University of Reading

Department of Computer Science

A comparison of machine learning models in predicting Stock Market prices.

Al Shaima Said Yaqoob Al Shuaili

Supervisor: Muhammad Shahzad

A report submitted in partial fulfilment of the requirements of
the University of Reading for the degree of
Bachelor of Science in Computer Science

May 2, 2023

Declaration

I, Al Shaima Said Yaqoob Al Shuaili, of the Department of Computer Science, University of Reading, confirm that this is my own work and figures, tables, equations, code snippets, artworks, and illustrations in this report are original and have not been taken from any other person's work, except where the works of others have been explicitly acknowledged, quoted, and referenced.
I understand that if failing to do so will be considered a case of plagiarism. Plagiarism is a form of academic misconduct and will be penalised accordingly.

I give consent to a copy of my report being shared with future students as an exemplar.

I give consent for my work to be made available more widely to members of UoR and public with interest in teaching, learning and research.

Al Shaima Said Yaqoob Al Shuaili
May 1, 2023

Abstract

Given the complexity and dynamic nature of the stock market, forecasting its behaviour is considered a difficult task. Machine learning has become an effective tool for stock market prediction, offering precise forecasts and assisting with investment decisions. This research examines the use of machine learning for stock market forecasting, with a focus on the most popular models including Regressing, Ensemble, Recurrent Neural Networks, and Convolutional Neural Networks. The report also discusses the steps involved in preparing the data for the models and for evaluating the performance of the used models.

In the report, the current state of research is mentioned, by assessing the strengths and challenges of the implemented models in stock market prediction. Future initiatives for study are also presented in the report. These include examining the efficacy of more recent machine learning models, creating hybrid models that incorporate the best features of several approaches, and examining how economic indicators and news sentiment analysis affect stock prices.

Keywords: Artificial Intelligence, Machine Learning, Deep Learning, Stock Market Prediction, Regression, Random Forest, ARIMA, LSTM, CNN, Decision Tree Classifier.

Acknowledgements

I want to start by thanking my supervisor Muhammad Shahzad for his advice and assistance during the course of this project. Additionally, I would like to thank my family and friends for their constant support, and for believing in me. Also, I want to express my appreciation to my academic tutor Atta Badii and any other professors who helped me complete this project.

Contents

1 Introduction 11

1.1 Background 11

1.2 Problem statement 12

1.3 Aims and objectives 12

1.4 Solution approach 12

1.5 Outline of the report 12

2 Literature Review 12

2.1 Theories and Solutions 13

2.2 Application 15

2.3 Summary 16

3 Methodology 17

3.1 Task description 17

3.1.1 Description of the dataset 17

3.1.2 Data Pre-processing 18

3.1.3 Exploratory Data Analysis (EDA) 18

3.2 Algorithms Description 21

3.2.1 Machine Learning Models 22

3.2.2 Deep Learning Models 23

3.2.3 Decision Tree Classifier 25

3.2.4 Evaluation Metrics 25

3.3 Implementations 26

3.4 Hardware and Running Times 27

4 Results 28

4.1 Linear Regression 28

4.2 Random Forest 29

4.3 ARIMA 30

4.4 LSTM 32

4.5 CNN 33

4.6 Module Evaluation 34

4.7 Decision Tree Classifier 35

5 Discussion and Analysis 36

5.1 Significance of the findings 36

5.2 Limitations 40

5.3 Summary 40

6 Conclusion and Future Work 41
6.1 Conclusions 41
6.2 Future work 41
7 Reflection 43
8 Bibliography 44
9 Appendices 47
9.1 Appendix A: Results of EDA 47
9.2 Appendix B: Results of ML models 50
9.3 Appendix C: Classification results 59

List of Figures

Figure 1: information on the Apple dataframe 19
Figure 2: pairplot() on the Apple dataframe 19
Figure 3: correlation heatmap of the Apple dataframe 20
Figure 4: average price of Apple stocks across time 20
Figure 5: distribution graphs of open, high, low, and close prices on from the Apple dataframe 21
Figure 6: distribution and density graph of the return of Apple stocks 21
Figure 7: Random Forest flowchart (Lin, et al., 2020)) 22
Figure 8: ARIMA model flowchart (Moro, et al., 2020) 23
Figure 9: structure of LSTM (Le, et al., 2019) 24
Figure 10: decision tree algorithm flowchart (Arian, n.d.) 25
Figure 11: visualising the Linear Regression on Apple Stocks 28
Figure 12: visualising the actual and predicted results of Random Forest on Apple Stocks 29
Figure 13: Linear Regression of Random Forest on Apple stocks 29
Figure 14: plot showing differencing 30
Figure 15: ACF and PACF plots on Apple stocks 30
Figure 16: visualising the actual and forecast results of ARIMA on Apple Stocks 31
Figure 17: Linear Regression of ARIMA on Apple stocks 31
Figure 18: actual vs predicted results of LSTM on Apple stocks 32
Figure 19: Linear Regression of LSTM on Apple stocks 32
Figure 20: actual vs predicted results of CNN on Apple stocks 33
Figure 21: Linear Regression of CNN 33
Figure 22: confusion matrix of decision tree classifier results (Apple dataset) 35
Figure 23: decision tree visualisation (Apple dataset) 35
Figure 24: confusion matrix division clarification (Zuhaib, 2019) 39
Figure 25: five-row sample of apple stock dataframe 47
Figure 26: five-row sample of alphabet stock dataframe 47
Figure 27: five-row sample of Microsoft stock dataframe 47
Figure 28: information of apple stock dataframe 48
Figure 29: information of Alphabet stock dataframe 48
Figure 30: information of Microsoft stock dataframe 48
Figure 31: five-row sample of Apple dataframe after data pre-processing 49
Figure 32: five-row sample of Alphabet dataframe after data pre-processing 49
Figure 33: five-row sample of Microsoft dataframe after data pre-processing 49
Figure 34: five-row sample of apple dataframe after adding new column 'Decision' 49
Figure 35: five-row sample of Alphabet dataframe after adding new column 'Decision' 50
Figure 36 :five-row sample of Microsoft dataframe after adding new column 'Decision' 50
Figure 37: Linear Regression on Alphabet Stocks 50
Figure 38: Linear Regression on Microsoft Stocks 51
Figure 39: actual vs predicted results of Random Forest on Alphabet stocks 51
Figure 40: Linear Regression of Random Forest on Alphabet stocks 52
Figure 41: actual vs predicted results of Random Forest on Microsoft stocks 52
Figure 42: Linear Regression of Random Forest on Microsoft stocks 53
Figure 43: differencing plot - Alphabet stocks 53
Figure 44: ACF & PACF plots - Alphabet stocks 53
Figure 45: visualising actual vs forecast results of ARIMA on Alphabet stocks 54
Figure 46: Linear Regression of ARIMA on Alphabet stocks 54

Figure 47: differencing plot - Microsoft stocks 55
Figure 48: ACF & PACF plots - Microsoft stocks 55
Figure 49: visualising actual vs forecast results of ARIMA on Microsoft stocks 55
Figure 50: Linear Regression of ARIMA on Microsoft stocks 56
Figure 51: LSTM results on Alphabet dataset 56
Figure 52: Linear Regression of LSTM on Alphabet dataset 56
Figure 53: LSTM results on Microsoft dataset 57
Figure 54: Linear Regression of LSTM on Microsoft dataset 57
Figure 55: CNN results on Alphabet dataset 57
Figure 56: Linear Regression of CNN on Alphabet dataset 58
Figure 57: CNN results on Microsoft dataset 58
Figure 58: Linear Regression of CNN on Microsoft dataset 58
Figure 59: Confusion Matrix of Decision Tree results - Alphabet dataset 59
Figure 60: decision tree visualisation - Alphabet dataset 59
Figure 61: Confusion Matrix of Decision Tree results - Microsoft dataset 60
Figure 62: decision tree visualisation - Microsoft dataset 60

List of Tables

Table 1: dataset columns’ descriptions 18
Table 2: running time comparison 27
Table 3: model evaluation (rounded to 2 decimal places) 34
Table 4: decision tree classifier model evaluation (rounded to 2 decimal places) 36

List of Abbreviations

ACF – Autocorrelation Function

ADF - Augmented Dickey-Fuller

AI – Artificial Intelligence

ANN – Artificial Neural Network

ANN – Artificial Neural Networks

AR - Autoregression

ARIMA – Autoregressive Integrated Moving Average

CNN – Convolutional Neural Network

DL – Deep Learning

EDA – Exploratory Data Analysis

GRU – Gated Recurrent Unit

LSTM – Long-Short Term Memory

MA – Moving Average

MAE – Mean Absolute Error

MAPE – Mean Absolute Percentage Error

ML – Machine Learning

ML – Machine Learning

MSE – Mean Square Error

PACF – Partial Autocorrelation Function

RBF – Radial Basis Function

RMSE – Root Mean Square Error

RNN – Recurrent Neural Network

SVM – Support Vector Machine

SVR – Support Vector Regression

1 Introduction

The stock market is an essential component of the world economy, and its movements have a big impact on people's and countries' financial security (Shah, et al., 2022). To benefit from the stock market, investors risk their chances by buying low and selling high (Shah, et al., 2022). But making future stock price predictions is a difficult task that needs a lot of information and expertise. Researchers are figuring out how to employ prediction tools in the stock market successfully because of technological advancements, particularly those in Artificial Intelligence (AI). Lowering risks not only helps investors, but also helps businesses draw in new investors (Shah, et al., 2022). Researchers' duty, in this situation, is to develop creative ways to use technology to support the stock market. Data analysis in the financial industry can look at and pinpoint theories, which in turn can highlight elements that have an impact on consumer behaviour and purchase choices. Making informed judgements about their product design, price strategy, distribution strategy, marketing strategy, and promotion strategy can be aided by these choices (Shelley, 2020).

To invest in the stock market, a significant amount of data is needed (Shah, et al., 2022). Several kinds of raw data can be taken into consideration, but not all of them can be used. Market data, text data, macroeconomic data, knowledge graph data, picture data, fundamental data, and analytics data have all been employed in recent years to forecast stock market prices. The use of prediction techniques in the stock market is essential for bringing new and existing investors together, as well as for understanding the nature and operations of the market by monitoring current market trends. This calls for the use of cutting-edge techniques like technical analysis and artificial neural networks (ANNs), which can capture data correlations and look at historical data (Shah, et al., 2022).

Machine Learning (ML) is a type of AI which delivers remarkable advancements in the field of business. It can recognise trends, learn from historical data, and forecast stock price. Ultimately, automating statistical data processing through the application of ML makes financial market forecasting possible (Shah, et al., 2022). There are many ML models that have been used for this field. This research seeks to assess the most suitable ML model for stock market forecasting.

1.1 Background

The financial sector and the global economy can be significantly impacted by accurate stock market forecast. Accurate forecasts can lower risk, boost returns, and assist traders and investors in making more educated decisions. Additionally, they can offer insights into market dynamics and trends, which can aid decision-making by regulators and policymakers.

For many years, investors and traders have been interested in stock market forecasting as a way to achieve financial growth. To increase the precision of their forecasts, researchers and analysts have turned to ML techniques over time. The use of ML in stock market forecasting is not new, but with the emergence of big data, cloud computing, and more effective algorithms, its applications and potential have increased tremendously. ML is being employed in a variety of stock market applications, such as portfolio optimisation, risk management, and price prediction. By locating patterns and correlations in huge datasets, ML approaches have demonstrated promising results in the prediction of stock values. Massive amounts of historical market data may be processed and analysed by ML models, and forecasts can be made based on that analysis. Decision trees, artificial neural networks, support vector machines, random forests, and deep learning models like convolutional neural networks and recurrent neural networks have all been utilised for stock market prediction. Each method has its advantages and disadvantages in terms of strength, accuracy, difficulty, and speed.

There are still several difficulties and gaps in the existing research despite the potential advantages of ML in stock market prediction. For instance, the quality and broadness of the data used for training and testing machine learning models impact their accuracy. Additionally, it is extremely difficult to predict market trends and make precise predictions due to the stock market's complexity and dynamic nature.

1.2 Problem statement

There is still disagreement over the most effective model despite multiple studies on applying machine learning algorithms to predict stock market prices. Investors and traders who depend on precise projections to make wise judgements are confused by this. Therefore, a thorough evaluation of machine learning models is required to identify the most precise and trustworthy technique for forecasting stock market prices. This study will evaluate the effectiveness of several ML algorithms and pinpoint each model's advantages and disadvantages. The objective is to shed light on the effectiveness of several ML models in predicting stock prices and determine the most appropriate model for precise and trustworthy stock market predictions.

1.3 Aims and objectives

This paper aims to compare how well various machine learning models perform in forecasting stock market prices. In order to accomplish this goal, the report's objectives include identifying and selecting a variety of machine learning models frequently used to forecast stock market prices, gathering, and analysing useful information on stock market prices and other relevant factors, training and testing the selected models using the gathered data, and evaluating and comparing the models' performance using a set of predetermined metrics.

1.4 Solution approach

This research is tackled by choosing and putting into practice various machine learning models, training them on historical stock market data, and assessing their performance using various metrics. Regression models, decision trees, neural networks, and ensemble techniques can all be used as models. The study's findings shed light on the best machine learning models for stock price prediction as well as their drawbacks in practical situations.

1.5 Outline of the report

In this report, a literature review is given at first in section 2. This section would discuss previous theories and solutions in published literature and how these has been applied in this research. Then, the methodology is given in section 3. This section describes the task implemented with the choice of datasets, pre-processing steps, and the exploratory data analysis (EDA) tasks performed. In the same section, the algorithms are described, including the machine learning models and the evaluation metrics used. It also includes the implementation steps, a description of the hardware used, and a comparison of the running duration of the models. Next, the results are given in section 4. This includes any tables, graphs and numerical values given by the models and the evaluation metrics. In section 5, the results are discussed and evaluated according to the results of the literature review. This shows the significance of the findings and considers he limitations of the project. Finally, the report is concluded in section 6, with mentions of potential advancements that can be applied in future projects.

2 Literature Review

This section highlights the project's importance. The literature displayed in this segment show implementations of ML in stock market prediction. Most of the academic literature emphasize on the potential that ML opens for the stock prices prediction. Additionally, some literature compares and evaluates the best ML models to use. The purpose of this literature review is to identify significant topics related to the integration of machine learning models in the stock market and consider several existing stock market prediction techniques that are adapted in research.

2.1 Theories and Solutions

The application of machine learning on prediction of stock market prices advanced throughout the years. This has been proven by recent research and studies published online. Some literature compares the models to use for stock price prediction in consideration of the market data. These studies show how a certain model is implemented with the given accuracy tests and associates the produced results with results from other models. This is to determine which model is better at handling time-series data that is highly random, such as stock prices. As an example, (Reddy, 2018) and (Sunny, et al., 2020) discussed models that can be used on historical market data, to predict trends and prices. On the other hand, there are literature that use text data, such as sentiment analysis, to enhance the accuracy of the given predictions, while assessing the models for the most suitable in these kinds of scenarios. This can be seen in the research done by (Shah, et al., 2022) and (Verma, et al., 2017).

Lussange et al. (2019) address the Artificial Intelligence method known as the Multi-Agent System. Each agent in this system has these three essential qualities: A reinforcement learning algorithm, a learning procedure to use fundamentals for stock price evaluation, and the integration of the trader's behavioural factors. It is suggested that this agent can be used for stock market prediction (Shah, et al., 2022).

The categories of raw data that can be considered for stock prediction are market data, text data, macroeconomic data, knowledge graph data, image data, fundamental data, and analytics data. Market data includes trading activities, while text data is contributed by individuals, such as social media, news, and web searches, which can be used for sentiment analysis. Macroeconomic data provides information on the economic circumstances of a country, region, or sector. Knowledge graph data deals with the relationships between different companies and markets. Image data uses candlestick charts as input images for stock prediction. Fundamental data includes accounting data reported quarterly, while analytics data comes from reports and recommendations provided by investment banks and research firms. However, not all categories of data can be applied in all studies, and the low frequency and inaccuracies of reporting in fundamental data pose a limitation for its use in deep learning models (Jiang, 2021).

According to the efficient-market hypothesis, many researchers confirm that market data considers all relevant information for price prediction. However, other sources of data have been used to forecast stock market outcomes. For example, Liu, Lu, and Du (2019) use market data, fundamental data, knowledge graphs, and news for their analysis. Weng, Ahmed, and Megahed (2017) assess the effectiveness of market data, technical indicators, Wikipedia traffic, Google news counts, and produced features.

Shah, et al. (2022) presented the most recent developments in this subject and examined the use of machine learning techniques for predicting stock market prices. The researchers mentioned an imprtant note about the assumptions that are made when using market data for stock price prediction. “(1) the stock price reflects all knowledge; (2) the stock price fluctuates following particular rules; and (3) A similar circumstance is likely to arise in the future.”

According to Shah, et al. (2022), researchers utilize linear prediction methods to forecast short-term stock prices using large historical information on statistics and probability of the dataset. However, the changes in stock prices are usually considered nonlinear. To solve this problem, the trend is given to ML models with an indicator, and the ML model predicts the future trend of the prices. On the other hand, to perform time series analysis on stock prices, the linearity of the data must be ensured. It is suggested for this to be done by confirming that the statistical properties such as the mean, variance, et cetera, do not change over time (Shah, et al., 2022).

Liu, et al., (2018) suggests that there are three analysis methods for stock investment: basic analysis, technical analysis, and evolutionary analysis. Technical analysis uses various statistical methods to analyze and predict market trends. It does that by looking at historical data, primarily price and volume, to attempt to forecast price movement and the overall trend (Seth, 2022). But the non-linearity of stocks and the complex principle on which the prices are determined makes it challenging for traditional prediction methods. This is supported by the research mentioned in Liu, et al. (2018), which suggests that the use of cluster analysis does not give reliable, accurate predictions. On the other hand, ANN is said to be a great option, as it shows "nonlinear mapping ability, good self-learning and adaptive performance". Shah, et al. (2022) also suggests that the sentiments of society constantly have an impact on the stock market since traditional methods of analysis only consider the financial perspective.

Nevertheless, with the suggestion of Jiang (2021), market dat ais considered in this research. By comparing the publiched research online, the ML models used in this research is determined. Htun, et al., (2023) examined literature that used Random Forest for stock market prediction. They concluded that by using common technical indicators, Random Forest provides acceptable prediction results. Previous published literature has also shown that regression is used for predicting stock values and evaluating the used ML models. This idea is mentioned and supported in both Bhuriya, et al. (2017) and Altay & Satman (2005). In Bhuriya, et al. (2017), Linear Regression is used, which showed a confidence value of 0.9774, the highest in comparison to polynomial and Radial Basis Function (RBF) regression methods. On the other hand, Altay & Satman (2005) show how regression can be used to evaluate other models, then used Mean Absolute Error (MSE), Root Mean Square Error (RMSE), and Mean Absolute Error (MAE) to evaluate the regression of the models applied.

Reddy (2018) investigated the training of ML models with previous market data to predict stock index movements. They also proposed several algorithms, mainly built on Support Vector Machine (SVM) and Radial Basis Function (RBF), that forecast the daily trend of market stocks. Althelaya, et al. (n.d.) mentioned a study by Yao, et al. (2017) that compares Support Vector Regression (SVR), SVM RBF, and Recurrent Neural Network (RNN). The results show that Gated Recurrent Unit (GRU), which is a type of RNN, performed better than other models in prediction, but SVM RBF gave higher accuracy scores.

Parmer, et al. (2018) evaluated the use of Linear Regression and Long-Short Term Memory (LSTM) on data from Yahoo Finance. In their study, LSTM showed better efficiency than Linear Regression, with an MSE score of 0.00875 and an R-square value of 0.86625 respectively. Additionally, they found that the accuracy of the predictions increases with the use of a larger dataset, and suggest that the use of other machine learning algorithms would work well on stock market data. Verma, et al. (2017), used SVM and LSTM to predict the effect of sentiment on stock prices. Sentiment analysis performed considered information available in relevent news articles. In this research, LSTM proved to be superior in performance than SVM on several stock datasets. Furthermore, Shah, et al. (2022) show how Autoregressive Integrated Moving Average (ARIMA) and LSTM have been used previously in literature and what methods were used for module evaluation. Shah, et al. (2022) argues that the

RMSE values demonstrate that LSTM algorithm produces a better outcome than ARIMA models, in terms of stock price prediction, by a significant margin between 84 and 87 percent.

Rout, et al. (2022) implement LSTM for Amazon, Google, Microsoft, and Apple stocks. The model is evaluated by calculating the loss using MSE. As a result, the accuracy is 0.84, 0.601, 0.605, and 0.685 for Amazon, Google, Microsoft, and Apple respectively. Moreover, Sunny, et al. (2020) applied LSTM for data analysis of Google stock prices. They used RMSE as an evaluation metric for multiple epochs. Their objective was to evaluate the LSTM for the best number of epochs. They confirmed that the training time increases with the increased number of hidden layers, and the increased number of hidden layers decrease the efficiency of the testing accuracy (Sunny, et al., 2020). In addition, Liu, et al. (2018) found that using multiple layers in LSTM increases the accuracy of the predictions. However, using more than five layers proved to be computationally expensive. Hence, they used three layers, and the accuracy result is given as 72%. Alternatively, Lee & Soo (2017) tuned LSTM algorithm parameters through experiments. They used grid search to find the best hyperparameters for the model.

Jiang (2021) listed combinations of CNN and RNN structures that were built for time-series prediction. For example, TreNet, which is a hybrid system of CNN and LSTM for stock trend classification. Similarly, other models were developed and used in Lee & Soo (2017) and Vargas, et al. (2017). Vargas, et al. (2017) compared between several types of RNN and CNN used together. The training and testing prediction evaluation results showed that RNN is a more effective model than CNN for index prediction. In a comparison between Bidirectional LSTM (B-LSTM) and Recurrent Convolutional Neural (RCN) models, the RMSE value showed greater result for four out of five of the datasets in the B-LSTM model (Lee & Soo, 2017). Another proposed DL model is a combination of three constructing algorithms: CNN layer, Inception module, and LSTM layer (Jiang, 2021).

As declared by Jiang (2021), DL models have demonstrated superior performance to both linear and machine learning models on tasks like stock market prediction due to their great capacity of handing large amounts of data, and ability to learn the nonlinear relationship between input features and predicted target.

A strategy by Vaiz & Ramaswami (2016) suggests the following: price variation can be utilised to predict the buy or sell trading decision. The price change is a binary classification variable which calculates the difference between the close price and the open price. If the price change is greater than zero, then the classifier indicates a buy signal. Otherwise, if the difference is less than zero, the decision tree must give a sell signal. This research conducted a comparison between different decision tree classifiers. The results given by this research showed and F-Measure of 0.8808 for C5.0.

By reflecting on the research done for the literature review, the steps in the implemented methodology of this report can be justified. The applications of these mentioned literature are provided in the following section.

2.2 Application

The use of prediction tools in the stock market can be quite effective in attracting more investors. This is beneficial to both the companies on the stock market and investors. The role of researchers at the moment lies in finding methods of exploiting technology, specifically AI, to enhance and support the stock market. As investors risk their money in trading, new technology can help them reduce the risks by understanding the future movement of the stock. There are many algorithms that can be used for stock market prediction, and most of them require improvements for application in this field. This is why it is crucial to evaluate the best method of predicting stock market prices.

Methods used in previous literature include regression, ARIMA, LSTM, and CNN models. Each of these models have been used in multiple scenarios and compared against each other in terms of prediction accuracy and overall performance. The general workflow of the models is as follows: given the data from the previous few days, the models are used to forecast the values of the forthcoming days. As long as the dataset is still valid, this method continues to repeat itself recursively (Mukherjee, et al., 2021).

Considering the literature mentioned in the previous subsection, the models to be used for this research is chosen. Linear regression is the first model selected. In many literatures, different types of regression are used for stock price prediction. But, in Bhuriya, et al. (2017), Linear Regression showed the best performance in comparison to other regression algorithms. Even though there is no linearity in stock prices, implementing Linear Regression to evaluate the linear relationship between the prices makes market data easier to understand and analyse. Moreover, Linear Regression is also used to evaluate the performance of the other models used. This is inspired by the work of Altay & Satman (2005). The Linear Regression is then evaluated with MAE, RMSE and Mean Absolute Percentage Error (MAPE) values.

Using LSTM showed better accuracy than several models in Parmer, et al. (2018), Verma, et al. (2017) and Shah, et al. (2022). The increased accuracy of LSTM is justified by the ability of keeping track of data from initial stages. Additionally, Rout, et al. (2022) showed how LSTM is used for technological stocks resulting in acceptable MSE values. This is implemented and tested in this study. Using a suitable number for epochs (Sunny, et al., 2020), and the suggested number of layers (Liu, et al., 2018), the parameters of the LSTM model in this paper is tuned accordingly.

The models Random Forest, ARIMA, LSTM and CNN are evaluated by a Linear Regression model (Altay & Satman, 2005) and MAE, RMSE and MAPE evaluation metrics. As listed in Shah, et al. (2022), each of these evaluation metrics have been used in at least four papers previously. These measures represent the error value given by the predictions of the algorithms used. The error calculates the accuracy of the models by determining the difference between the actual and predicted values of the test datasets using different methods.

Decision Tree Classifier

To make the most of the market data, represented by the open, high, low, and close prices of the stock, a new calculation is created for decision tree classifier application. This is to employ machine learning classifiers for giving reliable trading decisions (Sharma, et al., 2017). This algorithm is inspired by the work of Vaiz & Ramaswami (2016).

2.3 Summary

To prove the validation of the literature discussed above, and to apply their suggestions and test them, this report considers the use of Linear Regression, Random Forest, ARIMA, LSTM, and CNN. The literature review reflects on the previous research where these models are used for stock market prediction and compared for the performance of the models. Additionally, the accuracy of the predictions given by the models are checked using evaluation metrics MAE, RMSE, MAPE, et cetera. This research helps understand and develop a plan to tackle the aims of this project and provide the suitable methodology for implementation.

3 Methodology

3.1 Task description

As mentioned in (Shah, et al., 2022), a study has used the following stages for data manipulation: raw data collection, data pre-processing, feature extraction, model training, and output production. The data used in various studies is determined mostly on the location of the performed study and personal preference. As the research mainly considers the performance of the models themselves, the dataset does not make much of a difference as long as it fulfills basic conditions. The first and most important condition is that the dataset must be a stock prices dataset, whether it is a simulation or real historical data. This is because the research implemented concerns evaluating models for stock prices, so having a different dataset would not prove the applicability of research results on the stock market. Shah, et al. (2022) encourages the use of current stock price data for best results. Secondly, the dataset must be large enough to give reliable predictions and performance results (Parmer, et al., 2018). Thus, this project considers the use of data from stocks of the pioneers of the technology sector, such as in (Rout, et al., 2022) and (Sunny, et al., 2020). The field of technology is a dynamic sector that is always developing and growing, and the quick advancements influence many other businesses. This encourages investors to invest their money in this field, which is why these stocks are specifically considered for this project.

Data pre-processing is a crucial stage after selecting the dataset. It concerns preparing data for analysis by following certain steps. The steps of data pre-processing include cleaning, reduction, transformation, and segregation (Jain, 2023). Data cleaning considers checking the dataset for any missing, or irrelevant data. Then, the null values are either removed, or replaced with a statistical measure or a reasonable value (Banerejee, 2020). Furthermore, data reduction includes selecting a subset of the dataset that is relevant to the analysis problem. This is to avoid issues associated with underfitting and overfitting the ML models. Another data preprocessing step is transformation, which involves changing the data to a format that is ideal for analysis. Normalization is considered a form of data transformation. It is when the provided data is transformed to values between 0 and 1. This aids the training of the models. Then, after analysis, the data is retransformed to original values prior to data visualisation. Normalisation has been applied in many literatures including Sunny, et al. (2020).

Before defining the model, the dataset is split into a training set and a testing set. This can be chosen randomly, as long as the training set makes up most of the dataset. (Sunny, et al., 2020) used 88% of the dataset as training set, and 12% as testing set. The model is then defined, fitted with the train set, and used to predict the values of the test set. Then comes the model's performance evaluation using metrics from the available python libraries (Shah, et al., 2022).

3.1.1 Description of the dataset

Apple, Alphabet (Google), and Microsoft market data are chosen as the three datasets for this project. These datasets are retrieved from Yahoo Finance and downloaded from Kaggle in CSV format (Onyshchak, 2020). Each of the CSV files contain the following columns: Date, Open, High, Low, Close, Adj Close, and Volume (Table 1). The Apple dataset contains a total of 9909 entries, dating back from the end of 1980 to the start of January 2019. On the other hand. the Alphabet dataset contains a total of 3932 entries, dating back to 2004 and last updated on the beginning of 2019. Furthermore, the Microsoft dataset contains 8584 total entries, dating back to March 1980 to the start of year 2019.

Table 1: dataset columns' descriptions

COLUMN	DESCRIPTION
DATE	Date of trading day
OPEN	Opening price
HIGH	Maximum trading price during the day
LOW	Minimum trading price during the day
CLOSE	Closing price adjusted for splits
ADJ CLOSE	Adjusted closing price for both dividends and splits
VOLUME	The number of shares traded between the open and close prices.

In this project, Jupyter Notebook is used as a Python web data manipulation environment. These CSV files were converted into pandas Dataframes using the data analysis package, pandas, and the scientific computing library for Python, NumPy, for numerical manipulation (Bhuriya, et al., 2017).

3.1.2 Data Pre-processing

The stock price list is typically complex, noisy, unpredictable, and nonlinear. Therefore, financial forecasting of time series becomes difficult since it includes some complicated characteristics, such as volatility, inconsistencies, and shifting patterns (Gandhmal & Kumar, 2019). To ease the process of stock price analysis and forecasting, data pre-processing proves useful. First, reduction is applied to the dataset. From a selected period of time, a date range is selected to specify the required timeframe for the data in the dataset. Using insufficient data may increase the risk of underfitting, but using older data may produce outdated results. Thus, the chosen period is 10 years; from the start of the year 2009, to the end of the year 2019. This makes the dataframes consist of 2516 entries in total (Jiang, 2021).

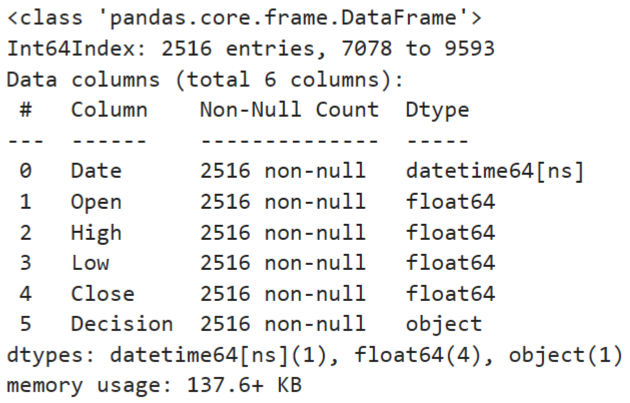
Then, data cleaning is performed. In the beginning, the unnecessary columns are dropped. The adjusted close and volume do not concern the prediction of the stock prices in the scenario of this project. Hence, the two columns are removed from the pandas dataframe. Next, duplicate rows are dropped. This is done by checking for duplicate rows, then removing duplicate, if any present, while keeping one unique row. Finally, the null values are dropped from the dataframe. Choosing to remove the null values instead of replacing them with a statistical calculation, such as the mean value, is due to the consequence that this change would influence on the predictions later on.

3.1.3 Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) refers to examining and analysing datasets to identify patterns, anomalies, and trends. EDA consists of overviewing the key attributes of the dataset and visualising the data. EDA is crucial for validating claims, verifying presumptions, and spotting patterns that may be hidden by simple statistical analysis (IBM, n.d.). In summary, EDA is to produce plots and make calculations to understand the values in the dataset and be able to make assumptions on the analysis to be performed.

The additional python modules used for this task are Matplotlib, Seaborn and SciPy. Matplotlib is a library for data visualisation, which produces plots and graphs, with the ability to customise the plots as preferred. Seaborn is another visualisation library built on Matplotlib specialised in producing complex plots with pre-calculated computations. It is usually associated with machine learning, as it grants further data investigation. In addition, SciPy is a library built on NumPy module, which contains statistical computations and a variety of functions.

The first step of EDA is displaying the general information of the dataframe. This includes the number of entries, the number of columns, the name, datatype, and non-null count of each column. There are six columns in total: Open, High, Low, Close, and Decision. Four of these columns are of the type 64-bit float. These are Open, High, Low and Close. The other two columns are datetime, for date column, and object, for decision column.



```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2516 entries, 7078 to 9593
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Date      2516 non-null   datetime64[ns]
 1   Open      2516 non-null   float64
 2   High      2516 non-null   float64
 3   Low       2516 non-null   float64
 4   Close     2516 non-null   float64
 5   Decision  2516 non-null   object
dtypes: datetime64[ns](1), float64(4), object(1)
memory usage: 137.6+ KB
```

Figure 1: information on the Apple dataframe

Furthermore, a pair plot is produced for the numerical columns. Pairplot is a method in Seaborn, which visualises the pairwise association between features of the dataset. By comparing each variable in the dataset to each other, this approach creates a grid of scatter plots and histograms. This is to display if the relationship is linear or non-linear and show outlier values and trends in the data.

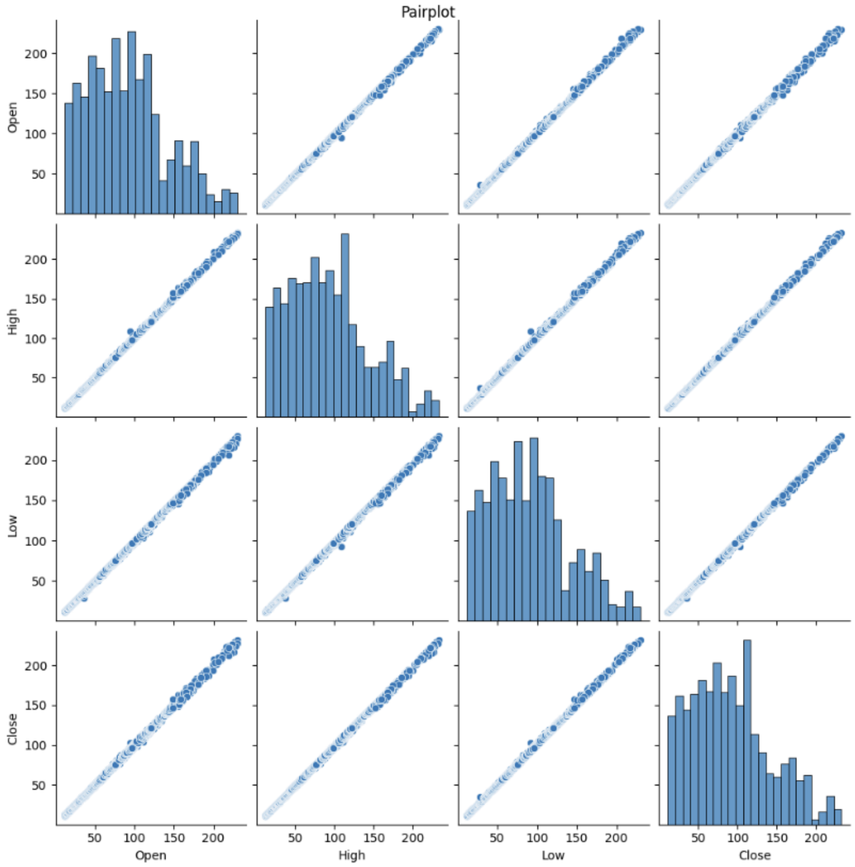


Figure 2: pairplot() on the Apple dataframe

Then, the correlation between the numerical values is evaluated. This shows the direction of the linear relationship between the variables, whether positive or negative correlation.

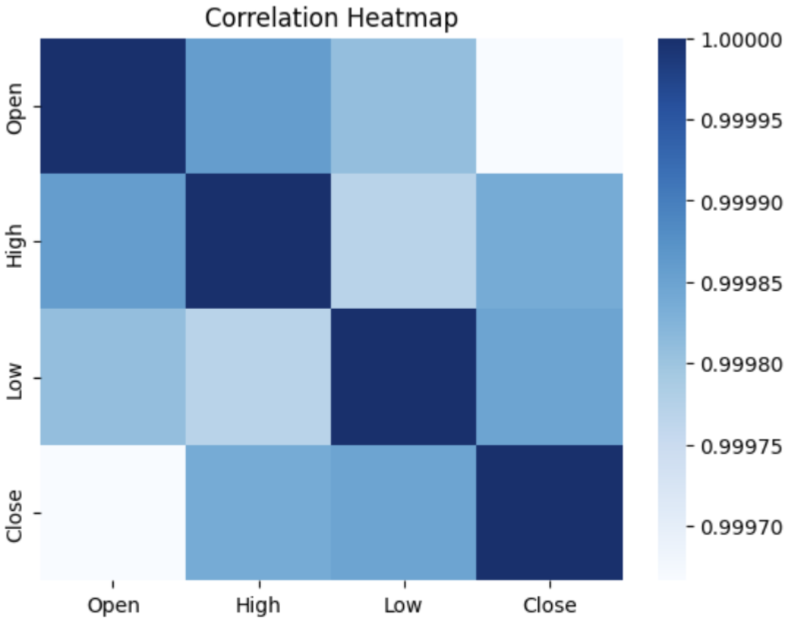


Figure 3: correlation heatmap of the Apple dataframe

Later on, the average price of the stock during the time period selected is calculated and presented in a line graph. This is calculated by considering the open and close prices of each day during the date range mentioned above.

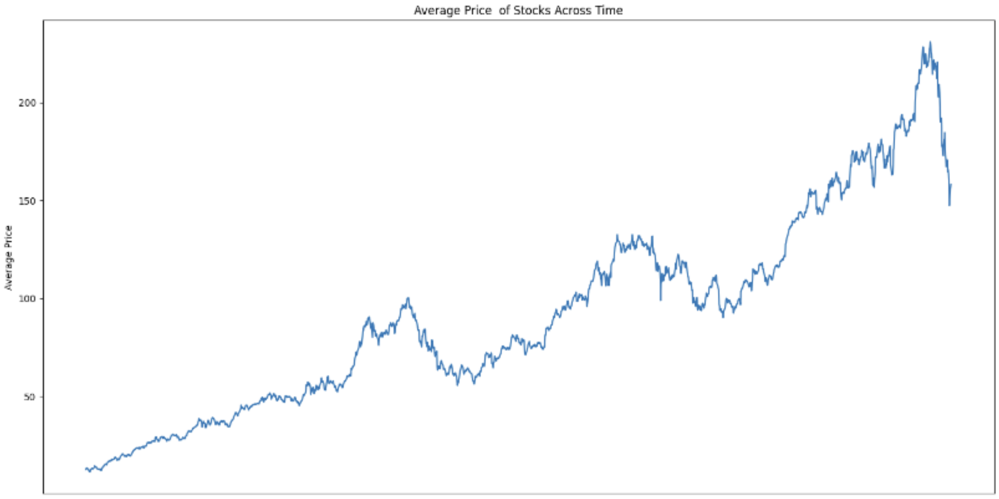


Figure 4: average price of Apple stocks across time

Moreover, a distribution histogram graph is plotted for each of the numerical columns; open, high, low, and close prices. A distribution graph shows the frequency of the values in the dataset. This can reveal information about the data's primary trend, spread, and skewness.

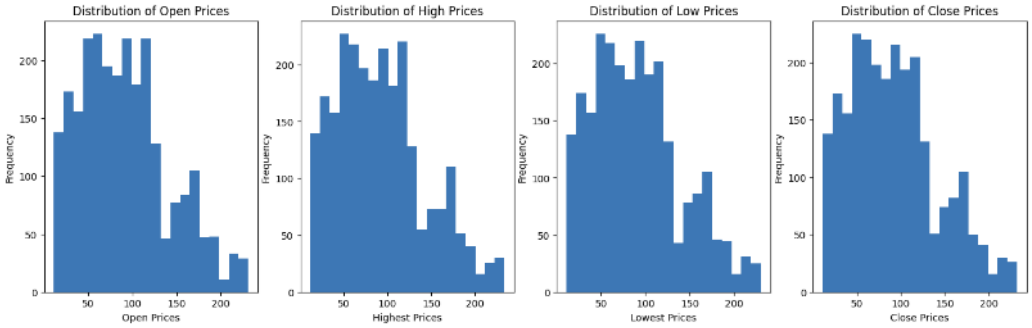


Figure 5: distribution graphs of open, high, low, and close prices on from the Apple dataframe

Afterwards, the return of the stocks is calculated. This is computed by figuring out the percentage change of the values in the chosen variable, close price. Then, the mean, standard deviation, skewness, and kurtosis are evaluated for the returns. And a distribution and density plot is created for the returns.

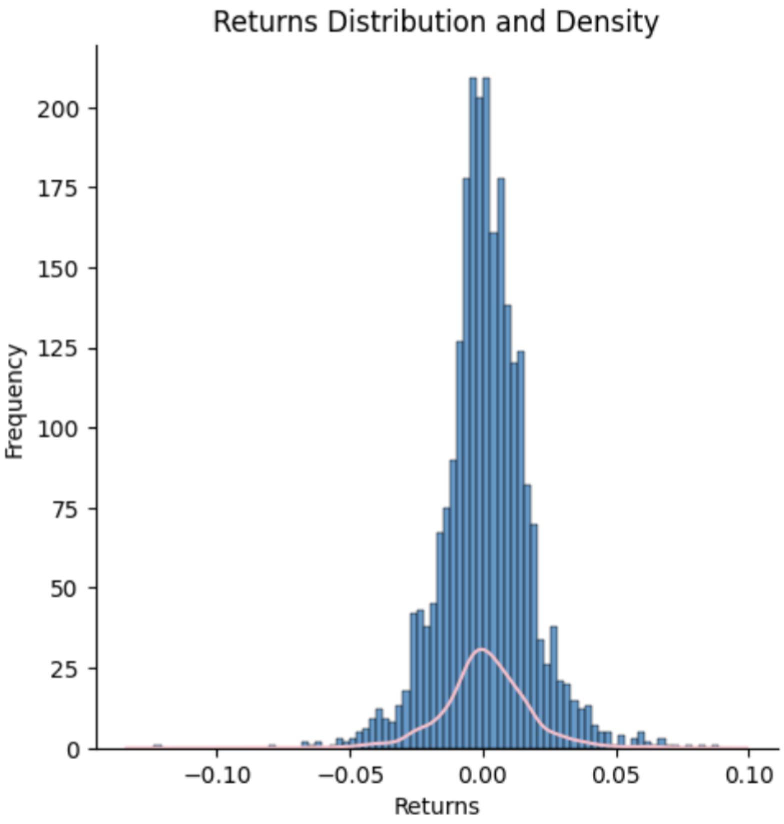


Figure 6: distribution and density graph of the return of Apple stocks

The remaining graphs and diagrams, which include screenshots of the other datasets can be seen in Appendix A.

3.2 Algorithms Description

Popular Python libraries used in machine learning and data analysis include time, Statsmodels, Sklearn, and Keras. These libraries were applied in the implementation of this project. Sklearn is a large machine learning package (short for Scikit-learn) offers a variety of supervised and unsupervised learning methods, such as classification, regression, clustering, and dimensionality reduction. Various pre-processing techniques, feature selection techniques, and model evaluation tools are also included. Additionally, Statsmodels is a library that offers classes and functions for performing statistical tests, investigating data, and running regressions, in addition to estimating a variety of statistical models. It contains techniques for regression analysis and time series analysis. Another high-level neural network API called Keras may be used for constructing, training, and assessing deep learning models. Python's time module offers several time-related methods. It is frequently used to sleep the execution of a programme for a predetermined period and to measure the time it takes for code to execute.

3.2.1 Machine Learning Models

3.2.1.1 Linear Regression

Regression analysis is used to determine both the magnitude and the direction of the relationship between the variables, to evaluate the dependant variable (y) based on the independent variable (x). Linear regression shows the linear relationship between the dependent and independent variables (GeekforGeeks, 2023). The most important feature in regression is that it is considered a supervised learning model. By using the patterns discovered from labelled data, it enables us to create predictive models that can make precise predictions on unobserved data. This is also why Linear Regression is a good choice for evaluating other models' performance (Altay & Satman, 2005).

The flow of the Linear Regression model used is as follows: selecting the feature intended for regression, splitting the feature into train and test sets, define and fitting model with train data, allowing model to make predictions with the test data, plotting a comparison graph, and evaluating model with MAE, RMSE, and MAPE.

3.2.1.2 Random Forest

Random forest is a form of ensemble learning technique that merges several decision trees to produce predictions. Ensemble machine learning algorithms merges various models to enhance the system's efficiency. Recursively, each decision tree is built by randomly choosing a subset of the input feature, which are then used to divide the data to smaller subsets. The final prediction is given by the median prediction made by all the trees. This is explained simply in figure 7. Random forest algorithm tackles the common problem of overfitting in decision trees and improves the precision of the predictions made by the model.

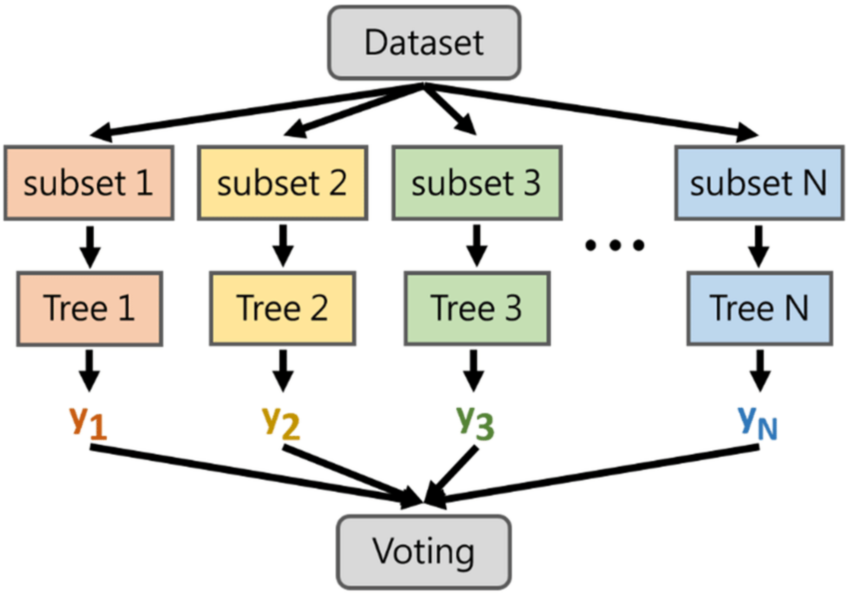


Figure 7: Random Forest flowchart (Lin, et al., 2020))

3.2.1.3 ARIMA

Mukherjee, et al., (2021) mentioned that autoregressive models are effective at forecasting the stock market because they provide insights into time series analysis and produce precise predictions. This is done by following multiple steps to define the parameters of the autoregressive model ARIMA. The first of these steps is to define the order of differencing (d), autoregression (AR), and moving average (MA) of the dataset. First, the data must be confirmed as nonstationary. The Augmented Dickey-Fuller (ADF) test is used to detect that. The data is considered stationary if the p-value produced is less than 0.05, otherwise the order of differencing (d) is measured. This is defined by the number of times the differencing is done to make the dataset stationary. Then, the MA and AR is evaluated. This is represented in the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots respectively (Shah, et al., 2022).

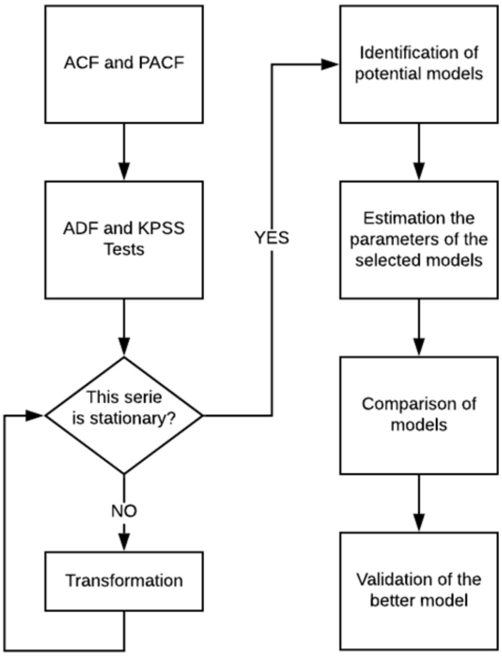

ACF and PACF
ADF and KPSS Tests
This serie is stationary?
NO
Transformation
YES
Identification of potential models
Estimation the parameters of the selected models
Comparison of models
Validation of the better model

Figure 8: ARIMA model flowchart (Moro, et al., 2020)

These steps, summarised in figure 8, define the parameters of the ARIMA model. After that, the model is initialised with the defined parameters and fitted with the training sets. Then, the forecast method is used for the model to make predictions on the test set. Finally, the model is evaluated with MAE, RMSE, MAPE, and the RMSE value of the Linear Regression applied to the results.

3.2.2 Deep Learning Models

Deep learning uses several nonlinear processing units in feedforward layers. Data processing, feature extraction, pattern learning, classification, and prediction are all carried out in different layers. Backpropagation is used by the layers to train the program's weights and parameters. After processing or training, the data is received by the input layer and forwarded to the following layer. Data is received by the input layer and produced by output layer (Althelaya, et al., n.d.). This applies to both LSTM and CNN.

3.2.2.1 LSTM

The primary motivation for utilising this model in stock market prediction is that the forecasts rely heavily on data and, in most cases, on the long-term performance of the market. It increases reliability of RNNs and accuracy of predictions by preserving data from earlier phases.

LSTM is a top tier RNN, prepared to handle the disappearing gradient problem observed in traditional RNNs (Rout, et al., 2022). It also improved learning long-term dependencies in RNNs. The hidden layer of the LSTM neural network uses memory cells instead of ordinary neurons as they are more effective in linking both input and memory. LSTM has the ability to memorise data sequences and store information from the past to the present. The sigmoid function is implemented to eliminate or modify the current state in memory, while pointwise multiplication and addition are employed for storing and updating the memory state (Shah, et al., 2022). There are three gates that control the flow of data as input data, saved data, and output data. These three gates are input gate, forget gate, and output gate. The input gate chooses fresh data to be stored in the cell, and the forgetting gate is used to erase or update the memory. The output gate then chooses the following hidden state (Shah, et al., 2022). Shah, et al. (2022) mentioned that LSTM provides lower risks than other methods, and that is possible due to the ability of LSTM in keeping track of the highest losses.

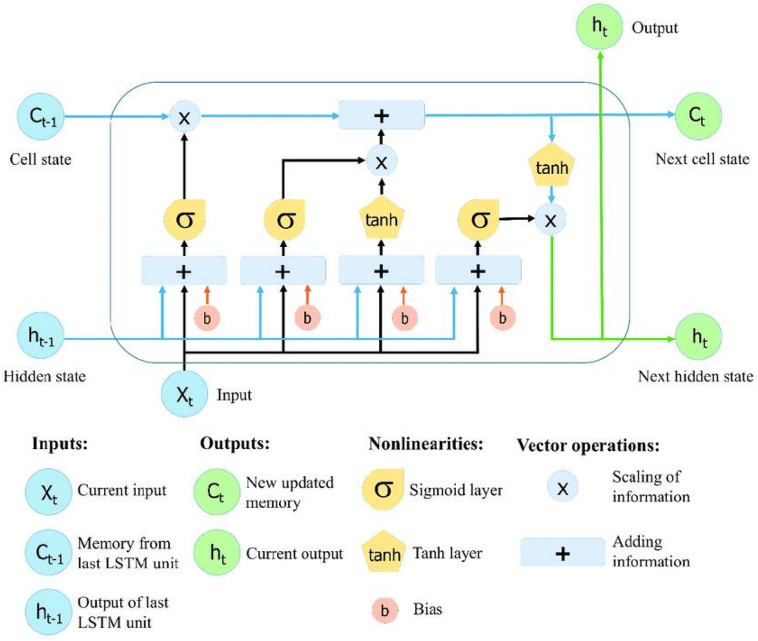


Figure 9: structure of LSTM (Le, et al., 2019)

The layers compiled for this model include one LSTM layer, and a dense layer. The LSTM layer oversees discovering the connections and patterns in the input data. Using the learnt patterns, the dense layer makes the final predictions accordingly. The number of epochs is 100, with the batch size set as 5. The optimiser used is Adam, with mean squared error (MSE) as the evaluation metric for the loss. Adam is an adaptive learning rate optimisation algorithm that combines RMSprop and Adagrad. These are the configurations of the LSTM models.

3.2.2.2 CNN

Convolutional neural networks, a subset of deep neural networks, are useful for time series forecasting. In this instance, features are extracted from the input tensor using a 1-dimensional CNN (Hamoudi & Elseifi, n.d.).

In terms of digital signal processing, 1D convolution on a time series is approximately equivalent to computing its moving average or applying a filter to the time series (Tam, 2021). To extract features, 1D CNNs combine 1D kernel with the input data (Rita, 2022).

The layers used for this model include two 1D convolutional layers, one MaxPool layer, a Flatten layer, and a dense layer. The output of the convolutional layers is down sampled using the MaxPool layer, which lowers the number of parameters and helps avoid overfitting. The output of the convolutional layers is transformed into a 1D array by the flatten layer and then passed on to the dense layer. The dense layer oversees making the final forecast on the features that were extracted. The epochs are set as 100 and the optimiser is specified as Adam, with mean squared error (MSE) as the loss calculation method. This is how CNN model is setup for the purpose of this research.

3.2.3 Decision Tree Classifier

Decision tree is a common algorithm for prediction. A separation of the data obtained by the implementation of straightforward principles is illustrated by an observable tree. Each observation assigned to a partition relies on the value of the single input. Due to their reasoning process, this method is used to find small or huge data structures and predict values since it is clear and easy to understand (Li, et al., 2017).

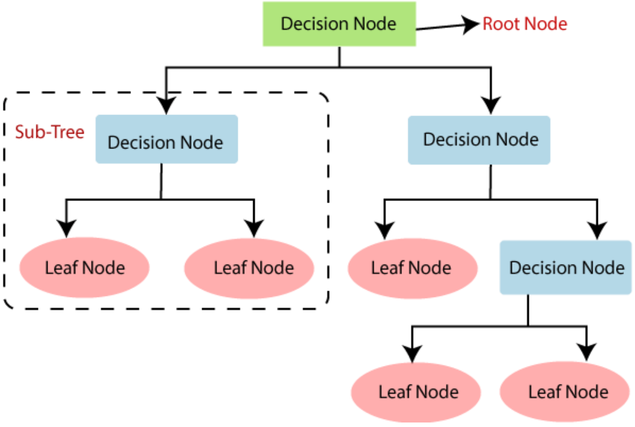


Figure 10: decision tree algorithm flowchart (Arian, n.d.)

As seen in figure 10, the decision tree works by producing sub-division nodes in the search for the best outcome. The best outcome is selected by the feature that best separates the data. Then the decision tree is evaluated with accuracy, precision, recall and F1 scores.

3.2.4 Evaluation Metrics

The calculations computed to evaluate the models used for this project is MAE, RMSE, MAPE. These measures calculate the accuracy of the models by determining the difference between the actual and predicted values of the test datasets using different methods. MAE computes the average of the total of absolute differences between the actual and predicted values. The value of the MAE is calculated by:

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

where n is the total number of observations, y_i is the actual value, and $\hat{y}_i$ is the predicted value of the test dataset (Saxen, 2021).

Furthermore, RMSE determines the degree to which a regression line accurately describes the data points. This is calculated by measuring the square root of the mean squared difference between the actual and predicted values. The equation for the RMSE value is:

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

(2)

where n is the total number of observations, y_i is the actual value, and $\hat{y}_i$ is the predicted value of the test dataset (Zach, 2021).

On the other hand, MAPE calculate the absolute measure of the difference between actual and predicted values and produces the mean of that as a percentage. This makes the measurement relevant and easy to understand by non-professionals. The equation for the MAPE value is:

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

(3)

where n is the total number of observations, A_t is the actual value, and $\hat{y}_i$ is the predicted value of the test dataset (apathak092, 2021).

3.3 Implementations

In the implementation of the algorithms mentioned above, there are several steps that are applied to the dataframe before directing it to the ML models. First, the input feature to be used for prediction is selected. Feature selection enhances the performance of ML models by lowering the cost of computation and resolving the overfitting issue by eliminating insignificant variables (Htun, et al., 2023). The chosen feature is the Close column in the dataframe. This choice has been inspired by many published program applications available online. Next, the input feature is given to the function specified for the model in the form of a Pandas Series. This function performs some pre-processing steps to the data, then forwards the data to the ML model. These pre-processing steps includes sampling, splitting the data into training and testing datasets, and scaling.

Given a sequence length, the data is sampled. The sequence length, which is decided as 7, represents the number of previous days that the model should consider influential to the price of the day to be predicted. In other words, before making a prediction on the price of a single day, the model should consider the price of the 7 days prior to that day (Mukherjee, et al., 2021). The sequence length is not a fixed number, and can be chosen according to personal preference, if it can be backed with a logical justification. Nonetheless, the sampling produces two dataframes that are then used for splitting the dataset. The dataset is split into 80% train set and 20% test set, using the train_test_split method from the Sklearn library. The training set is used in fitting the model for learning. However, the test set is applied to the model after training to evaluate the performance of the model. By comparing the accuracy of the difference between actual test set values and the predictions made by the model on the same values, the performance is measured.

Then, the data is normalised, which refers to the rescaling of the input feature to ensure that all the values are in the range between 0 and 1 (Jiang, 2021). This is done by the MinMaxScaler method in the Sklearn Library. Although normalisation is a step of data pre-processing, the reason why this step is postponed to this stage is because normalization would affect the EDA results. Additionally, normalization is only important in some of the models; decision tree classification -for example- does not require normalization.

Furthermore, the model is defined, fitted with the training set, then used to predict the test set. After making predictions, both the train and test sets are rescaled backed into their original values for visualization. A comparison line graph is created for visualizing the original and predicted test set values. Finally, the model is assessed using the evaluation metrics MAE, RMSE, and MAPE. For models other than Linear Regression, Linear Regression is used as an additional step in evaluation.

The technique of obtaining input features from unprocessed information based on required specifications is known as feature engineering (Jiang, 2021). This is applied in this project for the classification problem. A decision column is defined based on the values of the open and close prices of the stock. This is utilised by the Decision Tree Classifier to learn and give predictive decisions to advice investors.

The code implemented is optimised by separating the functions of each sub step in different python files that are then imported to a single Jupyter notebook. Separating functions into distinct Python files helps improve the code's modularity and organisation while also making it simpler to optimise. Hence, efficient time usage and the code's effectiveness can be increased.

3.4 Hardware and Running Times

The hardware used for this project is a Dell Inspiron 13 5310 laptop with a 64-bit operating system and an x64-based processor. It is equipped with an 11th Gen Intel® Core™ i5 processor that has four cores and eight logical processors. The laptop has 8 GB of RAM and a virtual memory of 21.7 GB. The operating system used is Microsoft Windows 11. Overall, this hardware offers appropriate computing power and memory for the data analysis and machine learning tasks of this project.

For training and operating large models, especially ones with a lot of parameters or data, a small RAM may not be enough. This would affect the speed of the training negatively. Additionally, having more cores can shorten model inference and training durations. Furthermore, the number and difficulty of models that can be trained or operated will be impacted if the system just uses the integrated graphics card. The size of datasets and models that can be saved or analysed can also be constrained by the quantity of accessible disc space. These effects can be seen in the duration of the training of the models. This is applied in this project by comparing the running time of each model, and the results are listed in table 2.

Table 2: running time comparison

DATASET	MODEL	TIME TAKEN FOR TRAINING MODEL (SECONDS)
APPLE	Linear Regression	0
	Random Forest	4
	ARIMA	1
	LSTM	304
	CNN	23
ALPHABET	Linear Regression	0
	Random Forest	4
	ARIMA	1
	LSTM	804
	CNN	29
MICROSOFT	Linear Regression	0
	Random Forest	4
	ARIMA	1
	LSTM	297
	CNN	24

Table 2 presents the amount of time taken for training each model on each of the three datasets. The duration is measured in seconds. For the three datasets, the Linear Regression model took zero seconds to train, while the Random Forest model took 4 seconds, and the ARIMA model took 1 second. The difference occurred in LSTM and CNN models. For the Apple dataset, LSTM took 304 seconds, and CNN model took 23 seconds. Moreover, for the Alphabet dataset, LSTM took 804 seconds to train, and CNN model took 29 seconds. Lastly, for the Microsoft dataset, LSTM model took 297 seconds to train, and CNN model took 24 seconds. Overall, the length of time required to train the models varied based on the dataset and model type. In all datasets, the LSTM models required the most time to train and the quickest to train was the Linear Regression model.

These results can be justified by multiple reasons, including the size of the dataset, the complexity of the model, and the computational resources available. Nevertheless, the long duration of training can be considered a trade-off between the training time and model performance. This means that the more complex a model is, the longer it takes to train. As a result, the model that take longer to train produce better predictions than the models that are trained in no time (Kumar, 2022).

4 Results

Following implementation of the listed models in the previous section, the results are given accordingly. The study's findings, which compared the effectiveness of various machine learning models at forecasting stock market values, are presented in this section. This section offers a thorough summary of the conclusions drawn from the examination and interpretation of the study's data. It contains the outcomes of the evaluation criteria applied as well as the performance of each model.

4.1 Linear Regression

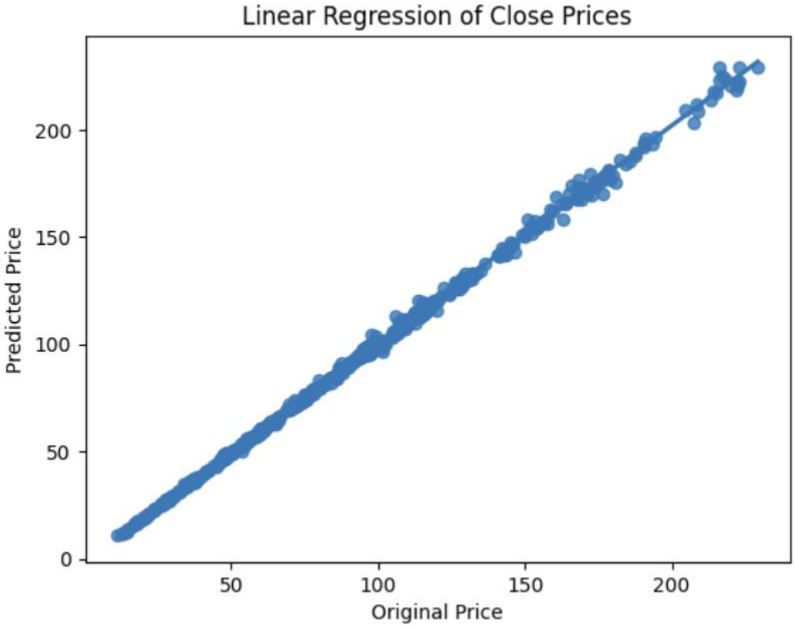


Figure 11: visualising the Linear Regression on Apple Stocks

Figure 11 shows a regression plot of the Apple stocks. The graph shows a positive correlation between the original and predicted price of the Linear Regression model. The scatter points on the graph are very close to the linear, best-fit line. Consequently, the fairly linear arrangement of the points in this specific plot shows that the Linear Regression model is able to capture the broad trend of the stock price movement. This shows that the model is reasonably accurate at predicting the stock price, yet further assessment must be considered in order to evaluate its accuracy.

4.2 Random Forest

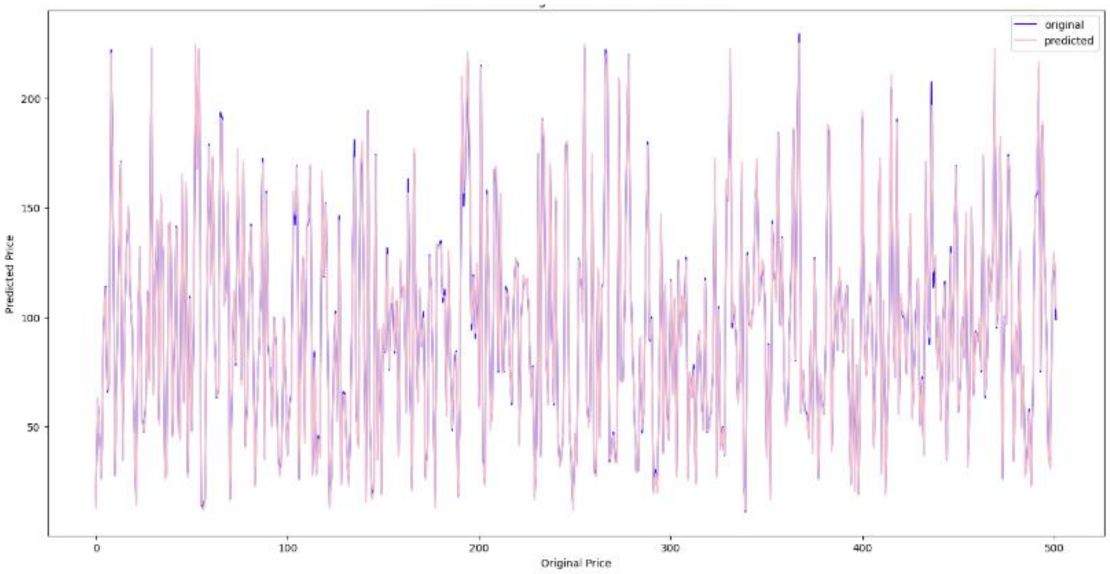


Figure 12: visualising the actual and predicted results of Random Forest on Apple Stocks

As seen in figure 12, the original and predicted values of the Random Forest are plotted in a line graph. These values concern the input feature, close price. Majorly, the original and predicted values are close, but this cannot be determined definitely just by looking at the graph. The original and predicted values are all from the testing set.

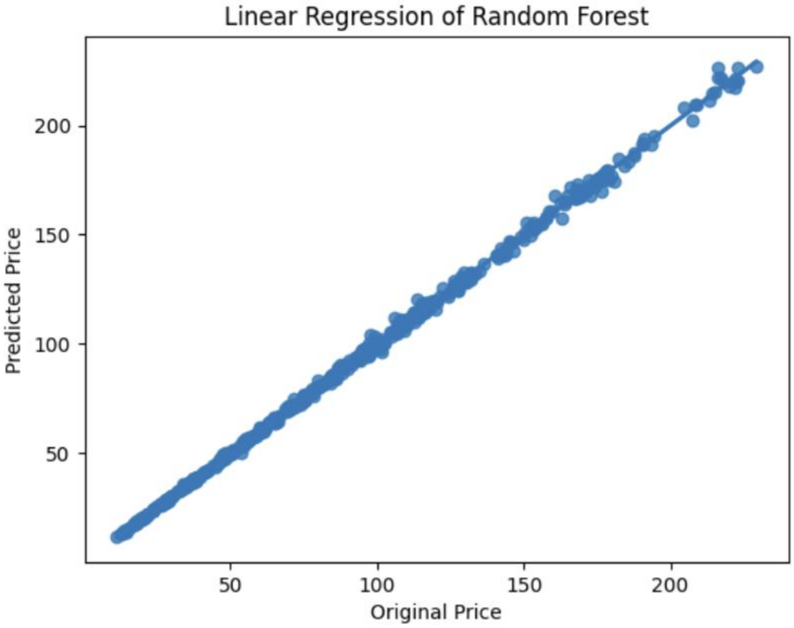


Figure 13: Linear Regression of Random Forest on Apple stocks

Figure 13 shows a regression plot of the Random Forest results. The line of best-fit shows a positive correlation of the original and predicted values. A line that roughly matches the trend of the data with a positive slope indicating a direct correlation between the actual and predicted variables. As a result, it is assumed that the ensemble learning model can accurately predict the target variable based on the input features, demonstrating that it has comprehended the underlying patterns and relationships in the data.

4.3 ARIMA

For the ARIMA model, there are multiple graphs that are plotted prior to training the model. These graphs show required information for the parameters of the model. Differencing plot, ACF and PACF are shown in figure 14 and figure 15 respectively.

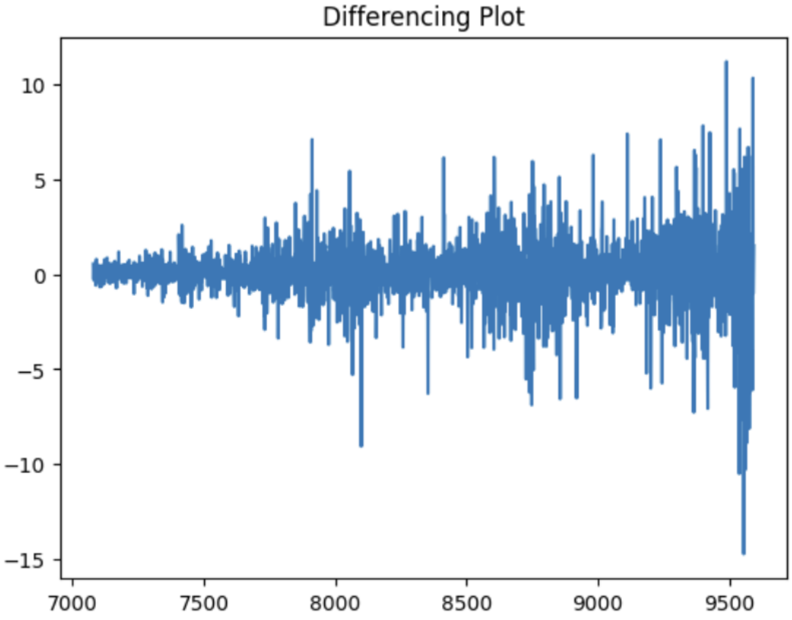


Figure 14: plot showing differencing

A differencing plot represents the difference between two successive observations in a time series. In simpler words, the difference between one value and the consecutive value is calculated and plotted. This plot is to show the order of differencing, which is a parameter for the ARIMA model, required to make time-series data stationary. The differencing plot for the close price values are represented in figure 14.

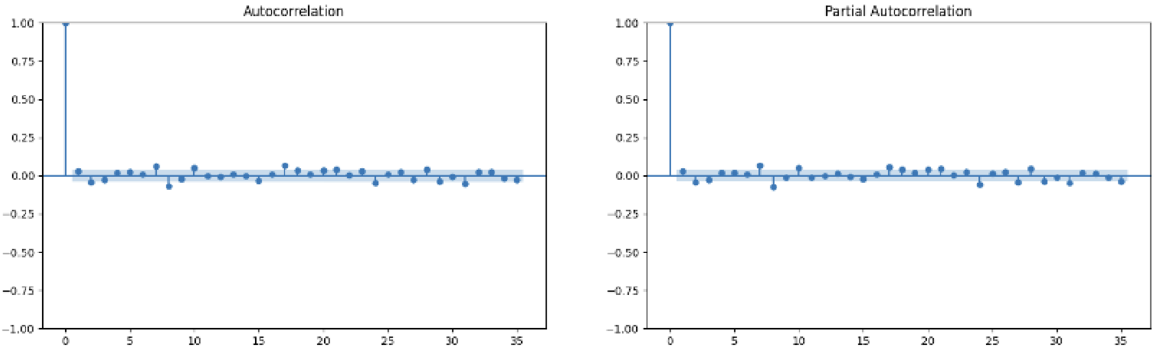


Figure 15: ACF and PACF plots on Apple stocks

Adding on this, ACF and PCF plots show the AR and MA values that define 2 of the parameters for ARIMA model. The moving average defines the trend for a stock. This is represented in the PACF graph on the right side of figure 15. On the other hand, the AR value describes the dependencies between each value and the value before it. This is displayed in the ACF plot on the left side of figure 15.

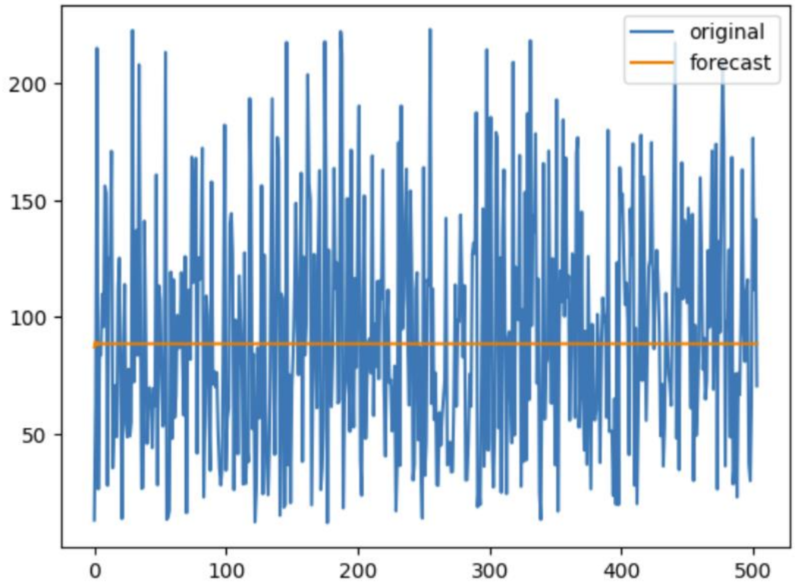


Figure 16: visualising the actual and forecast results of ARIMA on Apple Stocks

Figure 16 shows the actual and predicted values of the test set produced by the ARIMA model. The original values are shown as high volatile, while the forecasts are given in a straight line. This is irregular and indicates that an error occurred at some point. This will be later discussed with the results of the model evaluation in the Discussion section.

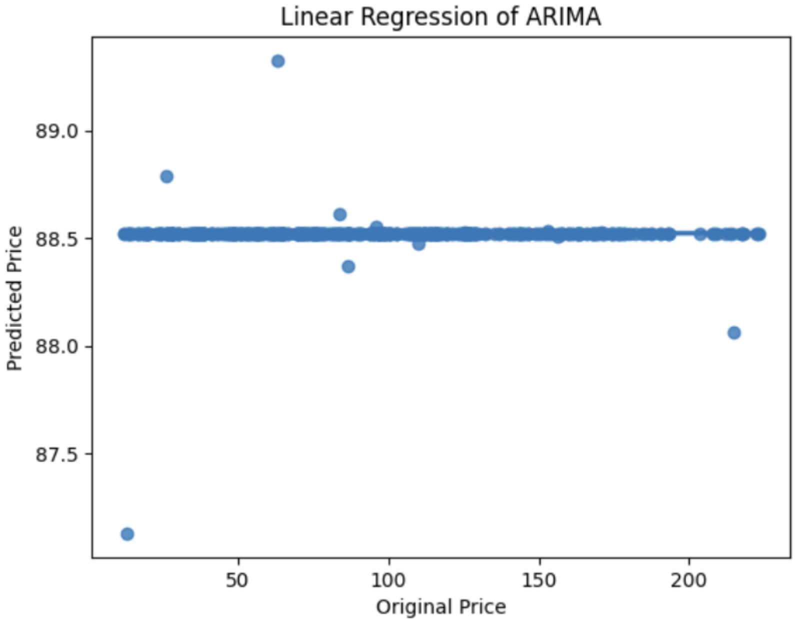


Figure 17: Linear Regression of ARIMA on Apple stocks

When adding the original and forecast values of the ARIMA model to a Linear Regression model, the results are represented in figure 17. The graph shows no correlation between the original and predicted values. This is shown by the vertical line of the values.

4.4 LSTM

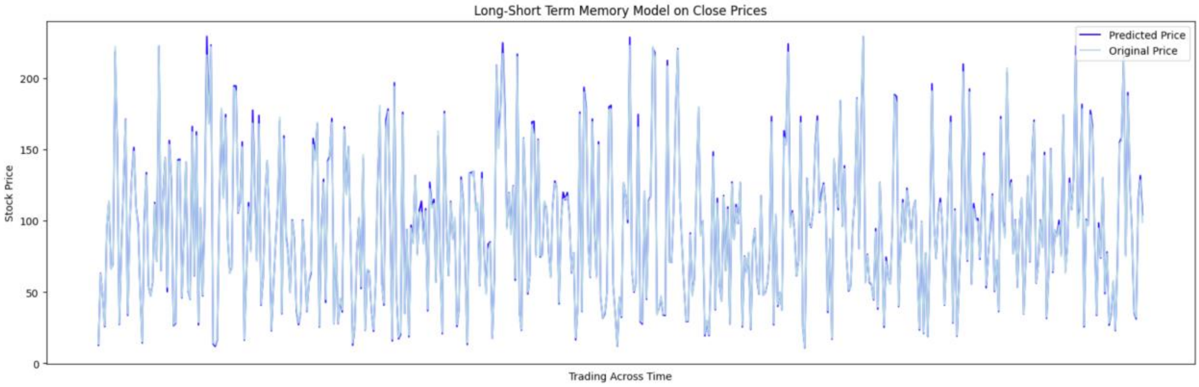


Figure 18: actual vs predicted results of LSTM on Apple stocks

Figure 18 shows the actual and predicted values of the test set produced by the LSTM model. These values concern the input feature, close price. Majorly, the original and predicted values are close, but this cannot be determined just by looking at the graph. The original and predicted values are all from the testing set. The two values are generally close, but it is impossible to say for sure from just glancing at the graph.

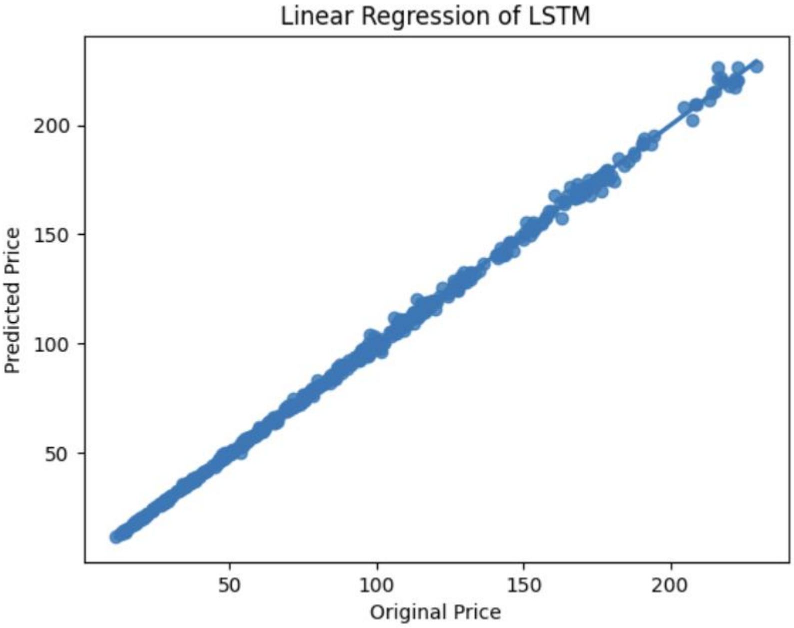


Figure 19: Linear Regression of LSTM on Apple stocks

A regression plot of the outcomes from LSTM is shown in figure 19. The original and forecasted values have a positive correlation, as shown by the line of best fit. A line with a positive slope that closely follows the data's trend and shows a direct link between the original and predicted values.

4.5 CNN

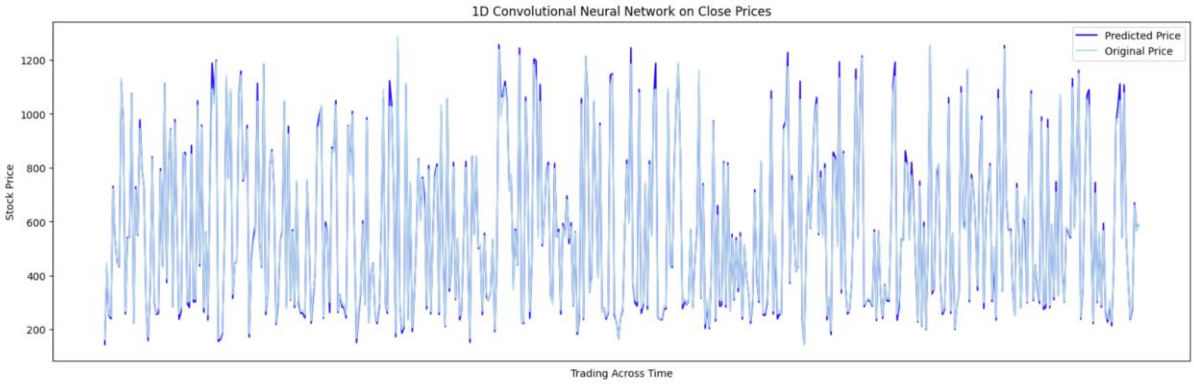


Figure 20: actual vs predicted results of CNN on Apple stocks

The test set's actual and predicted values, as computed by the CNN model, are displayed in figure 20. The original and prediction values are generally close, but it is impossible to say for sure from just glancing at the graph.

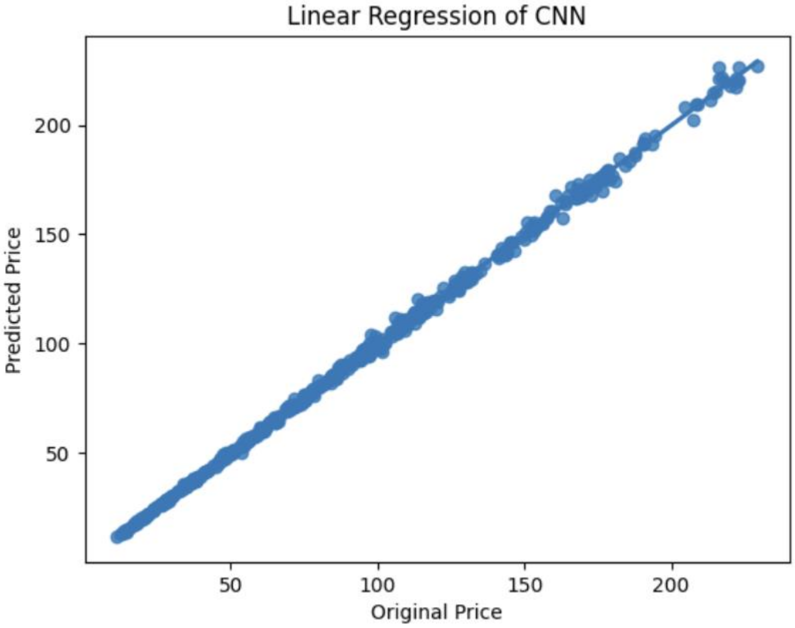


Figure 21: Linear Regression of CNN

Figure 21 displays a regression plot of the results from the CNN. The line of best fit demonstrates a positive correlation between the original and predicted values. The positive slope of the line, closely resembling the data's trend, demonstrates a clear connection between the original and predicted values.

The graphs of the results for the other datasets, Alphabet and Microsoft, are given in Appendix B. These graphs show a similar output to those presented in this section. However, it is not exactly the same, as the values in the datasets are different. Also, ML models do not perform the same for every iteration. This is because ML models learn patterns in the data, which are rough estimates that are not entirely accurate.

4.6 Module Evaluation

When applied to the three chose datasets, the evaluation of the models used for stock price prediction are displayed in the table 3. These results include the MAE, RMSE and MAPE of each model, and the RMSE of the Linear Regression applied to the predictions of the models.

Table 3: model evaluation (rounded to 2 decimal places)

DATASET	MODEL	EVALUATION METRIC			RMSE OF LINEAR REGRESSION
		MAE	RMSE	MAPE (%)	
APPLE	Linear Regression	1.33	1.97	1.71	-
	Random Forest	1.4	2.06	1.85	1.68
	ARIMA	41.4	50.41	74.36	88.52
	LSTM	1.43	2.14	1.72	1.68
	CNN	1.9	2.99	2.05	1.68
ALPHABET	Linear Regression	13.89	16.15	3.11	-
	Random Forest	13.51	16.41	3.01	10.24
	ARIMA	248.52	295.36	58.07	546.68
	LSTM	16.65	19.23	3.45	10.24
	CNN	13.33	19.42	2.65	10.24
MICROSOFT	Linear Regression	0.56	0.88	1.23	-
	Random Forest	0.63	0.95	1.43	0.85
	ARIMA	18.88	24.17	45.1	44.33
	LSTM	0.63	0.92	1.52	0.85
	CNN	0.91	1.35	2	0.85

For the Apple stocks dataset, the lowest MAE, RMSE, and MAPE values are all given by the Linear Regression. These values are 1.33, 1.97, and 1.71. On the other hand, the ARIMA model gave the highest MAE, RMSE, and MAPE with values 41.4, 50.41, 74.36. Other models showed values ranging from 1 to 3, for all evaluation metrics.

For the Alphabet dataset, the lowest MAE value is given by the CNN model, which is 13.51. Additionally, Linear Regression gave the lowest RMSE value, as 16.15. The lowest MAPE value, 2.65, is CNN too. The highest MAE, RMSE, and MAPE are 248.52, 295.36, 58.07 respectively, given by the ARIMA model. Alternative models gave results between 2 and 19.

For the Microsoft dataset, the lowest MAE, RMSE, and MAPE value given by the Linear Regression model. The results are 0.56, 0.88, and 1.23 respectively. On the other hand, the results of the ARIMA model were the highest for the MAE, RMSE and MAPE. These values are 18.88, 24.17, and 45.1 respectively.

Moreover, the Linear Regression produced a distinct RMSE values for the three datasets. The RMSE value of the models in the Apple dataset was 1.68, except the ARIMA model. The ARIMA model produced a value of 88.52. In addition, the RMSE value of the models in the Alphabet dataset was 10.24, but for ARIMA, 546.68. Finally, the RMSE value of the models in the Microsoft dataset was 0.85, excluding ARIMA model, which gave a result of 44.33.

In conclusion, the study of various models on the Apple, Alphabet, and Microsoft datasets revealed that the Linear Regression model mostly outperformed the others for all three datasets in terms of MAE, RMSE, and MAPE. Opposingly, the ARIMA model delivered the lowest results in all evaluation measures. The LSTM and CNN models yielded inconsistent results; however, the Random Forest model often gave outcomes similar to those of the Linear Regression model.

4.7 Decision Tree Classifier

The Decision Tree Classifier was set to predict results on buy, sell, or neutral decisions. The predictions are portrayed in a confusion matrix with four possible outcomes: true positive (TP), true negative (TN), false positive (FP), and false negative (FN).

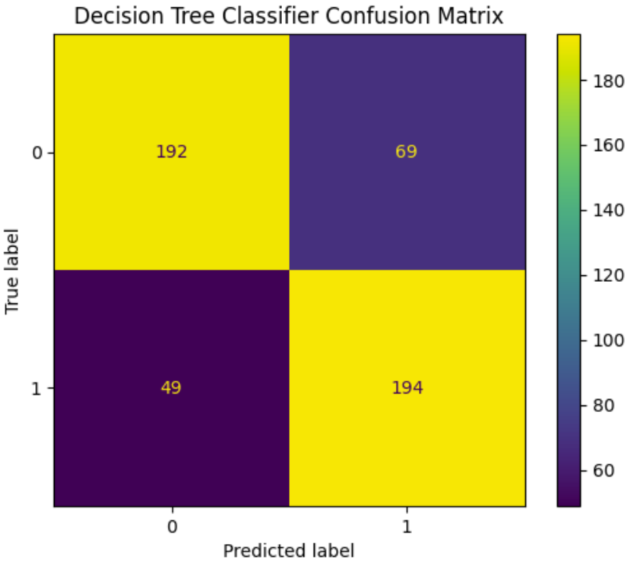


Figure 22: confusion matrix of decision tree classifier results (Apple dataset)

In the 2x2 matrix, it can be seen that the model correctly predicted a buy decision 192 times, and correctly predicted a sell decision 194 times, which are the true positives and true negatives, respectively. However, the model incorrectly predicted a buy decision 69 times when it should have been a sell decision, which are the false positives. Also, the model incorrectly predicted a sell decision 49 times when it should have been a buy decision, which are the false negatives (figure 22).

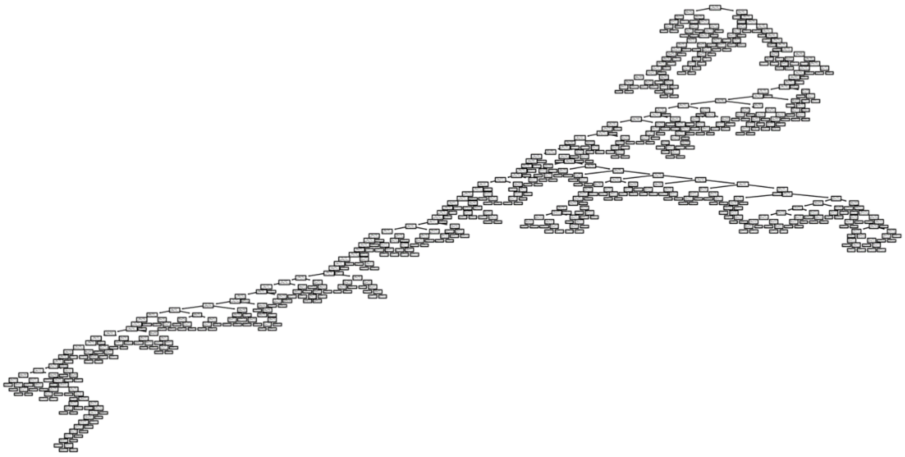


Figure 23: decision tree visualisation (Apple dataset)

Figure 23 shows a graphical representation of the decision tree model used in this project. It displays the sequence of steps for the decision tree model to make a prediction. In order to maximise the separation between the classes, the tree is built by recursively dividing the data into smaller groups based on the values of the input features. Each internal node of the tree represents a decision, and each leaf node represents a predicted class.

Table 4: decision tree classifier model evaluation (rounded to 2 decimal places)

DATASET	EVALUATION METRIC (%)			
	Accuracy Score	Precision Score	Recall Score	F1 Score
APPLE	76.59	76.72	76.7	76.59
ALPHABET	79.76	79.59	79.59	79.59
MICROSOFT	78.57	53.56	52.67	53.11

Table 4 shows the evaluation metrics for the three datasets: Apple, Alphabet, and Microsoft, for the classification model. The evaluation metrics used to measure the performance of the model are accuracy score, precision score, recall score, and F1 score, represented in percentages.

For the Apple dataset, the model achieved an accuracy score of 76.59%, indicating that the model's predictions were correct 76.59% of the time. The precision score and recall score were almost identical at 76.72% and 76.7%, respectively, indicating that the model was equally good at identifying true positives and avoiding false positives. The F1 score was also 76.59%.

For the Alphabet dataset, the model performed slightly better than the Apple dataset, achieving an accuracy score of 79.76%. The precision score, recall score, and F1 score were the same at 79.59%.

For the Microsoft dataset, achieve an accuracy score of 78.57%. However, the precision score and recall score were considerably lower, indicating that the model had a higher rate of false positives and false negatives. The F1 score was 53.11%, which was also lower than the other two datasets.

The results of the classification show that the Decision Tree provided consistent accuracy scores in all datasets. This also applies to the results of the precision test, recall test, and F1 test in the first two datasets. Meanwhile, Microsoft dataset has lower precision, recall and F1 scores.

5 Discussion and Analysis

5.1 Significance of the findings

Before deciding on the ideal model for a given task, it is crucial to assess several metrics and apply the chosen models to multiple datasets. This is because the dataset and evaluation metrics employed can have an impact on the models' performance. For example, the models perform well for the Apple and Microsoft dataset, but not as much for Alphabet. To get reliable results, the proper model must be chosen for a certain dataset.

A lower score on the evaluation metrics means better performance than higher scores. Based on the results of the evaluation metrics, there are certain assumptions that can be concluded. First, Linear Regression performs well across all three datasets, given that it consistently generates the lowest MAE, RMSE, and MAPE values of all the evaluated models. This means that the model showed good performance for predicting stock market data. Thus, Linear Regression is considered a good model for stock market prediction.

With a little variation in some circumstances, Random Forest model performs nearly as well as the Linear Regression model. The low values produced in the evaluation metrics supports the proven theory of Htun, et al. (2023). Their theory states that Random Forest produces satisfactory outcomes. The evaluation's overall findings are consistent with those of Htun et al. (2023) and imply that the Random Forest model can be a reliable substitute for the Linear Regression model with comparable performance.

Both LSTM and CNN models performed well for the three datasets. The MAE, RMSE, and MAPE values for LSTM range from 0.63 to 2.14 in the Apple and Microsoft datasets. However, in the Alphabet dataset, the MAE and RMSE showed extreme values of 16.65 and 19.23 respectively.

Based on research done by Rout, et al. (2017), the LSTM model showed low RMSE scores for the Apple, Alphabet and Microsoft datasets among other datasets. Respectively, the RMSE value for these datasets retrieved from this research are 0.828, 0.775, and 0.778. In comparison, the results of the MSE values for the same datasets in this study is 2.14, 19.42, 0.92. Different data pre-processing steps, model architectures, hyperparameters, and randomness in model training all contribute to justify the difference in the results given in this research in comparison to the results of the produced work of Rout, et al. The number of layers, for example, which are part of the model architecture, contribute to the RMSE values of the model. In conclusion, the results of the research showed better performance than the RMSE values of this study. This means that the LSTM model in this investigation could be defined to produce better results.

Meanwhile, the results of the CNN model show similar values to the LSTM scores. The MAE, RMSE, and MAPE values produced range from 0.91 to 2.99 in the Apple and Microsoft datasets. The Alphabet dataset shows exceptional values of MAE of 13.33, and RMSE of 19.42. Overall, for the Apple and Microsoft datasets, both LSTM and CNN models showed strong performance. Both models, however, showed extremely high MAE and RMSE values for the Alphabet dataset. To ascertain the cause of this inconsistency more research is needed.

Nonetheless, LSTM operated better than CNN in two of the three datasets. This is supported with the research done by (Vargas, et al., 2017), which suggest that RNN has proven more effective than CNN, and as LSTM is considered a type of RNN, this is applicable to this case too. In the mentioned research, the B-LSTM model showed better performance, in terms of RMSE values, in most of the datasets.

The LSTM model also performed better than the ARIMA model. This is relevant in the research done by Shah, et al. (2022), where the ARIMA model showed higher RMSE values than LSTM. Although, considering the extreme values of the ARIMA model, the results might not be reliable, and more research must be done to make sure of this fact.

Unlike the work of Parmer, et al. (2018), LSTM did not show better performance than Linear Regression. However, the results were very close. For example, the MSE scores for Linear Regression in the three datasets are 1.33, 13.89, and 0.56. Meanwhile, the MSE scores of the LSTM in the three datasets are 1.43, 16.65, and 0.63. The difference in two of the datasets does not exceed 0.1. As a result, both models can be considered equally effective in this investigation.

At first glance, it may seem that the ARIMA model is not appropriate for forecasting stock prices in the provided datasets. This may be assumed from the high MAE, RMSE, and MAPE values. The ARIMA model performed poorly compared to other models, especially for the Alphabet dataset, where it produced extremely high error values for the evaluation metrics. This can also be explained by many factors, starting with the parameters of the model. Selecting incorrect hyperparameters, such as learning rate, batch size, and number of layers, for a model influences the performance of the model poorly. For example, poor performance could result from the model being unable to reach a conclusion if the learning rate is too high. In addition, the model might not effectively identify the underlying patterns in the data if the batch size is too small. The model may overfit or underfit the data, respectively, if the number of layers is too high or too low. Therefore, to achieve optimal model performance, it is essential to make sure that proper hyperparameters are chosen for each model.

Other reasons of extreme error values in evaluation scores include missing or noisy data, overfitting or underfitting of model, incorrect feature selection, or model choice. These all lead to poor performance of the model. However, to hold a strong opinion about the actual reason for the model's inadequate efficiency, the results must be considered as a whole. The results in general show that all models perform either very well, or decently, on forecasting stock market data, excluding the ARIMA model and the performance of the models in the Alphabet dataset.

The first of the reasons for the occurrence of extreme values in a certain dataset is the quality of the data. There might be unhandled outliers influencing the model's performance. These outliers may distort the findings and increase error metrics. Alternatively, the dataset might be asymmetrical, which means that the number of observations for various classes differ significantly. Particularly for models vulnerable to class imbalance, this can result in biased performance measures. These models include Linear Regression and DL models. Other reasons that make producing precise predictions with the model difficult may include noisy data and high variability. These reasons all affect the trend and observation of fluctuations in the dataset, which would lead to incorrect predictions. Secondly, the performance of the model and the evaluation that follows are affected by the hyperparameters, choice of learning models, and evaluation metrics used. This means that the results would change with different model configurations.

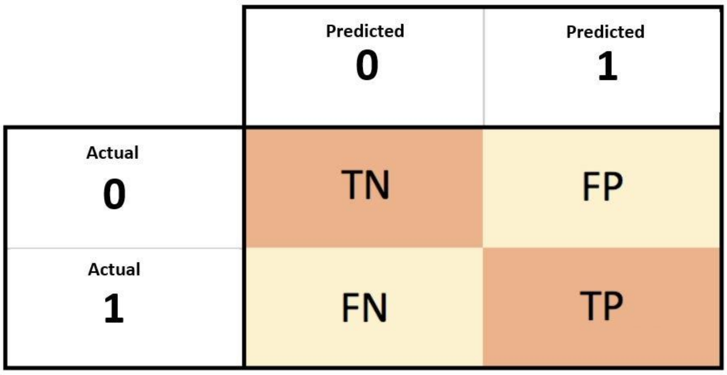
The other part of the evaluation considers the RMSE value of the Linear Regression. The models produced an identical value in each dataset. According to the Linear Regression results, all the models performed better in the Microsoft dataset, with the lowest RMSE value as 0.85 for all models except the ARIMA model which has and RMSE value of 44.33. The highest RMSE values were produced by the models in the Alphabet dataset, with a value of 546.68 for the ARIMA model and 10.24 for all the other models.

Understanding that a study's findings depend on the datasets and model configurations used is crucial. It is possible that the findings of one study do not apply to another with different data and model specifications. This is because a model which excels on one dataset does not necessarily work well on another dataset with other characteristics.

When considering the time taken for training the models, most of the models trained in a short time. The Linear Regression took no time to train at all, while the ARIMA model took 1 second to train. These are the quickest models trained from the five models, with the same results for all the datasets. Random Forest took 4 seconds to train, which is also the same for the three datasets. Only the LSTM an CNN showed a different timing on the three datasets, with LSTM giving a time range from 297 to 804 seconds, and CNN giving a range of 23 to 29 seconds.

Using this information, the proportionality between the training time and the performance of each model is determined. Linear Regression showed the best testing evaluation scores with the shortest training time. This is an additional indicator that Linear Regression is a good option for stock market prediction. Opposingly, ARIMA model showed bad performance, with a short training time. This shows that a short training duration does not necessarily mean that the model is good. Even though the training period was short, it is possible that the model was not sufficiently sophisticated to identify the underlying patterns in the data, which would explain the poor performance. Nonetheless, with the decent performance shown by both Random Forest and CNN, the duration taken for training is considered good. A short time and a good performance are desirable; hence these models are considered an effective option for stock market prediction. Finally, the LSTM showed a good performance but took the longest time to train, with a value of 804 seconds. Theoretically, the results are not very bad given that the LSTM model performed well, but the difference between the training time of LSTM and other models is not a good sign. The long training time justifies the complexity of the LSTM model. Practically, 13 minutes (804 seconds) is a very long duration as it is not efficient for a user to wait 13 minutes to get a prediction on a certain stock. In conclusion, considering the duration of training the models, Linear Regression, Random Forest, and CNN are recommended as reasonable solutions for stock market prediction.

Decision Tree Classification



	Predicted 0	Predicted 1
Actual 0	TN	FP
Actual 1	FN	TP

Figure 24: confusion matrix division clarification (Zuhaib, 2019)

Figure 24 shows a visual presentation of the division of the confusion matrix blocks. With reference to the mentioned figure, the true predictions of the Decision Tree Classifier are 194 and 192 for Sell and Buy decisions respectively. This means that the decision tree predicted 386 values out of 504 correctly. On the other hand, 118 values of the predictions were incorrect, with a tendency for the model to predicted false buy decisions. This shows a decent performance for the model. With more true predictions than false predictions, the model seems to be performing well. However, more research would be required to fully assess the model's performance.

The evaluation metrics used to assess the performance of the Decision Tree Classifier are Accuracy Score, Precision Score, Recall Score, and F1 Score. Precision score evaluates how accurately the model made predictions that turned out to be true. Meanwhile, the model's ability to accurately recognise each positive value is indicated by the recall score. F1 score is frequently used as a summary statistic to assess the effectiveness of a model. It offers a single score that finds a compromise between precision and recall scores.

There are a few implications that can be drawn from the evaluation metrics' results. By looking at the accuracy score, the model performed well on all datasets. This means that the model is able to correctly predict a buy/sell decision. The precision, recall, and F1 scores are as good as the accuracy scores. However, there is an exceptional case for the Microsoft dataset, where the scores are lower by around 25%. Compared to the other two datasets, the model is less effective at identifying positive instances for the Microsoft dataset. Several factors, including class imbalance, noisy data, or insufficient trends in the target variable may have affected the performance of the model. Recognising the source of the poor performance and taking the necessary steps would help the model perform better.

Buy and sell decisions are sensitive since they can lead to loss of valuable funds for investment. This is concerning, as it could lead to distrust of investors in prediction models, and this is not wanted. It is understandable that the results do not show sufficient, reliable decision (buy/sell). This is because stock trading decisions rely on more than just the previous day's given closing price. With this said, the algorithm shows a foundation of what could hold a great potential in making such decisions for aiding traders.

5.2 Limitations

There are limitations to the research done in assessing the performance of ML models in predicting stock market prices. These include the data type used, models' setup, influencers of stock market prices. First off, market prices might not always reflect basic factors. This implies that the stock's current market price might not accurately represent its underlying value (Seth, 2022). Stock prices are complicated and influenced by many factors. This may take into account elements like revenue growth, profits, and other financial metrics. Market data is straightforward and does not reflect all these factors. Which is why it is essential to study the impact of the other factors on the market data, and thus the stock prices.

These other data types include text data, macroeconomics, and analytics data. According to Bhuriya, et al. (2017), financial news, which is a type of text data, can have a considerable impact on stock market predictions. This effect isn't always reliable, and financial news has been shown to occasionally cause market disruptions and inaccurate projections (Shah et al., 2022). The contrasting points show that the data types must be weighted and evaluated before using them to predict stock prices. Nonetheless, using more than one type is more effective than one type (Jiang, 2021).

Secondly, the results of this research are highly impacted by the set-up configurations of the models. This means that using different parameters would lead to different results. The bigger the difference in the parameter values, the more the variance in the results. This is support by the research of Shah, et al. (2022), which suggests that the adjusting the number of hidden layers in the models is important for a reliable outcome. Additionally, the model's ability to project the time series of the stock does not always get better when the number of epochs is increased. This implies that making predictions is not always improved by more data.

Finally, Jiang (2021) discovered that hybrid models, which incorporate various prediction models, perform better than a single model for stock market predictions. taking this into consideration and using the best performing models to get predictions would improve the results. For example, LSTM and CNN models.

In general, these limitations emphasise the need for due diligence when evaluating the outcomes of stock market prediction algorithms. Due to the sensitivity of the topic, making financial decisions requires considering a variety of aspects.

5.3 Summary

After evaluating the results of the models' assessment, specific assumptions were made. These assumptions include that Linear Regression, Random Forest, LSTM and CNN are all good options for stock market prediction. Linear Regression, and LSTM are the best performing considering the results of the evaluation metrics. And Linear Regression, and Random Forest are the best in terms of training time. The CNN model falls in the middle, good enough in both evaluations. On the other hand, the ARIMA model performed badly, which can lead to the conclusion that it is not a good option for stock market prediction. However, further research is advisable to absolutely confirm these assumptions.

6 Conclusion and Future Work

6.1 Conclusions

Machine learning has become a powerful tool for predicting stock prices and making investment decisions. There are many different machine learning models available, and each has strengths and weaknesses. In this report, some commonly used models such as Linear Regression, Random Forest, ARIMA, LSTM, and CNN were reviewed. Additionally, the use of Decision Tree Classifier was tested and evaluated.

The steps followed in the project was dataset selection, data pre-processing, feature selection, model training and testing, evaluation of models. These steps were thoroughly explained and gone through to make sure the models show an ideal performance that can reflect good results. These results are then used for evaluation and comparison.

Based on the results of the evaluation metrics, Linear Regression is considered the best model for stock market prediction, followed closely by Random Forest. Both LSTM and CNN models also performed. However, the models showed poor performance in predicting stock prices for the Alphabet dataset. In contrast, the ARIMA model performed poorly in all three datasets.

The extreme error values observed in the evaluation metrics for some models and datasets could be attributed to several reasons. These include inadequate or inappropriate pre-processing techniques, limited feature selection, inadequate model architecture, poor choice of hyperparameters, and insufficient data. Other reasons could include the influence of external factors that could impact the stock market, such as economic or political factors, which the models may not have accounted for. It is crucial to identify these factors and improve the model accordingly to obtain better results.

Generally, predicting stock prices is a complex task that requires the use of appropriate models and techniques to achieve optimal performance. While different models may perform differently depending on the dataset and evaluation metrics employed. Linear Regression, Random Forest, LSTM, and CNN models have shown potential in predicting stock prices. To acquire the best results, it is crucial to make sure that the dataset, data pre-processing, assessment metrics, model architecture, and hyperparameters are selected carefully.

It is evident that machine learning has greatly improved the accuracy of stock market prediction, but there are still some challenges and gaps in current literature. Among these are the challenges associated with managing complex and high-dimensional data, the difficulty of interpreting some models, and the failure of predicting unforeseen market changes. Nonetheless, with continued advancements in technology and the availability of vast amounts of data, machine learning will likely continue to play an important role in the financial industry. Investors and financial institutions must be aware of the constraints and dangers posed by these models if they are to effectively employ them in conjunction with human expertise when making investment decisions.

6.2 Future work

With the continuous rapid growth of technology, and the increasing capacity of applying these technologies to real-life cases, the future work that can be done the sector of stock markets is various. Future work for research in stock market prediction using machine learning may include looking into the performance of more recent ML models like DL and reinforcement learning. These models may be more accurate in forecasting stock values because they have produced encouraging outcomes in other fields. Moreover, the creation of hybrid models that integrate the advantages of various machine learning methods. For instance, a hybrid model may combine the feature selection capabilities of Random Forests with the time-series forecasting capabilities of LSTM networks. Additionally, future research could be interesting in examining the effects of different economic indicators and news sentiment analysis on stock prices. These suggestions can improve the precision in the predictions, increasing the quality of the performance of ML models, and provide better results. Overall, machine learning's promising future in stock market forecasting offers considerable promise for the financial sector.

7 Reflection

Reflecting upon the work done in this project, there are many points worth mentioning that supported and impacted on the work produced. First of all, the scale of this project is quite challenging. Without expertise in the field of data science or python programming language, the project seemed almost impossible to accomplish. But tackling the tasks step by step helped me gain confidence in the continuation of this project. This is a great challenge for a final year student and helped me learn a lot. This is also a once-in-a-lifetime opportunity for developing skills professionally and personally, including research, writing, analysis, independence, initiative, and personal reflection skills.

This project aimed to show that ML shows immense potential in the stock market field. This is to study the scale of possibilities that are created with the integration of ML and AI in general into a challenging area such as the stock market. With the modern technologies found continuously, the ability of AI continues to grow, which is why this project can be improved in many ways to begin the application of its theories into real-life cases. Nonetheless, it is a foundation for interested parties to understand the basic concepts behind some of the ML models in the stock market.

If I had the opportunity to do this project all over again, I would focus more on two ML models; CNN and RNN. The aim of my project would be to exploit their capabilities and make the most of their features to get the best predictions for stock prices. This can be done by tuning the hyperparameters of the models to get the best results from each model. I would also direct my attention to the importance of feature selection, and the results that can be extracted from choosing a different feature at a time (open, low, high, close). I would also try to incorporate another type of data, mainly text data, maybe in the form of sentiment analysis. And find the further impacts of this analysis on the stock prices.

8 Bibliography

Altay, E. & Satman, M. H., 2005. STOCK MARKET FORECASTING: ARTIFICIAL NEURAL NETWORK AND LINEAR REGRESSION COMPARISON IN AN EMERGING MARKET. *Journal of Financial Management and Analysis,* 18(2), pp. 18-33.

Althelaya, K., El-Alfy, E.-S. & Mohammed, S., n.d. *Stock Market Forecast Using Multivariate Analysis with Bidirectional and Stacked (LSTM, GRU),* Dhahran: King Fahd University of Petroleum and Minerals.

apathak092, 2021. *How to Calculate MAPE in Python?.* [Online]
Available at: https://www.geeksforgeeks.org/how-to-calculate-mape-in-python/
[Accessed 1 April 2023].

Arian, F. N., n.d. *Decision Tree Classification Algorithm.* [Online]
Available at: https://www.devops.ae/decision-tree-classification-algorithm/
[Accessed 6 April 2023].

Banerejee, P., 2020. *Data pre-processing: A step-by-step guide.* [Online]
Available at: https://towardsdatascience.com/data-pre-processing-a-step-by-step-guide-541b083912b5
[Accessed 12 November 2022].

Bhuriya, D., Kaushal, G., Sharma, A. & Singh, U., 2017. *Stock Market Predication Using A Linear Regression,* s.l.: International Conference on Electronics, Communication and Aerospace Technology.

Gandhmal, D. & Kumar, K., 2019. Systematic analysis and review of stock market prediction techniques. *Computer Science Review,* Volume 34.

Ganti, A., 2020. *Adjusted Closing Price.* [Online]
Available at: https://www.investopedia.com/terms/a/adjusted_closing_price.asp
[Accessed December 2022].

GeekforGeeks, 2023. *ML | Linear Regression.* [Online]
Available at: https://www.geeksforgeeks.org/ml-linear-regression/
[Accessed 5 April 2023].

Hamoudi, H. & Elseifi, M., n.d. *Stock Market Prediction using CNN and LSTM,* s.l.: Stanford University.

Htun, H. H., Biehl, M. & Petkov, N., 2023. Survey of feature selection and extraction techniques for stock market prediction. *Financ Innov,* pp. 1-25.

IBM, n.d. *What is exploratory data analysis?.* [Online]
Available at: https://www.ibm.com/topics/exploratory-data-analysis
[Accessed 4 April 2023].

Jain, D., 2023. *Data Pre-processing in Data Mining.* [Online]
Available at: https://www.geeksforgeeks.org/data-preprocessing-in-data-mining/
[Accessed 3 April 2023].

Jiang, W., 2021. Applications of deep learning in stock market prediction: Recent progress. *Expert Systems with Applications,* Volume 184.

Kumar, A., 2022. *Model Complexity & Overfitting in Machine Learning.* [Online]
Available at: https://vitalflux.com/model-complexity-overfitting-in-machine-

learning/?utm_content=cmp-true
[Accessed 6 April 2023].

Lee, C.-Y. & Soo, V.-W., 2017. *Predict Stock Price with Financial News Based on Recurrent Convolutional Neural Networks.* Taipei, Conference on Technologies and Applications of Artificial Intelligence.

Le, X. H., Ho, H. V., Lee, G. & Jung, S., 2019. Application of Long Short-Term Memory (LSTM) Neural Network for Flood Forecasting. *Water,* 11(7).

Li, A., Wu, J. & Zhidong, L., 2017. Market Manipulation Detection Based on Classification Methods. *Procedia Computer Science,* Volume 122, pp. 788-795.

Lin, G.-W.et al., 2020. Towards Automatic Landslide-Quake Identification Using a Random Forest Classifier. *Applied Sciences,* 10(11).

Liu, S., Liao, G. & Ding, Y., 2018. *Stock transaction prediction modeling and analysis based on LSTM,* Wuhan: 13th IEEE Conference on Industrial Electronics and Applications.

Moro, M. F., Weise, A. D. & Bornia, A. C., 2020. Model Hybrid for Sales Forecast for the Housing Market of Sao Paulo. *Real Estate Management and Valuation,* 28(3).

Mukherjee, S. et al., 2021. Stock market prediction using deep learning algorithms. *CAAI Transactions on Intelligence Technology,* 8(1), pp. 82-94.

Onyshchak, O., 2020. *Stock Market Dataset.* [Online]
Available at: https://www.kaggle.com/datasets/jacksoncrow/stock-market-dataset
[Accessed 21 October 2022].

Parmer, I. et al., 2018. *Stock Market Prediction Using Machine Learning,* Jalandhar: First International Conference on Secure Cyber Computing and Communication.

Reddy, V. K. S., 2018. Stock Market Prediction Using Machine Learning. *International Research Journal of Engineering and Technology (IRJET),* 5(10), pp. 1032-1035.

Rita, 2022. *!D CNNs: An Introduction To Deep Learning For One-Dimensional Data.* [Online]
Available at: https://www.surfactants.net/1d-cnns-an-introduction-to-deep-learning-for-one-dimensional-data/#:~:text=There%20is%20a%20significant%20difference%20between%201D%20and,slides%20across%20both%20dimensions%20of%20the%20input%20data.
[Accessed 6 April 2023].

Rout, A., Bar, A. K., Saha, S. P. & Chaudhuri, A., 2022. Stock Market Prediction using Machine Learning Algorithm. *International Journal of Advanced Research in Computer and Communication Engineering,* 11(3), pp. 2319-5940.

Saxen, A., 2021. *How to Calculate Mean Absolute Error in Python?.* [Online]
Available at: https://www.geeksforgeeks.org/how-to-calculate-mean-absolute-error-in-python/
[Accessed 1 April 2023].

Seth, S., 2022. *Technical Analysis Strategies for Beginners.* [Online]
Available at: https://www.investopedia.com/articles/active-trading/102914/technical-analysis-strategies-beginners.asp

Shah, J., Vaidya, D. & Shah, M., 2022. A comprehensive review on multiple hybrid deep learning approaches for stock prediction. *Intelligent Systems with Applications.*

Shalloway, B., 2020. *Weighting Confusion Matrices by Outcomes and Observations.* [Online]
Available at: https://www.bryanshalloway.com/2020/12/08/weighting-classification-outcomes/
[Accessed 10 04 2023].

Sharma, A., Bhuriya, D. & Singh, U., 2017. *Survey of stock market prediction using machine learning approach,* Coimbatore: International conference of Electronics, Communication and Aerospace Technology (ICECA).

Shelley, M., 2020. *GeekforGeeks.* [Online]
Available at: https://www.geeksforgeeks.org/introduction-to-stock-market-algorithms/
[Accessed 14 November 2022].

Sunny, I., Maswood, M. & Alharbi, A., 2020. *Deep Learning-Based Stock Price Prediction Using LSTM and Bi-Directional LSTM Model,* Niles: 2nd Novel Intelligent and Leading Emerging Sciences Conference.

Tam, A., 2021. *Using CNN for financial time series prediction.* [Online]
Available at: https://machinelearningmastery.com/using-cnn-for-financial-time-series-prediction/
[Accessed 27 February 2023].

Vaiz, S. J. & Ramaswami, M., 2016. A Study on Technical Indicators in Stock Price Movement Prediction Using Decision Tree Algorithms. *American Journal of Engineering Research,* 5(12), pp. 207-212.

Vargas, M. R., Lima, B. & Evsukoff, A. G., 2017. *Deep learning for stock market prediction from financial news articles.* Annecy, IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications.

Verma, I., Dey, L. & Meisheri, H., 2017. *Detecting, Quantifying and Accessing impact of News events on Indian Stock Indices,* Leipzig: Proceedings of WI'17.

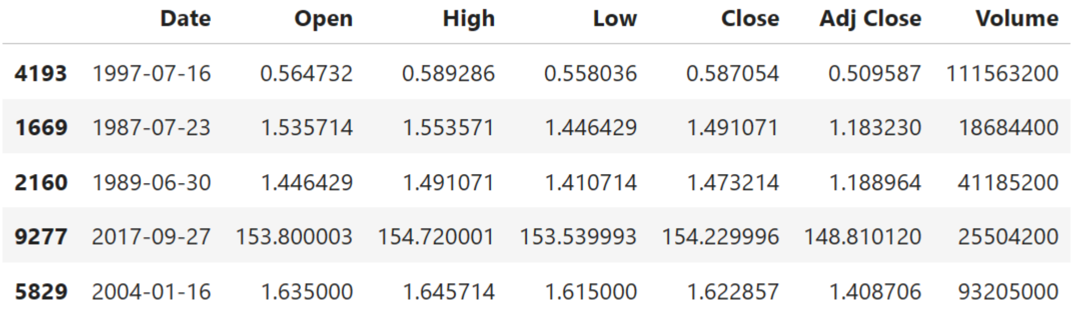
Zach, 2021. *How to Interpret Root Mean Square Error (RMSE).* [Online]
Available at: https://www.statology.org/how-to-interpret-rmse/
[Accessed 1 April 2023].

Zuhaib, M., 2019. *Demystifying the Confusion Matrix Using a Business Example.* [Online]
Available at: https://towardsdatascience.com/demystifying-confusion-matrix-29f3037b0cfa
[Accessed 10 April 2023].

9 Appendices

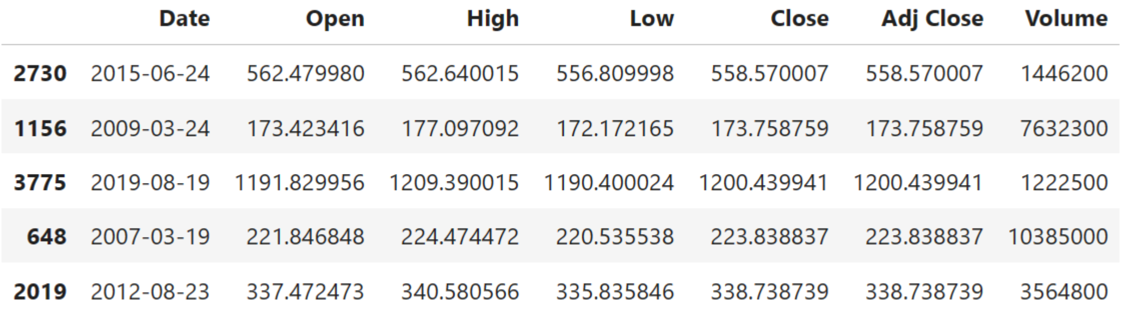
GitHub Link: https://github.com/alshaima201/FYP.git

9.1 Appendix A: Results of EDA



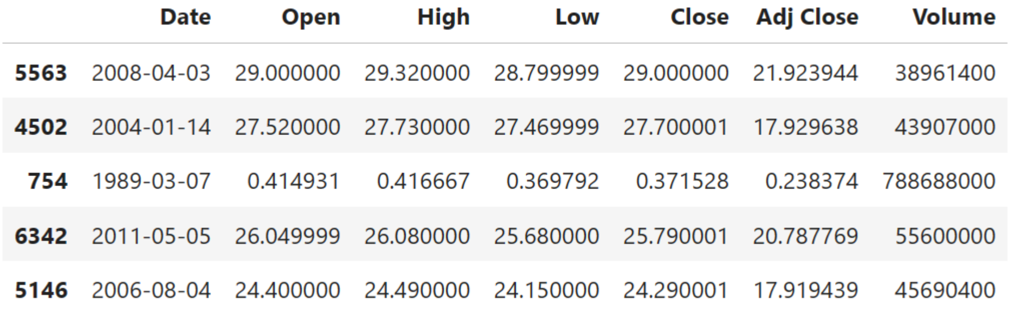
	Date	Open	High	Low	Close	Adj Close	Volume
4193	1997-07-16	0.564732	0.589286	0.558036	0.587054	0.509587	111563200
1669	1987-07-23	1.535714	1.553571	1.446429	1.491071	1.183230	18684400
2160	1989-06-30	1.446429	1.491071	1.410714	1.473214	1.188964	41185200
9277	2017-09-27	153.800003	154.720001	153.539993	154.229996	148.810120	25504200
5829	2004-01-16	1.635000	1.645714	1.615000	1.622857	1.408706	93205000

Figure 25: five-row sample of apple stock dataframe



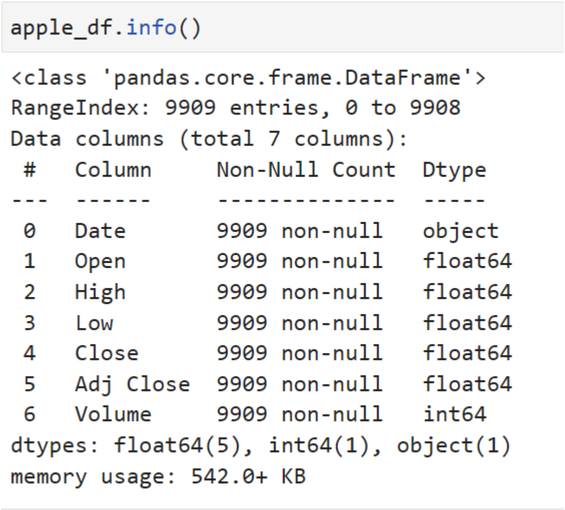
	Date	Open	High	Low	Close	Adj Close	Volume
2730	2015-06-24	562.479980	562.640015	556.809998	558.570007	558.570007	1446200
1156	2009-03-24	173.423416	177.097092	172.172165	173.758759	173.758759	7632300
3775	2019-08-19	1191.829956	1209.390015	1190.400024	1200.439941	1200.439941	1222500
648	2007-03-19	221.846848	224.474472	220.535538	223.838837	223.838837	10385000
2019	2012-08-23	337.472473	340.580566	335.835846	338.738739	338.738739	3564800

Figure 26: five-row sample of alphabet stock dataframe



	Date	Open	High	Low	Close	Adj Close	Volume
5563	2008-04-03	29.000000	29.320000	28.799999	29.000000	21.923944	38961400
4502	2004-01-14	27.520000	27.730000	27.469999	27.700001	17.929638	43907000
754	1989-03-07	0.414931	0.416667	0.369792	0.371528	0.238374	788688000
6342	2011-05-05	26.049999	26.080000	25.680000	25.790001	20.787769	55600000
5146	2006-08-04	24.400000	24.490000	24.150000	24.290001	17.919439	45690400

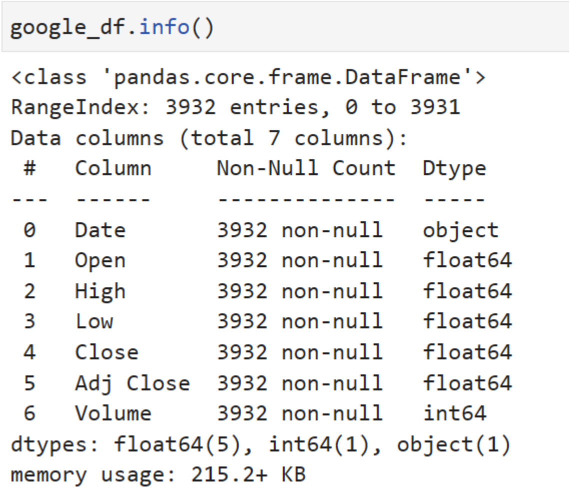
Figure 27: five-row sample of Microsoft stock dataframe



```
apple_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9909 entries, 0 to 9908
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       9909 non-null   object 
 1   Open       9909 non-null   float64
 2   High       9909 non-null   float64
 3   Low        9909 non-null   float64
 4   Close      9909 non-null   float64
 5   Adj Close  9909 non-null   float64
 6   Volume     9909 non-null   int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 542.0+ KB
```

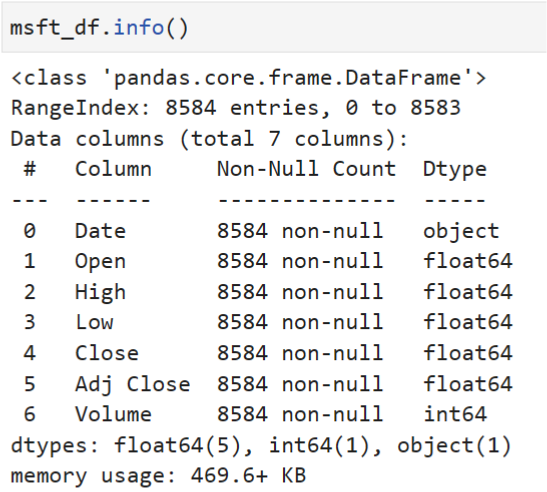
Figure 28: information of apple stock dataframe



```
google_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3932 entries, 0 to 3931
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       3932 non-null   object 
 1   Open       3932 non-null   float64
 2   High       3932 non-null   float64
 3   Low        3932 non-null   float64
 4   Close      3932 non-null   float64
 5   Adj Close  3932 non-null   float64
 6   Volume     3932 non-null   int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 215.2+ KB
```

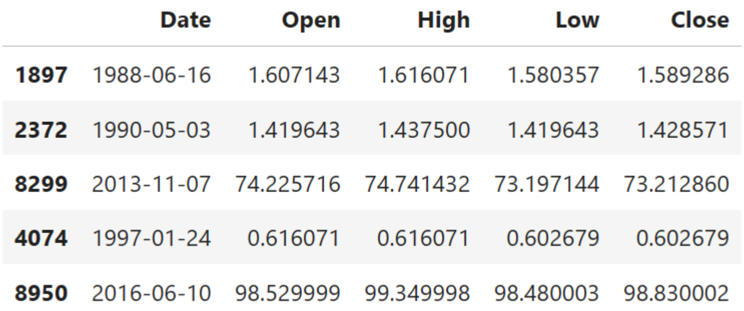
Figure 29: information of Alphabet stock dataframe



```
msft_df.info()
```

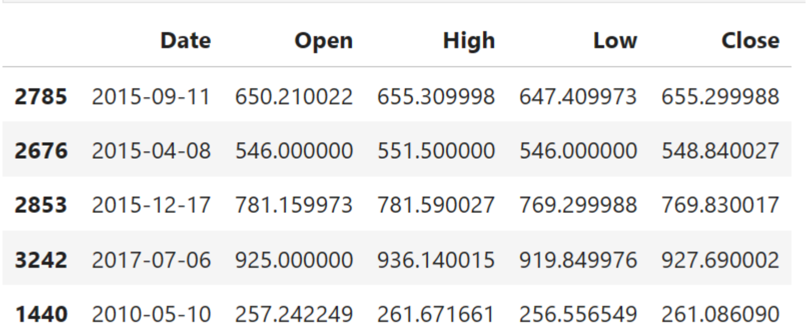
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8584 entries, 0 to 8583
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       8584 non-null   object 
 1   Open       8584 non-null   float64
 2   High       8584 non-null   float64
 3   Low        8584 non-null   float64
 4   Close      8584 non-null   float64
 5   Adj Close  8584 non-null   float64
 6   Volume     8584 non-null   int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 469.6+ KB
```

Figure 30: information of Microsoft stock dataframe



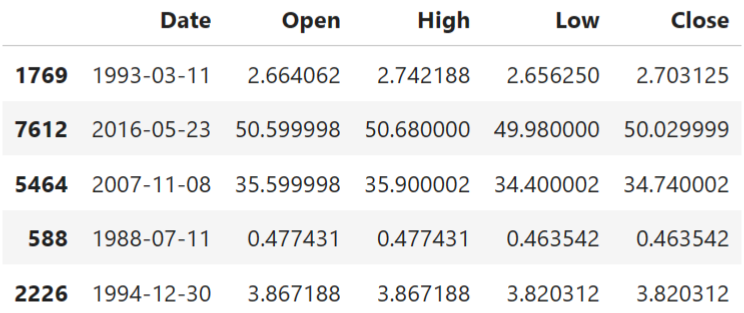
	Date	Open	High	Low	Close
1897	1988-06-16	1.607143	1.616071	1.580357	1.589286
2372	1990-05-03	1.419643	1.437500	1.419643	1.428571
8299	2013-11-07	74.225716	74.741432	73.197144	73.212860
4074	1997-01-24	0.616071	0.616071	0.602679	0.602679
8950	2016-06-10	98.529999	99.349998	98.480003	98.830002

Figure 31: five-row sample of Apple dataframe after data pre-processing



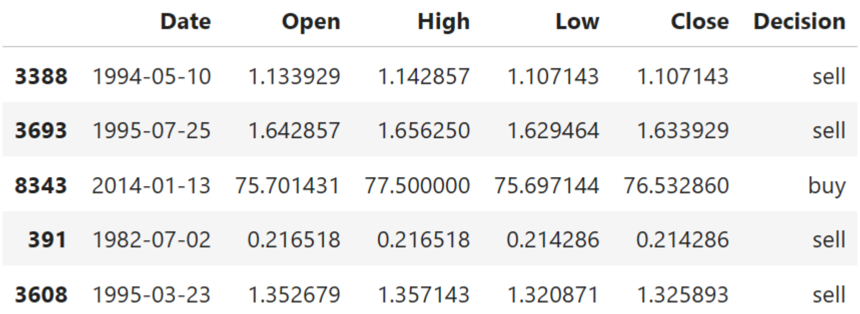
	Date	Open	High	Low	Close
2785	2015-09-11	650.210022	655.309998	647.409973	655.299988
2676	2015-04-08	546.000000	551.500000	546.000000	548.840027
2853	2015-12-17	781.159973	781.590027	769.299988	769.830017
3242	2017-07-06	925.000000	936.140015	919.849976	927.690002
1440	2010-05-10	257.242249	261.671661	256.556549	261.086090

Figure 32: five-row sample of Alphabet dataframe after data pre-processing



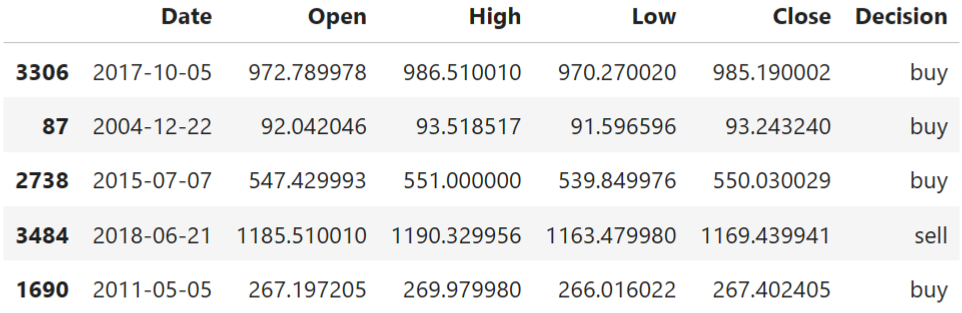
	Date	Open	High	Low	Close
1769	1993-03-11	2.664062	2.742188	2.656250	2.703125
7612	2016-05-23	50.599998	50.680000	49.980000	50.029999
5464	2007-11-08	35.599998	35.900002	34.400002	34.740002
588	1988-07-11	0.477431	0.477431	0.463542	0.463542
2226	1994-12-30	3.867188	3.867188	3.820312	3.820312

Figure 33: five-row sample of Microsoft dataframe after data pre-processing



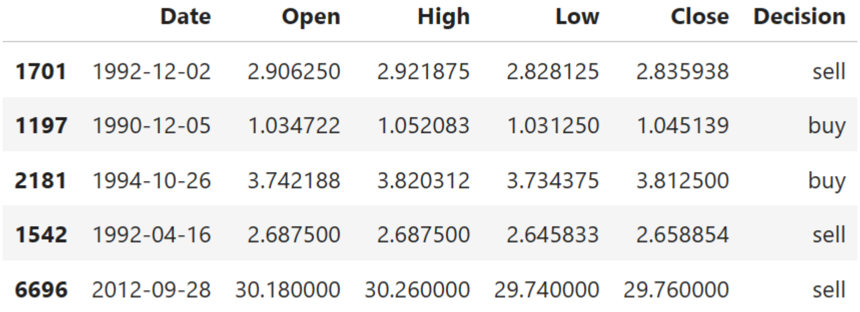
	Date	Open	High	Low	Close	Decision
3388	1994-05-10	1.133929	1.142857	1.107143	1.107143	sell
3693	1995-07-25	1.642857	1.656250	1.629464	1.633929	sell
8343	2014-01-13	75.701431	77.500000	75.697144	76.532860	buy
391	1982-07-02	0.216518	0.216518	0.214286	0.214286	sell
3608	1995-03-23	1.352679	1.357143	1.320871	1.325893	sell

Figure 34: five-row sample of apple dataframe after adding new column 'Decision'



	Date	Open	High	Low	Close	Decision
3306	2017-10-05	972.789978	986.510010	970.270020	985.190002	buy
87	2004-12-22	92.042046	93.518517	91.596596	93.243240	buy
2738	2015-07-07	547.429993	551.000000	539.849976	550.030029	buy
3484	2018-06-21	1185.510010	1190.329956	1163.479980	1169.439941	sell
1690	2011-05-05	267.197205	269.979980	266.016022	267.402405	buy

Figure 35: five-row sample of Alphabet dataframe after adding new column 'Decision'



	Date	Open	High	Low	Close	Decision
1701	1992-12-02	2.906250	2.921875	2.828125	2.835938	sell
1197	1990-12-05	1.034722	1.052083	1.031250	1.045139	buy
2181	1994-10-26	3.742188	3.820312	3.734375	3.812500	buy
1542	1992-04-16	2.687500	2.687500	2.645833	2.658854	sell
6696	2012-09-28	30.180000	30.260000	29.740000	29.760000	sell

Figure 36 :five-row sample of Microsoft dataframe after adding new column 'Decision'

9.2 Appendix B: Results of ML models

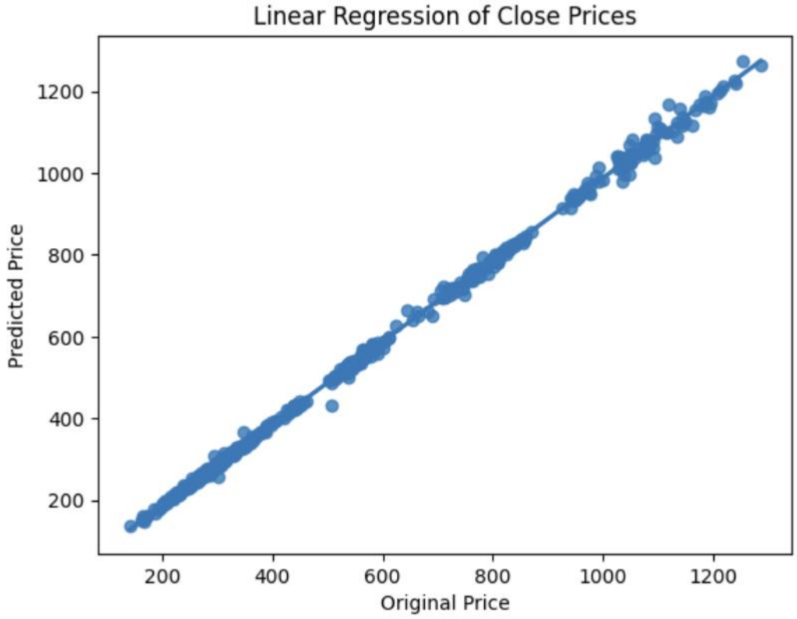


Figure 37: Linear Regression on Alphabet Stocks

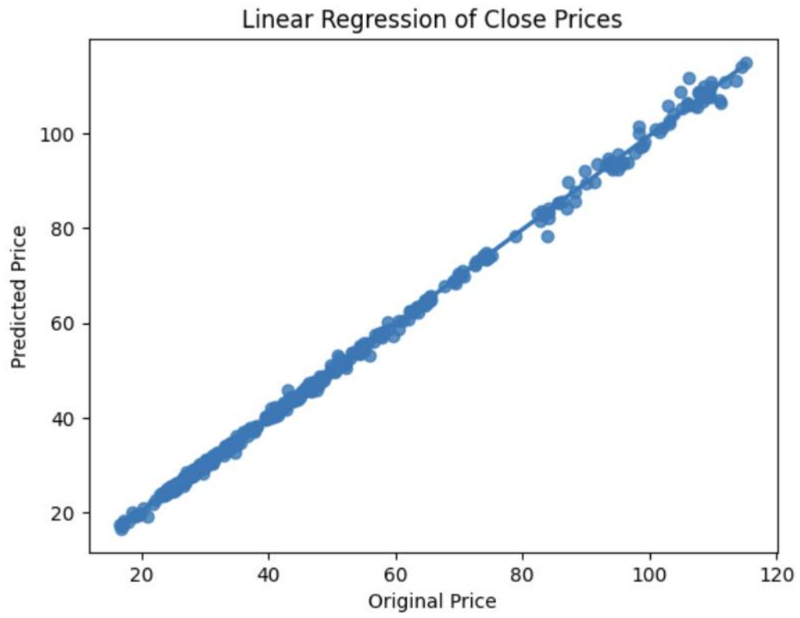


Figure 38: Linear Regression on Microsoft Stocks

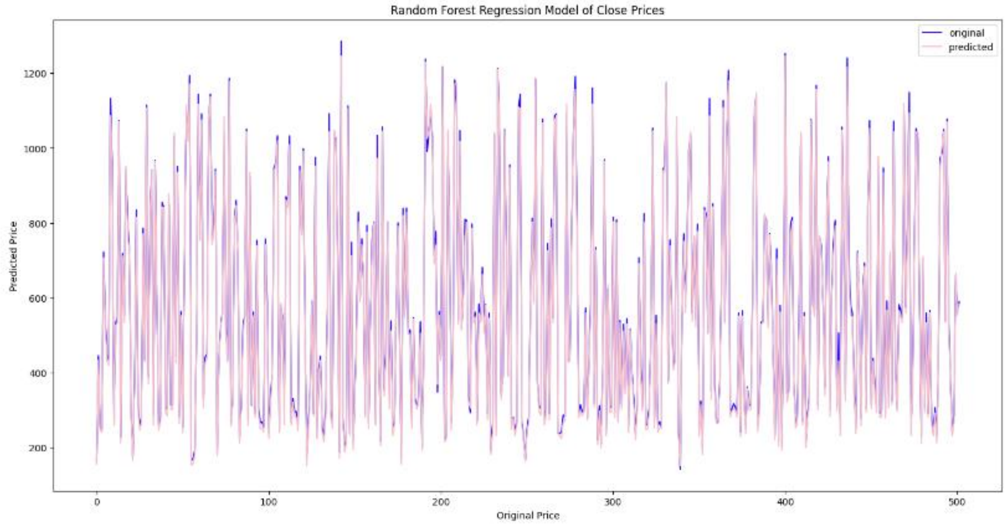


Figure 39: actual vs predicted results of Random Forest on Alphabet stocks

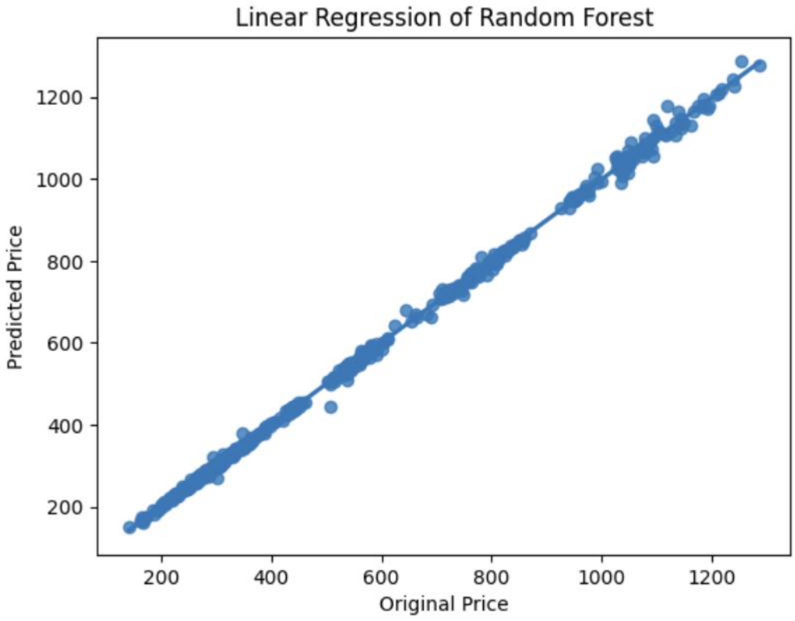


Figure 40: Linear Regression of Random Forest on Alphabet stocks

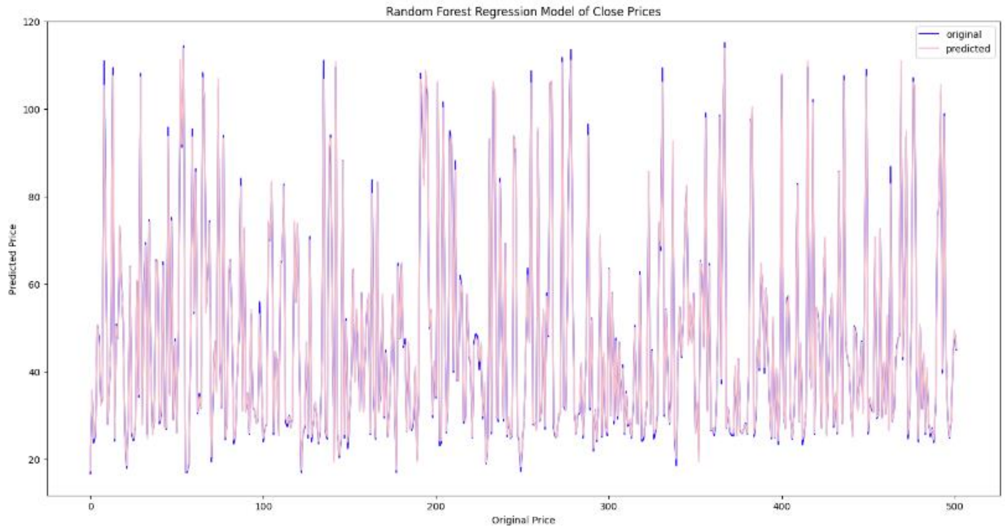


Figure 41: actual vs predicted results of Random Forest on Microsoft stocks

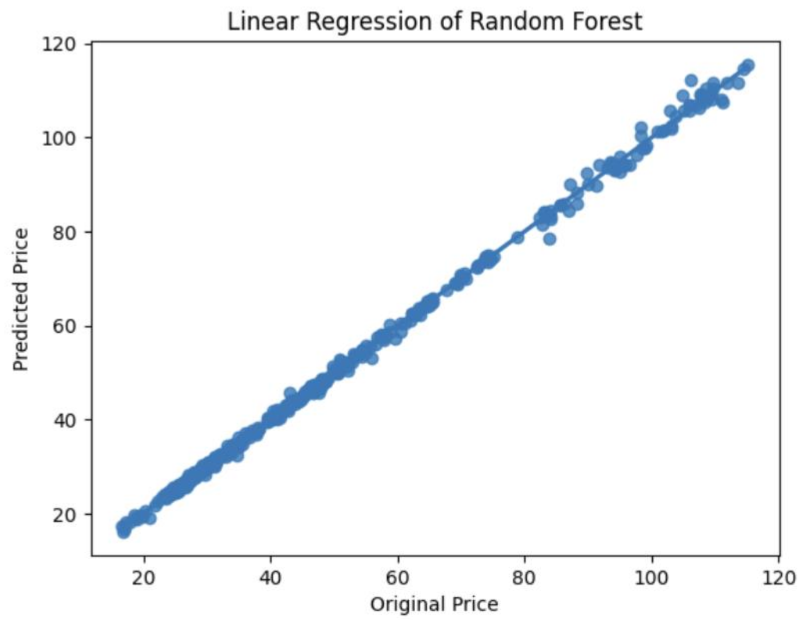


Figure 42: Linear Regression of Random Forest on Microsoft stocks

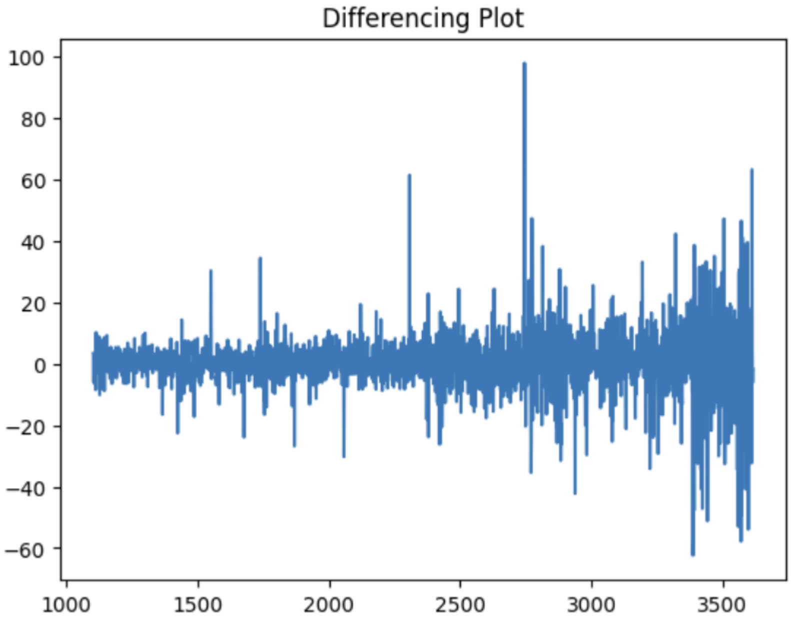


Figure 43: differencing plot - Alphabet stocks

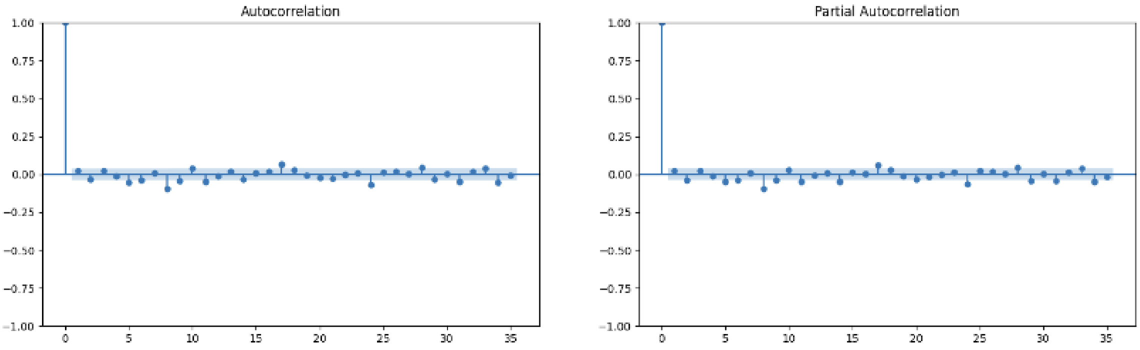


Figure 44: ACF & PACF plots - Alphabet stocks

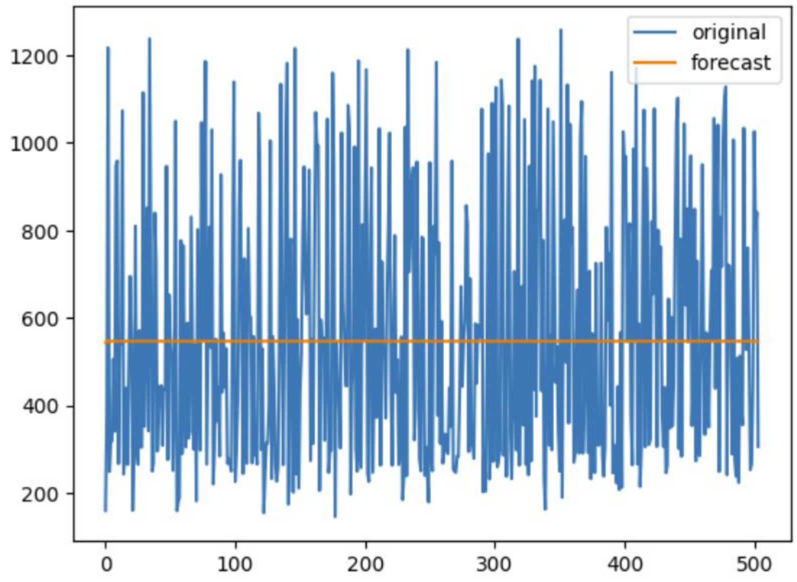


Figure 45: visualising actual vs forecast results of ARIMA on Alphabet stocks

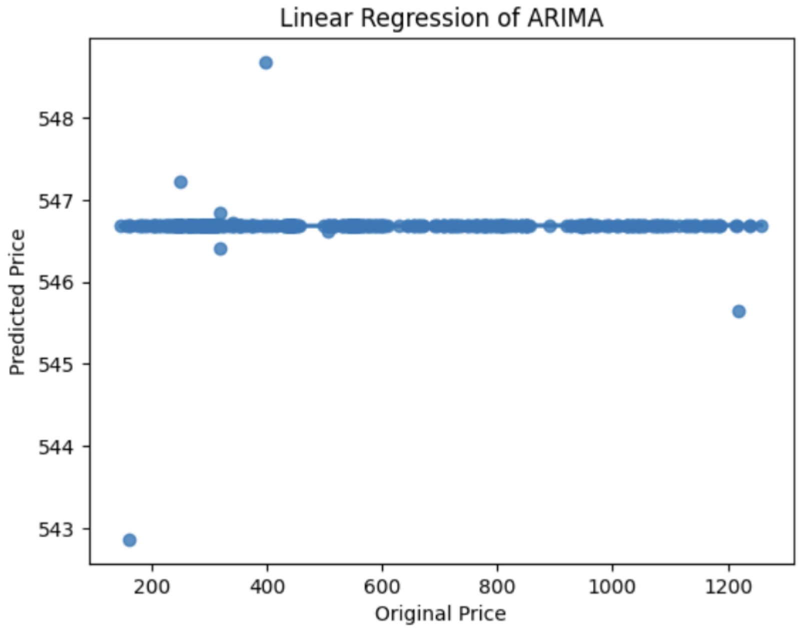


Figure 46: Linear Regression of ARIMA on Alphabet stocks

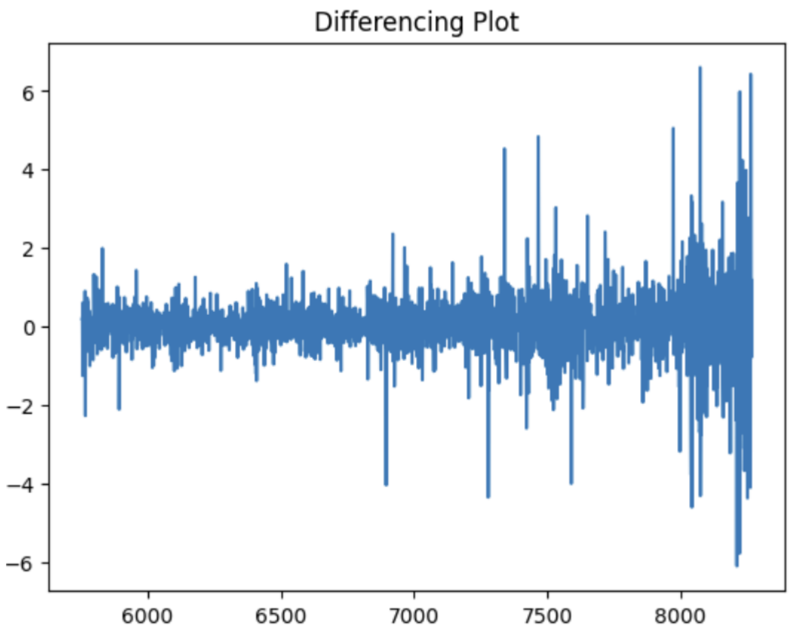


Figure 47: differencing plot - Microsoft stocks

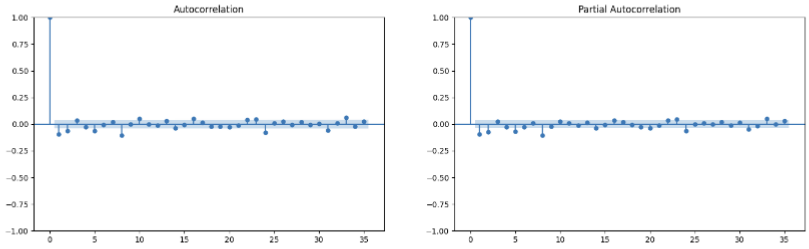


Figure 48: ACF & PACF plots - Microsoft stocks

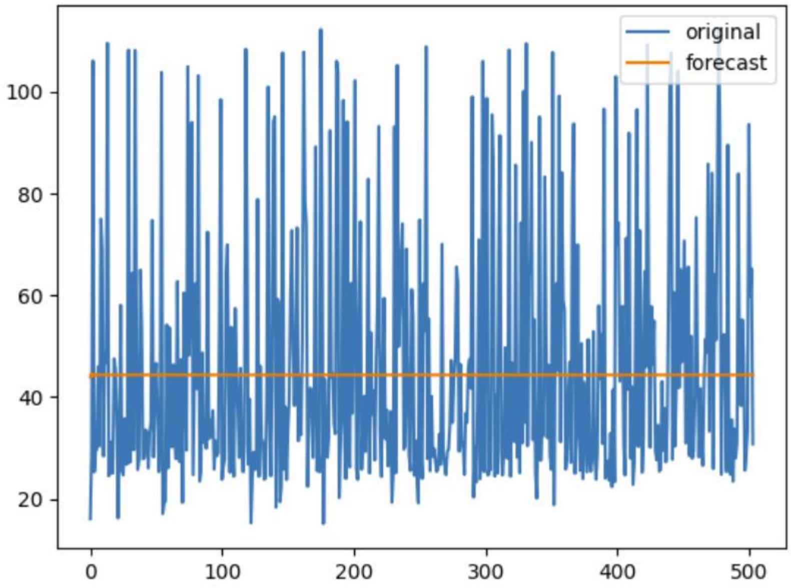


Figure 49: visualising actual vs forecast results of ARIMA on Microsoft stocks

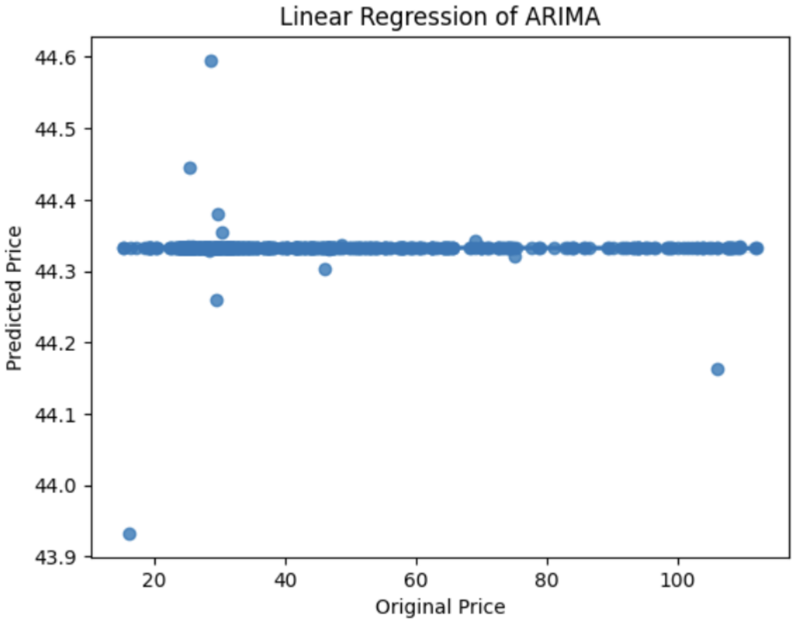


Figure 50: Linear Regression of ARIMA on Microsoft stocks

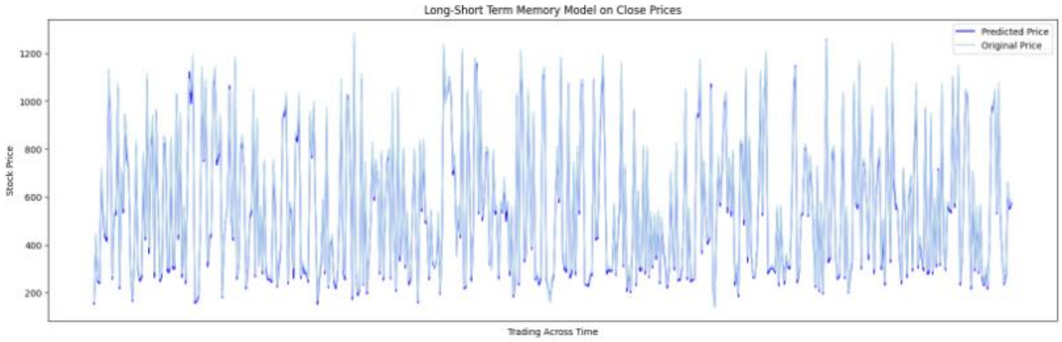


Figure 51: LSTM results on Alphabet dataset

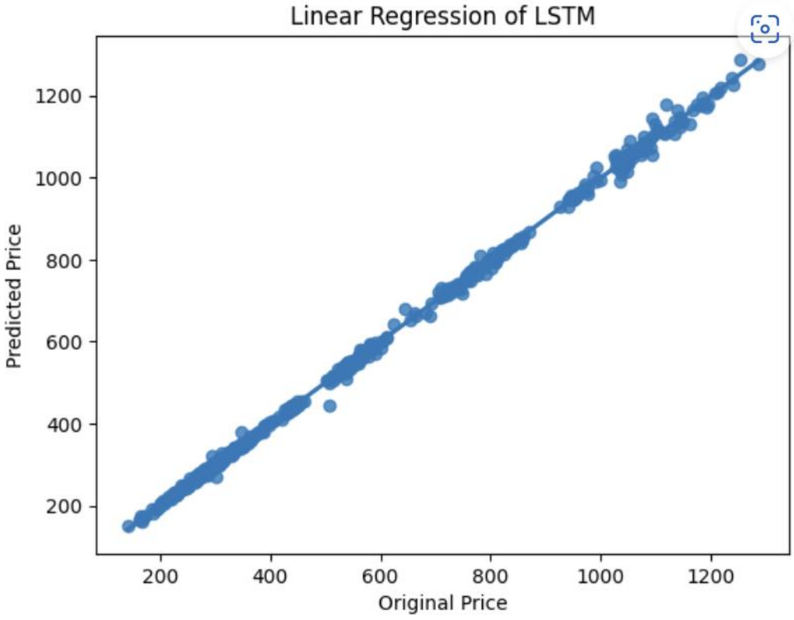


Figure 52: Linear Regression of LSTM on Alphabet dataset

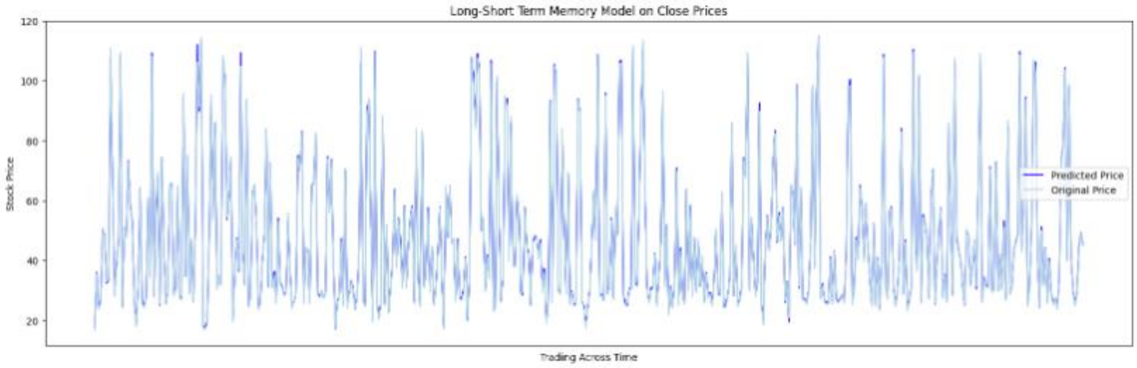


Figure 53: LSTM results on Microsoft dataset

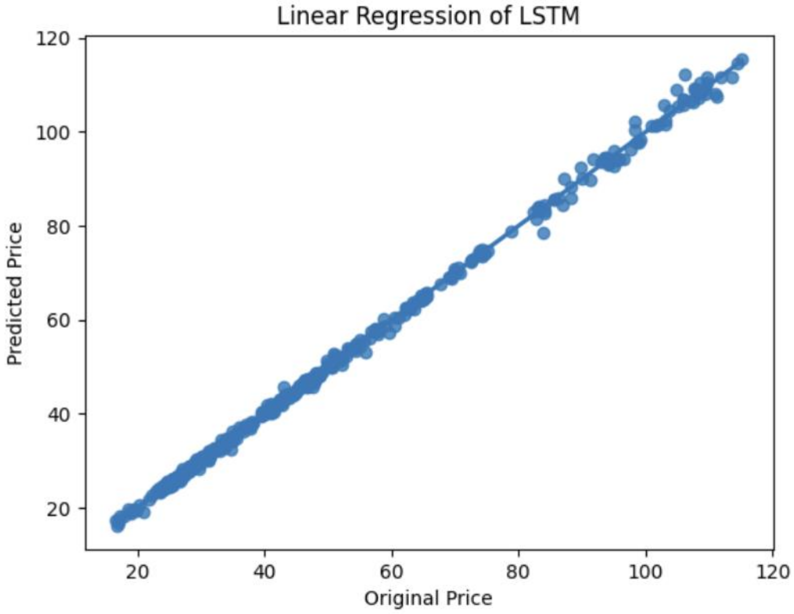


Figure 54: Linear Regression of LSTM on Microsoft dataset

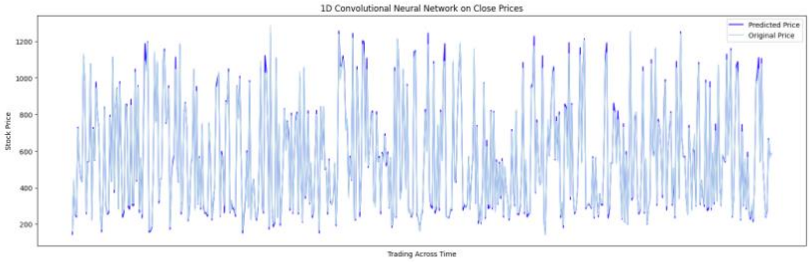


Figure 55: CNN results on Alphabet dataset

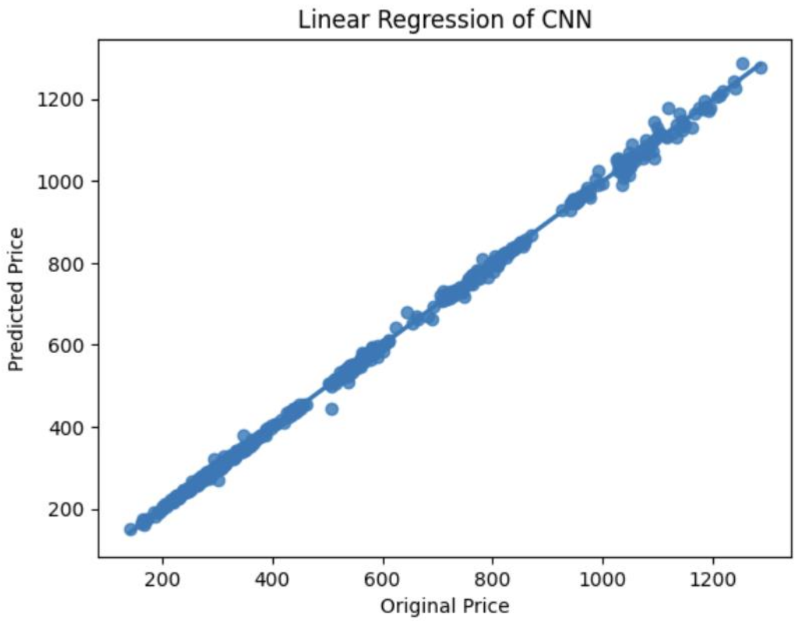


Figure 56: Linear Regression of CNN on Alphabet dataset

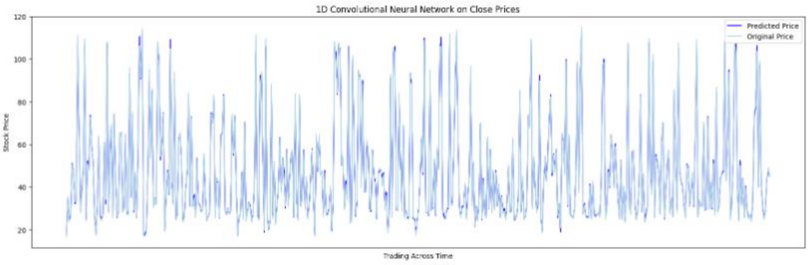


Figure 57: CNN results on Microsoft dataset

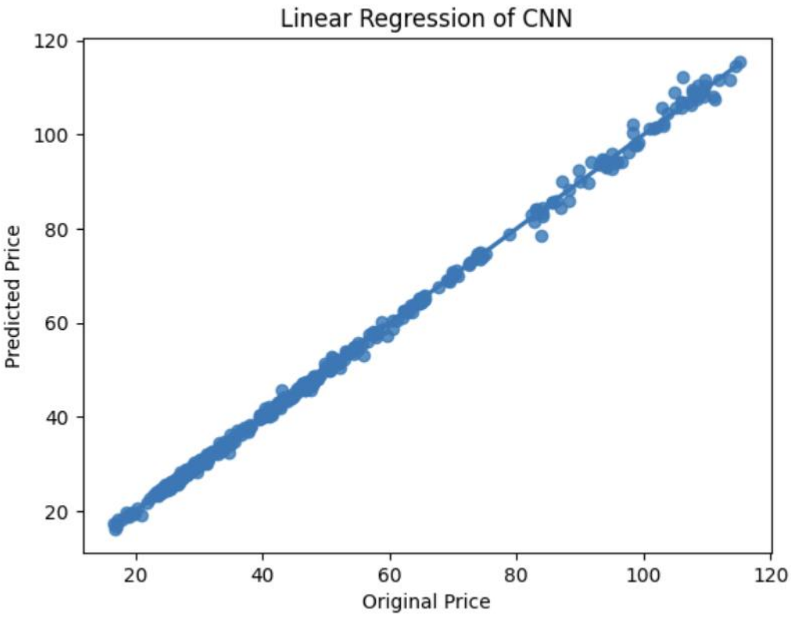


Figure 58: Linear Regression of CNN on Microsoft dataset

9.3 Appendix C: Classification results

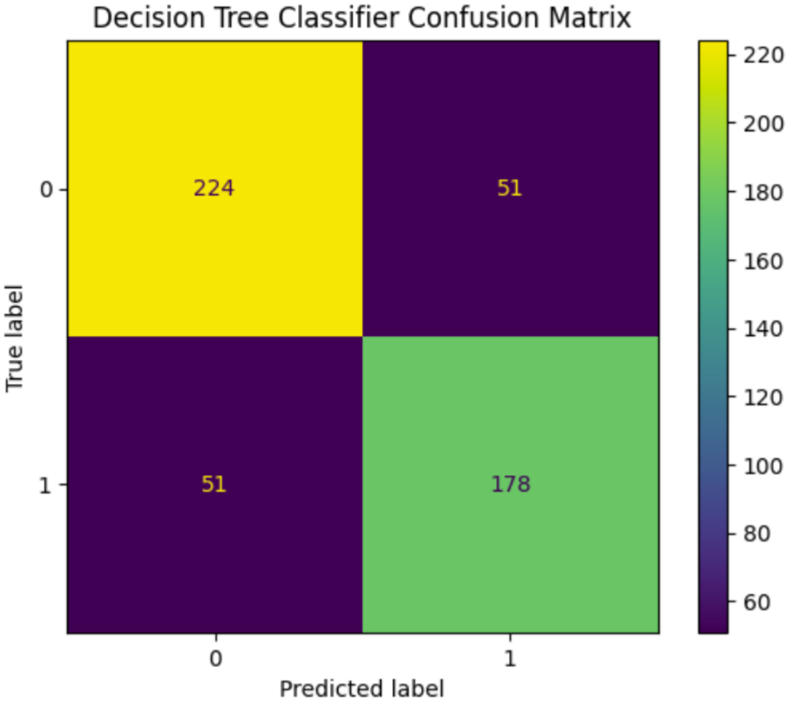


Figure 59: Confusion Matrix of Decision Tree results - Alphabet dataset

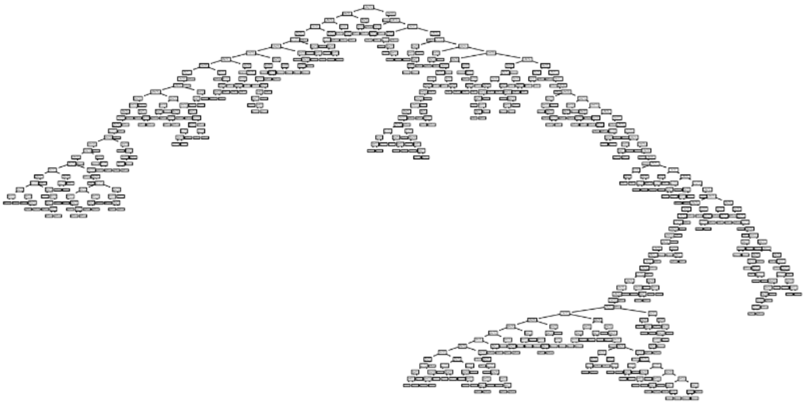


Figure 60: decision tree visualisation - Alphabet dataset

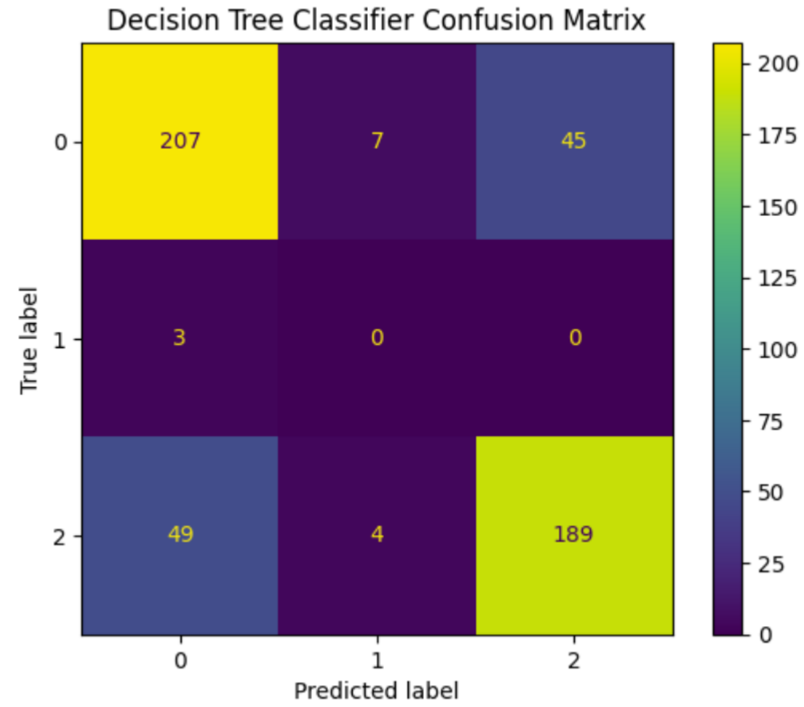


Figure 61: Confusion Matrix of Decision Tree results - Microsoft dataset

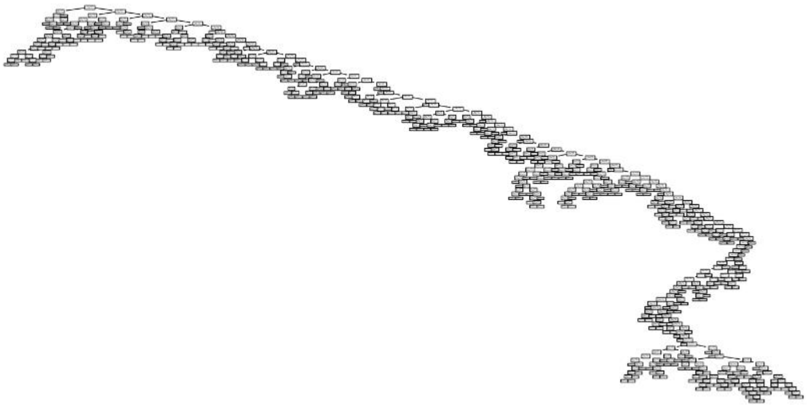


Figure 62: decision tree visualisation - Microsoft dataset