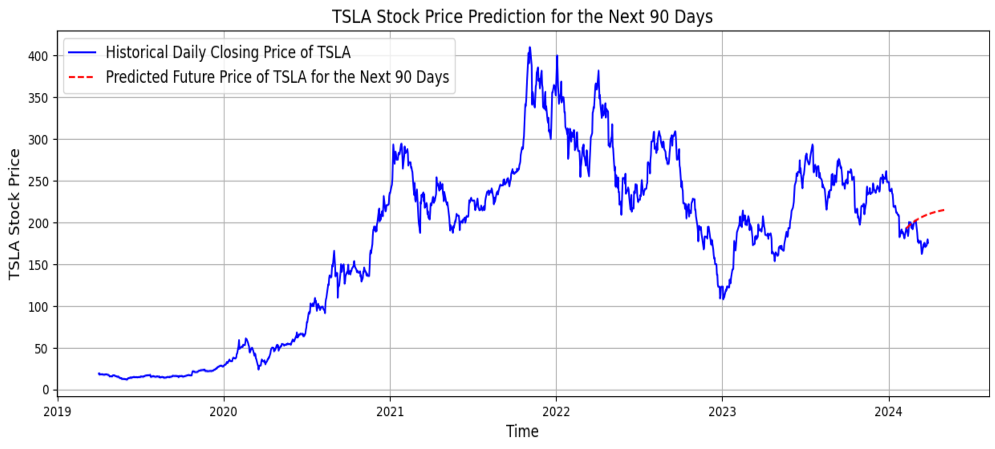


Figure 26, GRU TSLA Future Prediction

The LSTM model, which closely tracked the actual stock prices (the blue line), outperformed the other predictive models (Bi-LSTM and GRU) in an analysis comparing their performance on TSLA stock prices. While the Bi-LSTM and GRU models forecasted a steeper decline and an incline, the LSTM model consistently mirrored past trends, proving its effectiveness in capturing the dynamics of Tesla's stock movements. This precision makes the LSTM model particularly useful for investors seeking reliable market forecasts.

6.3 UNH Stock Prediction

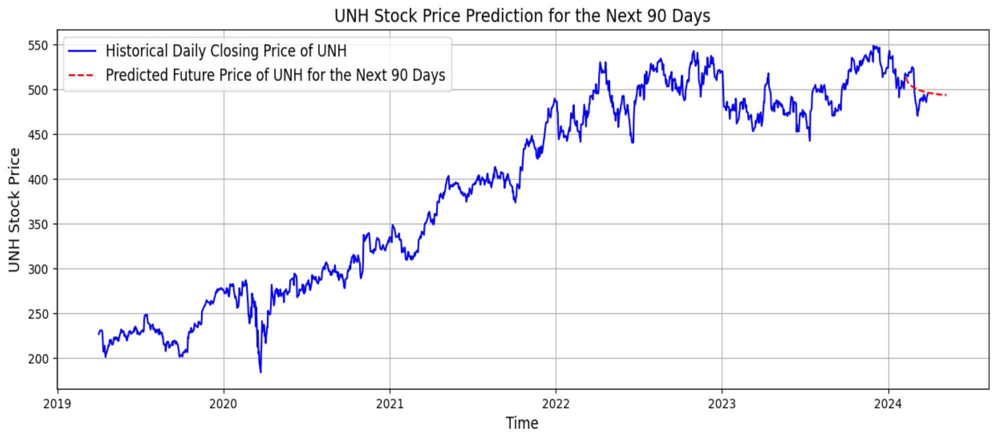


Figure 27, LSTM UNH Future Prediction

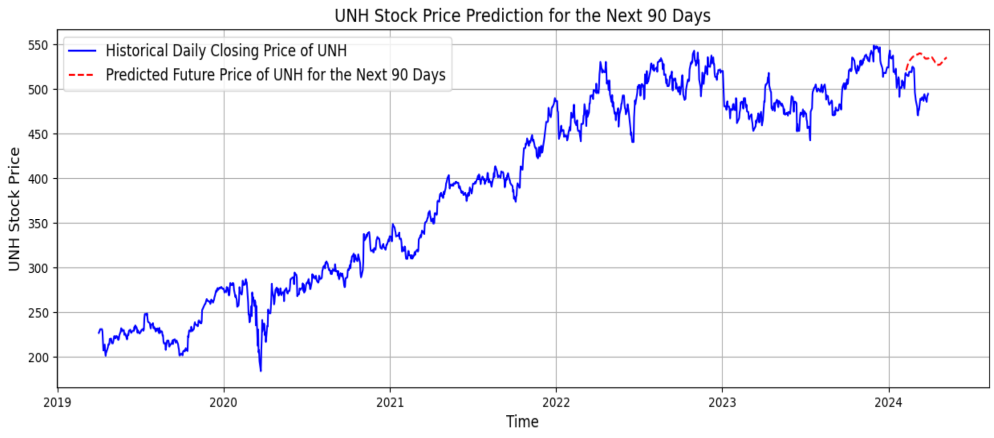


Figure 28, Bi-LSTM UNH Future Prediction

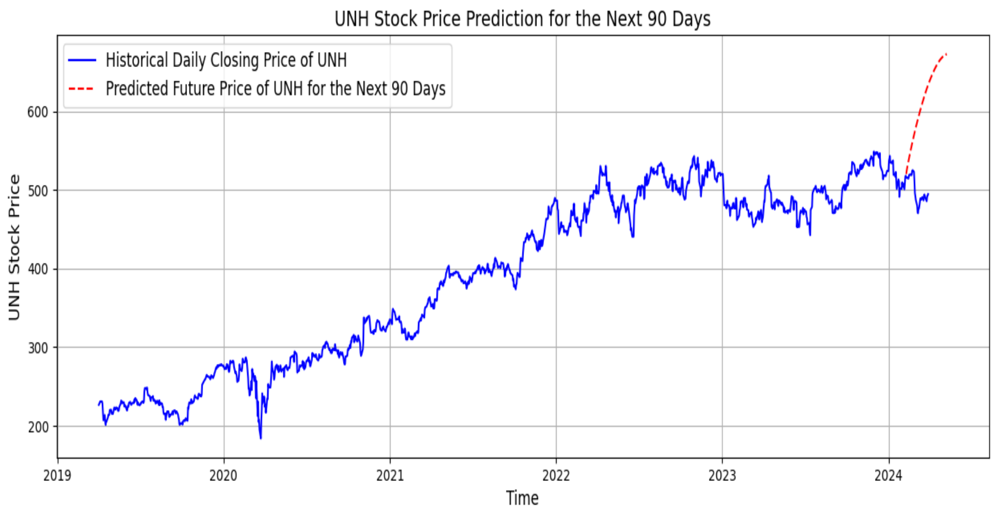


Figure 29, GRU UNH Future Prediction

Among the LSTM, Bi-LSTM, and GRU models applied to UNH stock predictions, the LSTM model most accurately matches the historical data, as evident from its close tracking of the blue line on the graph. While the Bi-LSTM suggests a modest upward trend and the GRU predicts a sharper rise, the LSTM model provides the most consistent and reliable forecast, making it a preferred tool for those needing precise market predictions.

6.4 WMT Stock Prediction

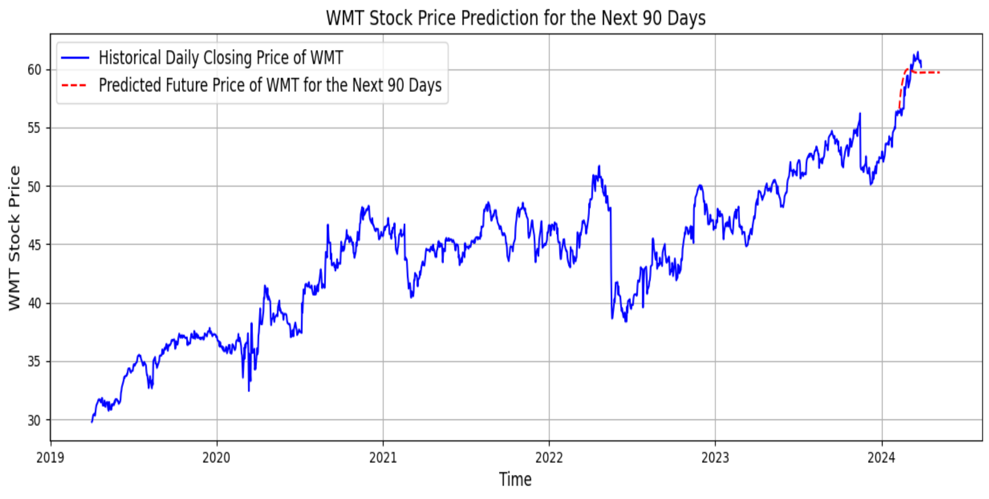


Figure 30, LSTM WMT Future Prediction

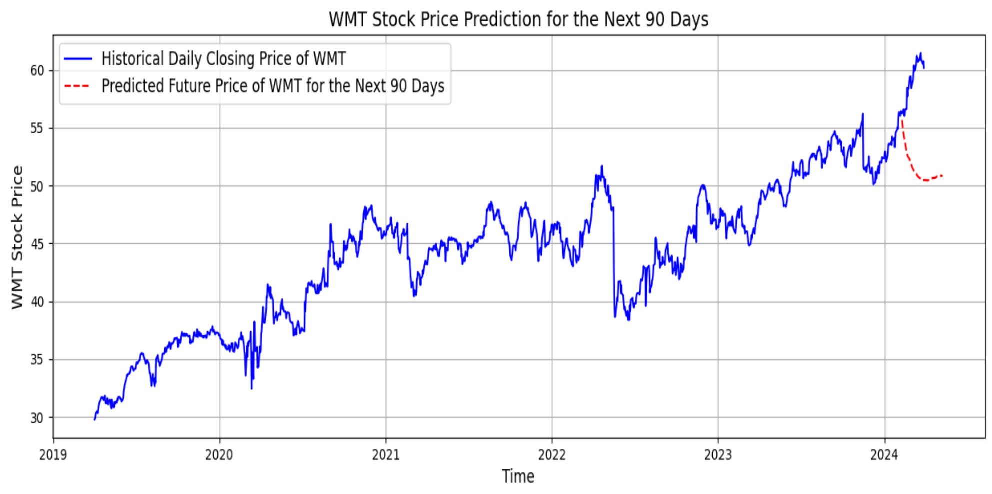


Figure 31, Bi-LSTM WMT Future Prediction

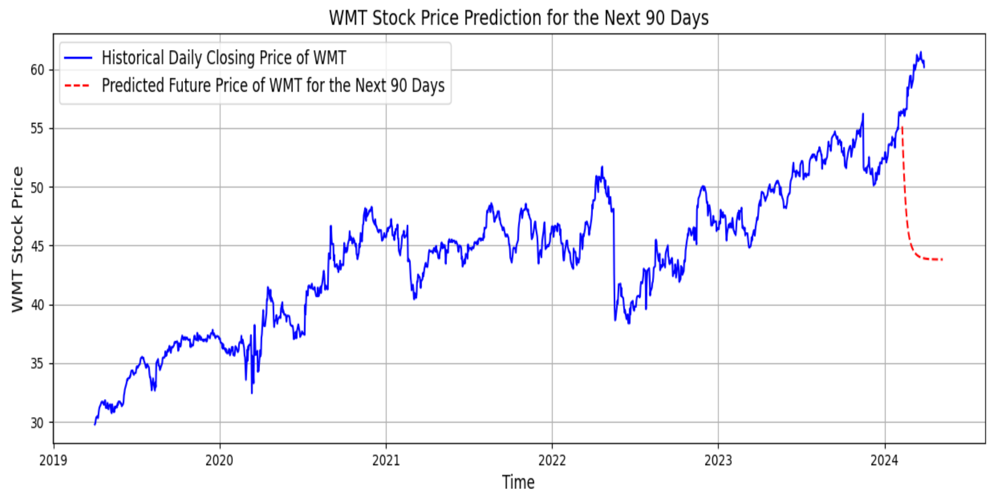


Figure 32, GRU WMT Future Prediction

Among the LSTM, Bi-LSTM, and GRU models analyzing WMT stock, the LSTM model aligns most closely with the historical data, accurately reflecting Walmart's stock trends. While the Bi-LSTM and GRU predict a downturn following recent gains, with the GRU suggesting a sharper fall, the LSTM model stands out for its precise and reliable predictions, making it a preferred choice for investors seeking accurate market forecasts.

6.5 XOM Stock Prediction

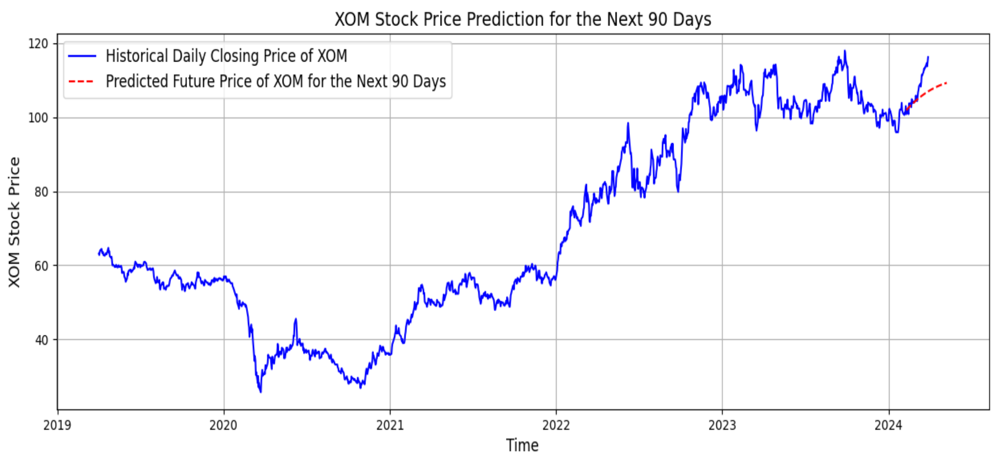


Figure 33, LSTM XOM Future Prediction

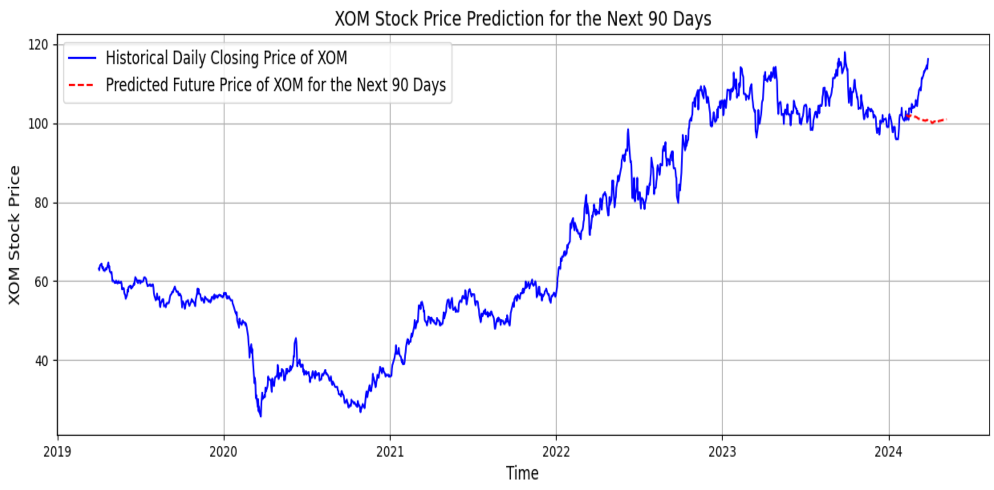


Figure 34, Bi-LSTM XOM Future Prediction

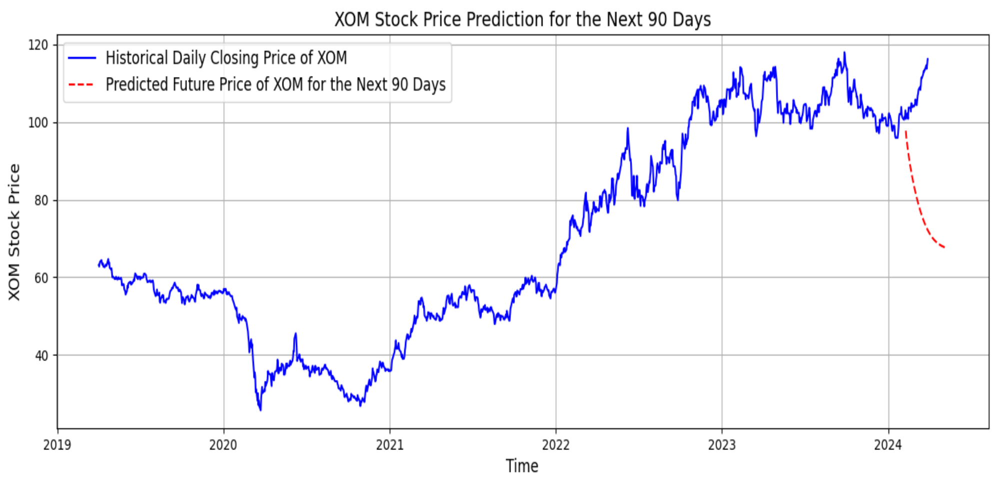


Figure 35, GRU XOM Future Prediction

Among the LSTM, Bi-LSTM, and GRU models used to predict XOM stock prices, the LSTM model most accurately reflects historical trends, closely following the actual stock price movements shown by the blue line. While the Bi-LSTM predicts a brief increase before stabilizing, and the GRU anticipates a sharp decline, the LSTM provides the most precise and reliable forecast, aligning closely with Exxon Mobil's historical data. This accuracy makes the LSTM model especially useful for investors needing dependable market predictions.

6.6 Summary

The comparative analysis of LSTM, Bi-LSTM, and GRU models for predicting stock prices across the top 5 S&P 500 companies (META, TSLA, UNH, WMT, XOM) underscores the robustness of the LSTM model in closely mirroring historical stock trends, offering the most consistent and reliable forecasts. This suggests its superiority in capturing key market dynamics, which is in line with existing literature that often highlights the efficacy of LSTM models in time-series predictions due to their ability to remember long-term dependencies. However, limitations arise from the models' varying degrees of accuracy in predicting sudden market changes, which points to the need for incorporating additional data or improving model robustness against market volatility.

These findings align with the study's goals to improve investment decision-making processes by providing robust predictive tools for financial professionals. By incorporating quantitative assessments like RMSE and MAE, and visual comparisons of predicted against actual stock prices, this research underscores the potential of accurate stock price predictions to aid investors in risk management and strategic investment planning. This utility extends to different financial applications, where accurate predictions can significantly influence investment strategies and financial forecasting.

7. Conclusion

This study demonstrated the efficacy of LSTM, Bi-LSTM, and GRU models in forecasting the stock prices of the top 5 S&P 500 companies in different sectors. The LSTM model consistently delivered the most precise predictions, closely resembling past patterns. The results indicate that the LSTM's ability to retain long-term connections makes it especially helpful for predicting financial outcomes. Precise forecasts generated by these models can greatly assist in risk reduction, portfolio optimisation, and strategic decision-making, providing valuable information for investors and financial analysts trying to navigate the complex terrain of the stock market.

7.1 Drawbacks and Future Work

While the models used in this study showed promising results, they primarily focused on price data without incorporating technical indicators, which could potentially enhance predictive accuracy. Future research should explore the integration of technical indicators like moving averages, RSI (Relative Strength Index) and MACD (Moving Average Convergence Divergence), which might provide deeper insights into market sentiment and price momentum. Additionally, experimenting with hybrid models that combine features of LSTM, Bi-LSTM, and GRU could offer improvements in capturing market dynamics and predicting abrupt price changes, addressing some of the limitations noted in the current model performance. This approach could lead to more robust predictions and help develop advanced tools for financial market analysis and decision-making.

8. References

Chaajer, P., Shah, M. and Kshirsagar, A. (2021). The applications of artificial neural networks, support vector machines, and long-short term memory for stock market prediction. Decision Analytics Journal, 2, p.100015. doi:https://doi.org/10.1016/j.dajour.2021.100015.

Gao, Y., Wang, R. and Zhou, E. (2021). Stock Prediction Based on Optimized LSTM and GRU Models. Scientific Programming, 2021, pp.1–8. doi:https://doi.org/10.1155/2021/4055281.

Istiake Sunny, Md.A., Shahriar Maswood, M.M. and G. Alharbi, A. (2020). *Deep Learning-Based Stock Price* Prediction Using LSTM and Bi-Directional LSTM Model | IEEE Conference Publication | IEEE Xplore. [online] ieeexplore.ieee.org. Available at: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9257950.

Ji, H. (2022). Stock price prediction based on SVM, LSTM, ARIMA. BCP Business & Management, 35, pp.267–272. doi:https://doi.org/10.54691/bcpbm.v35i.3302.

Venikar, I., Joshi, J., Jalnekar, H. and Raut, S. (2022). Stock Market Prediction Using LSTM. International Journal for Research in Applied Science and Engineering Technology, 10(12), pp.920–924. doi:https://doi.org/10.22214/ijraset.2022.47967.

Wikipedia Contributors (2019). List of S&P 500 companies. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/List_of_S%26P_500_companies.

Yahoo Finance (2023). Yahoo Finance – stock market live, quotes, business & finance news. [online] uk. finance. yahoo. com. Available at: https: //uk. finance. yahoo. com.

9. Appendices

9.1 Code

```
# Importing necessary libraries and functions
import pandas as pd
import yfinance as yf
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
import time
from tensorflow.keras.layers import Embedding, LSTM, GRU, Bidirectional, Dense, Dropout,
Input
from keras.layers import GRU
from keras.models import Sequential
from keras.optimizers import Adam
from tensorflow import keras
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from tensorflow.keras.preprocessing.sequence import TimeseriesGenerator
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from sklearn.model_selection import train_test_split

# Read S&P 500 companies data from Wikipedia
sp500 = pd.read_html('https://en.wikipedia.org/wiki/List_of_S%26P_500_companies')[0]

# Replace any dots in the symbol column because financial databases use hyphens instead
sp500['Symbol'] = sp500['Symbol'].str.replace('.', '-')

# Get a list of unique symbols
symbols_list = sp500['Symbol'].unique().tolist()

# Extract symbols and sectors
symbols_and_sectors = sp500[['Symbol', 'GICS Sector']]

# Define start and end dates for data retrieval
end_date = '2024-03-31'
start_date = pd.to_datetime(end_date) - pd.DateOffset(365 * 5)

# Download historical stock data for S&P 500 companies
df = yf.download(tickers=symbols_list, start=start_date, end=end_date).stack()

# Calculate dollar volume and Daily Returns for each stock
df['Dollar Volume'] = (df['Adj Close'] * df['Volume'])//1e6
df['Daily Returns'] = df.groupby('Ticker')['Adj Close'].pct_change()
# Drop 'High', 'Low', 'Open' columns
df.drop(columns=['High', 'Low', 'Open'], inplace=True)
```

```
df

# Change the column name from 'Symbol' to 'Ticker'
symbols_and_sectors.rename(columns={'Symbol': 'Ticker'}, inplace=True)

# Set the 'Ticker' column as the index column
symbols_and_sectors.set_index('Ticker', inplace=True)

# Display the symbols_and_sectors DataFrame
symbols_and_sectors.head()

# Calculating Monthly Dollar Volume mean of each Stock
dlr = df.unstack('Ticker')['Dollar Volume'].resample('M').mean()
dlr = dlr.stack().to_frame('Dollar Volume')

# Merging dlr and symbols_and_sectors
merged_dlr = dlr.merge(symbols_and_sectors, left_index=True, right_index=True)

# Calculates the 5-year rolling average of the 'Dollar Volume'
merged_dlr['Dollar Volume'] = merged_dlr['Dollar
Volume'].unstack().rolling(5*12).mean().stack()

# Creating a Rank Column to rank accodring to the Dollar Volume
merged_dlr['Rank'] = (merged_dlr.groupby('Date')['Dollar Volume'].rank(ascending=False))

# Dropping NaN values
merged_dlr.dropna()

# Finding the top Stocks from each Sector
top_tickers = merged_dlr.groupby('GICS Sector')['Rank'].idxmin()

# Display the top_tickers
top_tickers

# selecting the top tickers from the df dataframe and reset index
df_corr = df.loc[df.index.get_level_values('Ticker').isin(['META', 'TSLA', 'WMT', 'XOM',
'JPM', 'UNH', 'BA', 'AAPL', 'LIN', 'AMT', 'NEE'])]
df_corr.reset_index(inplace=True)
df_corr.dropna()
df_corr

# Pivot the DataFrame to have tickers as columns and dates as index
pivot_df = df_corr.pivot(index='Date', columns='Ticker', values='Daily Returns')
pivot_df.dropna()
pivot_df

# Compute the correlation matrix
correlation_matrix = pivot_df.corr()
# Create a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation of Top Stocks From each Sector')
plt.xlabel('Tickers')
plt.ylabel('Tickers')
plt.savefig('Heatmap.png')
plt.show()


# Selecting the best stocks based on the correlation matrix manually
df = df.loc[df.index.get_level_values('Ticker').isin(['META', 'TSLA', 'WMT', 'XOM', 'UNH'])]
df = df.drop(columns=['Dollar Volume', 'Daily Returns'])
df = df.unstack()
adj_close = df['Adj Close']
# Resample data to business days,filling missing values with previous day
adj_close                                                                                    =
adj_close[['META','TSLA','UNH','WMT','XOM']].asfreq('B').fillna(method='ffill')
adj_close.reset_index(inplace=True)
adj_close.head()

# Plotting closing prices over time
plt.figure(figsize=(14, 5))
plt.plot(adj_close['Date'], adj_close['META'], label='META', color='blue')
plt.plot(adj_close['Date'], adj_close['TSLA'], label='TSLA', color='green')
plt.plot(adj_close['Date'], adj_close['UNH'], label='UNH', color='red')
plt.plot(adj_close['Date'], adj_close['WMT'], label='WMT', color='purple')
plt.plot(adj_close['Date'], adj_close['XOM'], label='XOM', color='brown')
plt.title('Adjusted Closing Prices Over Time',fontsize=14)
plt.xlabel('Date',fontsize=12)
plt.ylabel('Adjusted Closing Price',fontsize=12)
plt.legend(fontsize=12)
plt.savefig('figure1.png')
plt.show()

# Filter the DataFrame to include only data till 2024-02-10
selected_data = adj_close[adj_close['Date'] <= '2024-02-10']
selected_data.set_index('Date', inplace=True)
selected_data.head()
selected_data.describe()
selected_data.reset_index(inplace=True)

# Scaling data to a specified range using MinMaxScaler
def scale_data(data, feature_range=(0, 1)):
    scaler = MinMaxScaler(feature_range=feature_range)
    scaled_data = scaler.fit_transform(data.values.reshape(-1, 1))
    return scaled_data, scaler

# Splits scaled data into training and testing
def split_data(scaled_data, look_back=80, train_ratio=0.80, batch_size=20):
    train_size = int(len(scaled_data) * train_ratio)
    train_data = scaled_data[:train_size]
    test_data = scaled_data[train_size - look_back:]

    train_generator = TimeseriesGenerator(train_data, train_data,
                                          length=look_back, batch_size=batch_size)
    test_generator = TimeseriesGenerator(test_data, test_data,
                                         length=look_back, batch_size=1)

    return train_generator, test_generator, test_data

# Build LSTM model
def build_lstm_model(train_generator, epochs=100):
    lstm_model = Sequential()
    lstm_model.add(LSTM(units=100, return_sequences=True,
                        input_shape=(train_generator.length, 1)))
    lstm_model.add(Dropout(0.2))
    lstm_model.add(LSTM(units=100, return_sequences=True))
    lstm_model.add(Dropout(0.2))
    lstm_model.add(LSTM(units=100))
    lstm_model.add(Dropout(0.2))
    lstm_model.add(Dense(1))
    lstm_model.compile(optimizer='adam', loss='mean_squared_error')
    lstm_model.fit(train_generator, epochs=epochs)
    return lstm_model


 # Build BI-LSTM model
def build_bi_lstm_model(train_generator, epochs = 100, look_back = 80):
    model = Sequential()
    model.add(Bidirectional(LSTM(units=100,                    return_sequences=True,
input_shape=(look_back, 1))))
    model.add(Dropout(0.2))
    model.add(Bidirectional(LSTM(units=100, return_sequences=True)))
    model.add(Dropout(0.2))
    model.add(Bidirectional(LSTM(units=100)))
    model.add(Dropout(0.2))
    model.add(Dense(1))
    model.compile(optimizer='adam', loss='mean_squared_error')
    model.fit(train_generator, epochs=epochs)
    return model


# Build GRU model
def build_gru_model(train_generator, epochs=100, look_back=80):
    model = Sequential()
    model.add(GRU(units=100, return_sequences=True, input_shape=(look_back, 1)))
    model.add(Dropout(0.2))
    model.add(GRU(units=100, return_sequences=True))
    model.add(Dropout(0.2))
    model.add(GRU(units=100))
    model.add(Dropout(0.2))
    model.add(Dense(1))
    model.compile(optimizer='adam', loss='mean_squared_error')
    model.fit(train_generator, epochs=epochs)
    return model
```

```
# Generates predictions using the model and test generator
def generate_predictions(model, test_generator, scaler, test_data, look_back=80):
   predictions = model.predict(test_generator)
   predictions = scaler.inverse_transform(predictions)
   actual_prices = scaler.inverse_transform(test_data[look_back:])
   return predictions, actual_prices

# Function to predict future
def predict_future_days(model, base_data, days_to_predict, scaler, look_back = 80):
   # Start with the last days of the base data
   input_data = base_data[-look_back:].reshape(1, look_back, 1)

   # Predict future days
   future_predictions = []

   for _ in range(days_to_predict):
      # Make a prediction
      prediction = model.predict(input_data)

      # Append the prediction
      future_predictions.append(prediction[0, 0])

      # Update the input data to include the prediction and exclude the oldest data point
      # Correctly reshape the prediction to (1, 1, 1) before appending
      input_data = np.append(input_data[:, 1:, :], prediction.reshape(1, 1, 1), axis=1)

   # Invert the scaling
   future_predictions = scaler.inverse_transform(np.array(future_predictions).reshape(-1, 1))

   return future_predictions

# Visualizes the actual and predicted stock prices
def visualize_stock_prices(stock_name, actual_prices, predictions, font_size=12):
   plt.figure(figsize=(15, 5))
   plt.plot(actual_prices, color='red', label=f'{stock_name} Stock Price with LSTM')
   plt.plot(predictions, color='green', label=f'Predicted {stock_name} Stock Price with LSTM')
   plt.title(f'{stock_name} Stock Price Prediction',fontsize=font_size + 2)
   plt.xlabel('Time',fontsize=font_size)
   plt.ylabel(f'{stock_name} Stock Price',fontsize=font_size)
   plt.legend(fontsize=font_size)
   plt.grid(True)
   plt.savefig(f'{stock_name}figure Prediction.png')
   plt.show()


# Plots historical and predicted future stock prices
def plot_stock_price_prediction(stock_name, adj_close, selected_data, days_to_predict,
future_days, font_size=12):
   last_date = selected_data['Date'].iloc[-1]
   predicted_dates = pd.date_range(start=last_date, periods=days_to_predict)
   plt.figure(figsize=(15, 5))
```

```
    plt.plot(adj_close['Date'], adj_close[stock_name], color='blue', label=f'Historical Daily
Closing Price of {stock_name}')
    plt.plot(predicted_dates, future_days, color='red', label=f'Predicted Future Price of
{stock_name} for the Next {days_to_predict} Days', linestyle='dashed')
    plt.title(f'{stock_name} Stock Price Prediction for the Next {days_to_predict}
Days',fontsize=font_size + 2)
    plt.xlabel('Time',fontsize=font_size)
    plt.ylabel(f'{stock_name} Stock Price',fontsize=font_size)
    plt.legend(fontsize=font_size)
    plt.grid(True)
    plt.savefig(f'{stock_name}_Future_Prediction.png')
    plt.show()

days_to_predict = 90
# Top 5 S&P 500 stocks
top_tickers = ['META','TSLA','UNH','WMT','XOM']

# Process and evaluate LSTM predictions for a list of stocks
for stock_name in top_tickers:

    lstm_start_time = time.time()

    # Assuming y and scaler are properly initialized
    y, scaler = scale_data(selected_data[stock_name])
    train_generator, test_generator, test_data = split_data(y)

    # Build LSTM model
    lstm_model = build_lstm_model(train_generator)

    # End time
    lstm_end_time = time.time()

    # Generate LSTM predictions
    lstm_predictions, lstm_actual_prices = generate_predictions(lstm_model, test_generator,
scaler, test_data)

    # Predict future days
    future_days_lstm = predict_future_days(lstm_model, y, days_to_predict, scaler,look_back
= 80)


    # Calculate time taken
    lstm_training_time = lstm_end_time - lstm_start_time

    # Calculate evaluation metrics
    lstm_rmse = np.sqrt(mean_squared_error(lstm_actual_prices, lstm_predictions))
    lstm_mae = mean_absolute_error(lstm_actual_prices, lstm_predictions)

    print(f"Stock: {stock_name} (LSTM) ")
    print(f"Training Time: {lstm_training_time} seconds")
    print(f"RMSE: {lstm_rmse}")
    print(f"MAE: {lstm_mae}")
    # Visualize results for LSTM model
    visualize_stock_prices(f'{stock_name} (LSTM)', lstm_actual_prices, lstm_predictions)

    plot_stock_price_prediction(f'{stock_name}', adj_close, selected_data, days_to_predict,
future_days_lstm)


# Process and evaluate Bi-LSTM predictions for a list of stocks
for stock_name in top_tickers:

    bi_lstm_start_time = time.time()

    # Assuming y and scaler are properly initialized
    y, scaler = scale_data(selected_data[stock_name])
    train_generator, test_generator, test_data = split_data(y)

    # Build Bi-LSTM model
    bi_lstm_model = build_bi_lstm_model(train_generator)

    # End time
    bi_lstm_end_time = time.time()

    # Generate Bi-LSTM predictions
    bi_lstm_predictions, bi_lstm_actual_prices = generate_predictions(bi_lstm_model,
test_generator, scaler, test_data)

    # Predict future days
    future_days_bi_lstm = predict_future_days(bi_lstm_model, y, days_to_predict, scaler,
look_back = 80)

    # Calculate time taken
    bi_lstm_training_time = bi_lstm_end_time - bi_lstm_start_time

    # Calculate evaluation metrics
    bi_lstm_rmse = np.sqrt(mean_squared_error(bi_lstm_actual_prices, bi_lstm_predictions))
    bi_lstm_mae = mean_absolute_error(bi_lstm_actual_prices, bi_lstm_predictions)

    print(f"Stock: {stock_name} (Bi-LSTM) ")
    print(f"Training Time: {bi_lstm_training_time} seconds")
    print(f"RMSE: {bi_lstm_rmse}")
    print(f"MAE: {bi_lstm_mae}")

    # Visualize results for Bi-LSTM model
    visualize_stock_prices(f'{stock_name} (Bi-LSTM)', bi_lstm_actual_prices,
bi_lstm_predictions)

    plot_stock_price_prediction(f'{stock_name}', adj_close, selected_data, days_to_predict,
future_days_bi_lstm)


# Process and evaluate GRU predictions for a list of stocks
```

```
for stock_name in top_tickers:

   gru_start_time = time.time()

   # Assuming y and scaler are properly initialized
   y, scaler = scale_data(selected_data[stock_name])
   train_generator, test_generator, test_data = split_data(y)

   # Build GRU model
   gru_model = build_gru_model(train_generator)

   # End time
   gru_end_time = time.time()

   # Generate GRU predictions
   gru_predictions,   gru_actual_prices   =   generate_predictions(gru_model,   test_generator,
scaler, test_data)

   # Predict future days
   future_days_gru = predict_future_days(gru_model, y, days_to_predict, scaler, look_back =
80)

   # Calculate time taken
   gru_training_time = gru_end_time - gru_start_time

   # Calculate evaluation metrics
   gru_rmse = np.sqrt(mean_squared_error(gru_actual_prices, gru_predictions))
   gru_mae = mean_absolute_error(gru_actual_prices, gru_predictions)

   print(f"Stock: {stock_name} (GRU) ")
   print(f"Training Time: {gru_training_time} seconds")
   print(f"RMSE: {gru_rmse}")
   print(f"MAE: {gru_mae}")

   # Visualize results for GRU model
   visualize_stock_prices(f'{stock_name} (GRU)', gru_actual_prices, gru_predictions)

   plot_stock_price_prediction(f'{stock_name}',  adj_close,  selected_data,  days_to_predict,
future_days_gru)
```