Department of Computer Science

Submitted in part fulfilment for the degree of BSc.

Financial Forecasting: Comparing Traditional and Modern Approaches When Forecasting Index Funds

Harry Hillman

Version 3.0, 2024-May

Supervisor: Dimitar Kazakov

To future growth and progress

Acknowledgements

I would like to thank my parents for their unwavering support and the effort they have always put in. Their constant encouragement has got me to a position where I can study Computer Science and pursue my passions as well as my goals, I am profoundly grateful for their love and nurturing care.

Additionally, thank you to my supervisor, Dimitar Kazakov, for his support and guidance throughout this study.

Contents

List of Figures

List of Tables

Executive Summary

This study aims to focus on comparing methods of providing forecasts, so as to allow for an effective financial strategy to be built around the predictions. With the benefits - namely profit - that come from being able to predict the financial markets, lots of research has been put into developing newer, better ways to produce a forecast for the behaviour of historical data. A higher accuracy forecast produced by models can allow for better performance when executing different trading strategies, as higher certainty about the future can allow for better preparation for the future. The forecasts can be applied in numerous fields to allow better planning, whether it be: budgeting and planning, allowing businesses to manage money better; insurance, performing risk assessment of individuals/activities based on past participants with similar characteristics; financial forecasting, executing stock market analysis to indicate when is the best time to buy/sell a stock for maximum profit.

When examining the goal of this study from a social standpoint, it is important to note that people/entities should only invest money which they can afford to lose; this is because unforeseen circumstances can affect the financial markets, causing a stock to lose all value - and with it the investors money. The ethics of investing are a complicated topic, as when investing there is competition against other investors to gain the maximum profit; this competition can ultimately lead to other entities losing their hard earned money [1]. The results from this study are to compare the effectiveness of different predictive models, and they are intended for academic purposes only, with any use separate from academics to be performed at the users own risk. Practitioners should carefully evaluate the appropriateness for a forecasting model to be applied to their own data

There are diverse applications for financial forecasting, ranging from predicting macroeconomic indicators [2] like GDP to forecasting the financial market indices for trading purposes. Whilst forecasting economic indicators aids in the understanding of economic trends and informing policy decisions, focusing on financial markets - particularly index funds - allows traders to anticipate market movements and make informed investing decisions. In the context of this study, the emphasis lies on forecasting market indices to capture trends within the financial markets.

The data this study covers a range of index funds, which include various markets and different companies. The funds analysed are those typically in the spotlight for good performance: Dow Jones Industrial Average (DJI), FTSE 250 (FTMC), S&P 500 (GSPC), NASDAQ (IXIC), and Russell 2000 (RUT). When dealing with these datasets it is important to note the varying scale at which they operate (some with prices ranging up to £30000, whilst others remain at around £2000. This data was retrieved from publicly accessible sources, so as to keep the findings of this study accessible and reproducible. A range of traditional statistical methods and modern techniques were employed to produce forecasts for the financial data. The statistical methods covered were ARIMA (AutoRegressive Moving Average) and VARMA (Vector AutoRegressive Moving Average), whilst the modern techniques employed were LSTM (Long Short-Term Memory) and Random Forest. These models were produced and analysed using public, open-source libraries. The results found that Random Forest, a modern technique, performed better than all other models covered in this study. However, ARIMA and VARMA provided more consistent error rates than LSTM - as LSTM proved inaccurate when dealing with all the different indices.

The studys conclusion found that, after performing ANOVA(ANalysis Of Variance) on the forecasts produced, traditional methods are outperformed by more modern approaches to financial forecasting. The forecasts generated by the modern techniques, Random Forest and LSTM, exhibit closer alignment with the actual data, when compared with those produced by traditional methods. This disparity can be attributed to advancements in mathematics and technology, as machines used for modern computing have significantly more capabilities than older systems. However, when ANOVA was performed on the RMSE found that the errors across all models were not significantly different. Although the models experienced similar error, the advantage of modern techniques lies in their ability to more closely trace the market behaviour.

1 Introduction

Financial Forecasting plays a crucial role in the strategic planning and decision making processes that take place within organisations across the globe, in a broad spectrum of sectors. This tool can be utilised to guide businesses down a path which leads them to more positive financial performance in the future. In modern society, when dealing with companies that experience rapid growth and bring on huge technological advancement, the ability to accurately predict the path financial data will follow has become treasured in fields such as business and economics.

Recently, financial forecasting has been paired with machine learning techniques due to the potential these models have to increase the accuracy of models and automate the intricate and complex decision making processes involved in the financial markets, and the obvious gains which can be acquired through having a predictive model that performs even marginally better than the markets. Modern machine learning methods can offer a data-driven approach to the analysis of the markets, with more adaptability and accuracy than traditional, statistics based methods - which may struggle to capture intricate relationships and non-linear patterns.

This study aims to compare traditional and modern methods of financial forecasting, aiming to explore their effectiveness and evaluate their strengths and weaknesses against each other, whilst assessing the performance of each model by measuring their accuracy. Through making use of vast amounts of historical data, machine learning algorithms can identify more complex patterns than the less resource heavy statistics and traditional methods. When forecasting indices in industry the aim is to make money, however this study does not focus on financial gain - which requires a combination of trading strategies and forecasting, and is therefore out of the scope of this study; instead of focusing on financial gain, the focus of this study is forecasting market indices.

Throughout this report the effectiveness of multiple techniques will be evaluated and compared with other state of the art techniques. The techniques being employed will be:

- AutoRegressive Integrated Moving Average (ARIMA)
- Vector AutoRegressive Moving Average (VARMA)

- Long Short-Term Memory (LSTM)
- Random Forest

These methods will be applied to forecast the path index funds value will take, using past historical data to gain forecast patterns and dependencies within the data.

This study will research the challenges associated with financial forecasting, in general and specific to each method, including data quality, and overfitting. What also makes forecasting the financial markets difficult is the large scale in real-time causing unpredictability, along with the many participants leading to the "observer effect", where the actions and beliefs of the market participants influence market outcomes .Even with the presence of these challenges, the benefits of financial forecasting are clear; it allows for improved risk management, better future planning, and more stable money management.

To conclude, the potential for the integration of machine learning techniques holds lots of promise to drastically revolutionise existing financial forecasting practices. Using machine learning to gain data-driven insights and recognise advanced patterns that human analysts may not identify, organisations stand to gain a lot from the new technology being researched, and can also reduce the risks they take with the increased knowledge they are granted.

2 Literature Review

2.1 Financial Time Series Forecasting

Time Series Forecasting predicts future values based on the past historical data, commonly used to analyse stock prices, sales figures, population or even astronomy and weather data. In stock price forecasting, two main approaches are employed: *Fundamental analysis* and *technical analysis* [3]. Fundamental analysis focuses on company financial information and industry trends, while technical analysis assumes market behaviour is influenced by previous market behaviour and repeats itself.

The application of time series forecasting offers significant business benefits [4] driving extensive research to enhance forecasting models, especially in financial institutions like banks - where obvious gain is to be made by having an accurate forecasting method. Models range from statistical methods like ARIMA (AutoRegressive Integrated Moving Average) and VARMA (Vector AutoRegressive Moving Average) to advanced techniques such as LSTM (Long Short-Term Memory) and Random Forest, utilising advancements in artificial intelligence and machine learning.

Understanding Time Series Time Series is a sequential arrangement of data points, typically representing measurements of a variable over time, such as: daily stock price, hourly temperature, or monthly population. Analysis of these data points reveals trends and seasonal patterns, aiding decision making processes. Time series data often exhibits regular patterns, such as daily, weekly, monthly, or annual cycles, aswell as irregular noise - which must be filtered out to ensure accurate forecasts [5].

2.2 Financial Forecasting

Origins of Financial Forecasting Financial Forecasting involves utilising historical data, statistical analysis techniques, and expert judgement to predict the future financial performance of a company [6]. The process focuses on key financial indicators, such as past financial statements, sales data, and current/previous market trends.

The importance of financial forecasting is upheld by its ability to enable companies to make informed decisions regarding their business practices. It aids in an organisation's budget creation, goals setting, and optimising operations for economically efficiency [7]. Moreover, the modelling can help companies anticipate large economic changes, market shifts, and industry trends, facilitating early adjustments [8].

Financial forecasting methods have evolved significantly since their emergence in the 1920s [9], facing challenges with accuracy ever since [10]. While qualitative approaches relied on intuition, the modernisation with quantitative approaches offers data-driven, unbiased predictions.

Types of Financial Forecasting Traditionally financial forecasting normally employs two different techniques, qualitative and/or quantitative [11]. Qualitative techniques have the ability to provide insight beyond numerical data and can be very useful when historical data is scarce, although it is subjective which can be difficult to deal with when wanting an objective answer. Quantitative techniques are strictly objective, allowing for precise numerical predictions based on historical data; however, the drawback of quantitative techniques being strictly data-driven is they assume that historical patterns will always continue, whilst real world scenarios have proven this is not the case.

Qualitative Techniques Qualitative techniques, such as interviews, focus groups, and making observations [12], alongside extensive market research, provide insight to forecasted concepts; The Delphi method, for example, involves multiple rounds of surveying a collection of experts on a particular topic for forecasting. Open-ended interview questions yield descriptive insights into consumer behaviour and industry trends, through conversation. These insights, alongside quantitative analysis, give contextual understanding to guide decision making. Literature reviews also contribute to exploring existing ideas and theories [13].However, qualitative techniques are susceptible to underlying bias, such as the Hawthorne Effect [14], due to influence from the researcher and the participants altered behaviour.

Quantitative Techniques Quantitative forecasting relies on historical data and statistics [15]. Multiple-choice surveys, experiments, and recording observations yield time series data for analysis, enabling predictions of future trends, establishing hidden facts. Time series methods, such as moving average, are key in financial modeling when identifying trends. linear regression is a basic quantitative approach to show a relationship between variables. However, quantitative techniques can experience challenges with bias in the data [16]. Errors in data collection can distort forecasts, and overlooking important variables can impact model accuracy. Random sampling is crucial to collect reflect varying experiences and views.

Challenges Within Financial Forecasting Financial forecasting faces the obstacle of unpredictable future events [17], such as the *2007 Subprime Mortgage Crisis*, which led to a global financial downturn [18], highlighting the need for frequent forecast revisions to maintain accuracy amidst real-world fluctuations. External factors can cause deviations from predictions, as models rely on available information. Moreover, forecasting accuracy diminishes with increasing distance from the last observed data point.

2.3 Traditional Approaches to Financial Forecasting

Stationarity Many models rely on the assumption that the data being analysed is stationary, where the statistical properties of data remain consistent over time, i.e. the mean of the data doesn't change. It is an important property to be considered when passing time series data into various forecasting models, like ARIMA [19], as the model assumes that the patterns displayed in the data will continue into the future. If the data is not stationary it becomes more challenging for the model to provide accurate predictions. To get stationary data from non-stationary data the most common approach is to apply differencing to it [20]. First-order differencing subtracts consecutive values, performed by the following equation:

$$X'_t = X_t - X_{t-1} \tag{2.1}$$

Higher order differencing can be applied if First-order differencing does not make the data stationary, applied iteratively until eventually the series produced is stationary [20].

ARIMA (AutoRegressive Integrated Moving Average) ARIMA (AutoRegressive Integrated Moving Average) is a statistical model for analysing and predicting time series data [21]. ARIMA combines three key components: AutoRegressive (AR), Integration (I), Moving Average (MA). The AR component predicts future values based on past values, implying that data behaviour will remain consistent. Integration transforms non-stationary data to stationary data, ensuring its statistical properties remain constant over time. The MA component smooths the data, using a lagged Moving Average to reduce short-term irregularities. ARIMA is commonly used in technical analysis to forecast the price of a given security.

The AutoRegressive portion of ARIMA is defined by the equation [22]:

$$X_t = c + \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p X_{t-p} + \varepsilon_t \tag{2.2}$$

where the error term e_t is assumed a white noise process, X_t is the value of the series at time t, X_{t-1} is the first lagged value of the series, ϕ_1 is the coefficient of lag1, and c is a constant term.

The Moving Average portion of ARIMA can be outlined by the following formula, with a Moving Average of order (q) [20]:

$$X_t = \mu + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \ldots + \theta_q \varepsilon_{t-q} \tag{2.3}$$

where X_t represents the value of the time series at the time of measurement, t. μ is a constant term, and ε_t is the stochastic term at time t, at the time. The Moving Average coefficients represented by θ.

These two equations, along with the addition of differencing, can then be collated to form the equation for ARIMA [23]:

$$X_t = c + \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p X_{t-p} + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \ldots + \theta_q \varepsilon_{t-q} + \varepsilon_t \tag{2.4}$$

Variations of ARIMA ARIMA models can be fine tuned by parameters (p,d,q), to adjust the model for specific characteristics [24] p refers to the number of AutoRegressive terms in the model, which are used to capture the relationship between the current value and its lags. d represents the degree of differencing which has to be applied to make the data stationary, to ensure it remains consistent throughout. q is used for the number of moving average terms, are used to improve predictions being made. Different combinations of ARIMA parameters resemble existing models, ARIMA(0,0,0) is a white noise model (a random signal with equal intensity at different frequencies) [25]. ARIMA(0,1,1) without a constant is equivalent to a simple exponential smoothing model (a model which assumes the data has no trend).

VARMA (Vector AutoRegressive Moving Average) VARMA (Vector AutoRegressive Moving Average) is a tatistical multivariate model that is used for time series forecasting and understanding correlations between multiple variables [26]. Combining Vector AutoRegressive (VAR) and Moving Average (MA), VARMA considers interactions between multiple variables over time.

In VAR, each variable is regressed with its lagged values, as well as the lagged values of all the other variables in the system. This allows VAR to capture the interactions between the variables over time [26]. The AutoRegressive portion draws the implication that future behaviour of a variable is influenced by its past patterns and also the past values of other variables, which differs from how ARIMA behaves.

The Moving Average (MA) component models the stochastic distribution, smoothing short-term irregularities in the data (those that cannot be captured by the AutoRegressive component), making it easier for the VARMA model to capture underlying trends and patterns.

The equations for VARMA can be represented as follows:

Vector AutoRegressive (VAR) [27]:

$$Y_t = \alpha + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \ldots + \phi_p Y_{t-p} + \varepsilon_t \tag{2.5}$$

Moving Average (MA) [20]:

$$Y_t = \mu + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \ldots + \theta_q \varepsilon_{t-q} \tag{2.6}$$

Vector AutoRegressive Moving Average (VARMA):

$$Y_t = \alpha + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \ldots + \phi_p Y_{t-p} + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \ldots + \theta_q \varepsilon_{t-q} + \varepsilon_t \tag{2.7}$$

Where Y_t is a vector of variables at time t, α is an intercept constant, ϕ is a matrix of AR coefficients, θ is a matrix of MA coefficients and ε_t represents white noise.

VARMA is utilised in various fields, particularly finance and economics [28], due to its ability to consider multiple variables. Its multivariate capabilities facilitate forecasting by considering the relationship between data, like stocks prices connection to gold prices [2]. However, this complexity can reduce the models interpretability, requiring an understanding of variable relationships. Large parameter spaces in VARMA models increase the risk of overfitting and prediction uncertainty, while increasing computational complexity, by necessitating resource intensive likelihood estimation and optimisation techniques.

Historical Applications ARIMA and VARMA, as widely used traditional time series forecasting models, have found widespread use across a range of fields, including business, pollution modelling, environmental monitoring, financial time series analysis, and healthcare, for disease outbreaks and the demand put on hospitals [29]. While ARIMA handles time related forecasts, VARMA is applied to model the interdependencies among multiple variables. Both models have significantly contributed to improving decision making and societal outcomes, by providing insights from historical data.

2.4 Modern Approaches to Financial Forecasting

Deep Learning Deep learning interest has surged in both research and practical applications since 2010, driven by recent advancements in computational power and also the increased availability of large datasets [30]. This rise has impacted fields such as computer vision(e.g., image classification), speech recognition, natural language processing (e.g.,language translation), and robotics (for example, autonomous vehicles). Within Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs) have been two main practical usages of RNNs [31]; LSTM introduced a more powerful and flexible method of modelling sequential data with the use of specialised gates, compared to simple RNNs [32].

LSTM (Long Short-Term Memory) LSTM (Long Short-Term Memory) is a type of Recurrent Neural Network (RNN) that is designed to mitigate the “vanishing gradient problem” by enabling the model to maintain information from earlier in the input sequence for extended periods [33]. Unlike conventional RNNs, LSTM is proficient at identifying long-term dependencies, making it well suited for time series forecasting [34].

LSTM’s superiority stems from its composition of memory cells, which are used to hold and process information. Each memory cell is composed of three main components called gates, in particular:Input gate, Forget gate and Output gate [34]. A diagram of an LSTM memory cell is shown in Figure 2.1 [35]:

The forget gate discards irrelevant information from the previous state [34], a crucial difference between LSTM and simple RNNs. The equation for the forget gate is defined below [36]:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.8}$$

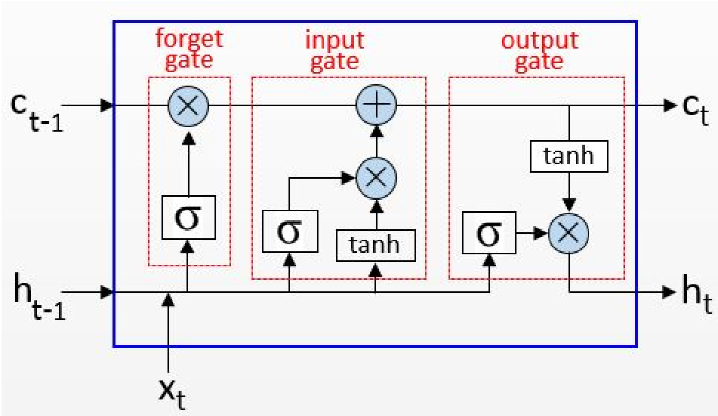

Figure 2.1: LSTM Memory Cell [35]

The Input Gate regulates the flow of relevant information into the memory cell [34], using a sigmoid function. The equation for the input gate is defined below [36]:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.9}$$

After the forget gate has selected relevant information, and the input gate has selected the new information to be input, the cell is updated to contain remembered and new information. The equations contributing to cell state are shown below [36]:

$$C_t' = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{2.10}$$

$$C_t =_t *C_{t-1} + i_t * C_t' \tag{2.11}$$

Then the output gate is activated [34], this controls the final output of the LSTM memory cell at each timestep. The relevant equations are defined below [36].

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.12}$$

$$h_t = o_t * \tanh(C_t) \tag{2.13}$$

Ensemble Learning Ensemble Learning combines multiple models to improve predictive performance, beyond the capability of one sole model. The underlying principle of ensemble learning is that aggregating the predictions of various models leads to a more accurate, robust, and general prediction [37]. Diversity among the models is sought after, to mitigate errors, with ensemble methods reducing bias and variance in the overall prediction. One common ensemble technique, Bagging (Bootstrap Aggregating) [38], involves training multiple base models on random subsets of the training data and then averaging their predictions to obtain the final output.

Random Forest Random Forest (RF) is a machine learning algorithm, that employs ensemble learning principles, for classification and regression across multiple domains, including financial forecasting [39]. In regression tasks, it operates by creating multitudes of decision trees during training and outputting their mean prediction. The "Random" component involves training each individual tree on random subsets of data and features, this introducing diversity and reducing the risk of overfitting [40]. The "Forest" component refers to its ensemble of decision trees, which enhances predictive performance.

To prepare data for RF training, relevant features, particularly lagged data for time series forecasting, are incorporated. During training, RF randomly samples data subsets with replacement and selects random features for each tree split. Trees are grown recursively to minimise errors until a stopping criteria is met, such as a maximum tree depth or a minimum node size. Predictions are then calculated by averaging the predictions of all trees. A diagram of how a random forest operates is included in Figure 2.2 [41]:

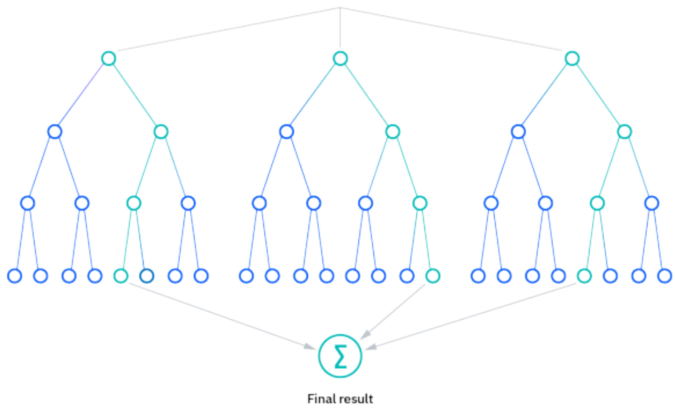

Figure 2.2: Random Forest Diagram [41]

Random Forest's combination of decision trees mitigates overfitting, and increases the prediction accuracy. Through Bootstrap Aggregation [38], RF produces higher quality forecasts, especially effective for, large, high dimensionality datasets. Despite its high accuracy, Random Forest lacks interpretability when compared against simpler models, like ARIMA or a standard linear regression [42], posing challenges when trying to understand its inner workings and influential factors.

2.5 Model Comparative Analysis

Comparison ARIMA, VARMA, LSTM and Random Forest models provide clear uses in Financial Time Series Forecasting [43].

LSTM neural networks outperform ARIMA models when dealing with complex sequential data with long-term dependencies, but require higher computational cost. Memory cells allow LSTM to adapt effectively to the unpredictable nature that financial markets often exhibit, whilst also maintaining a recollection of past patterns in the financial data.

ARIMA relies on traditional time series analysis techniques to capture the linear dependencies in datasets and it is a more computationally efficient method. However, a key drawback of ARIMA is its lack of ability to deal with non-linear patterns, this is an issue in forecasting stocks as they can exhibit volatile properties and often do not follow exactly linear trends.

ARIMA assumes linearity and that stationary data will be used, while LSTM excels at handling complex, nonlinear patterns. ARIMA outperformed LSTM in short-term predictions for NASDAQ data [44], but conflicting reports exist regarding their comparative performance [45]. LSTM's adaptability makes it better suited for forecasting in dynamic environments with frequent updates and real-world events, as it can include external features - such as news sentiment.

VARMA, like LSTM, excels in handling multivariate time series data, by capturing interdependencies between variables, making it flexible enough to model linear and non-linear relationships. However, VARMA requires similar data preprocessing like ARIMA - as it assumes stationary data.

Random Forest reduces overfitting and enhances the models robustness by combining the output of multiple decision trees. By averaging the forecasts from trees in the forest, Random Forest is, like VARMA, able to capture non-linear relationships within the data. However Random Forest is more difficult to interpret than ARIMA and LSTM, it can often operate as a "Black Box" [46] - where a system is only viewed as inputs and outputs without understanding of the internal workings.

Overall, whilst Random Forest and VARMA can offer superior predictive capabilities in certain situations, the lack of interpretability could hinder their utility in a practical sense - especially when understanding the models behaviour is key. At the cost of computational efficiency, the newer and more modern models, Random Forest and LSTM, perform well at handling lots of complex data with long-term dependencies.

When comparing models, ANOVA (ANalysis Of Variance) can be employed to assess the differences between multiple groups of data [47]. This can be useful as it allows for insight into which models significantly outperform others, allowing for informed decision making when selecting the most suitable models. Tukey's Tests can complement ANOVA by providing specific insights into where the differences [48], between model performance, identified lie; this post hoc test allows researchers to perform pairwise comparisons between all models to identify which individual pairs are significantly different.

Suitable error metrics to compare LSTM, ARIMA, Random Forest, and VARMA include MAPE (Mean Average Percentage Error) calculation and MSE (Mean Squared Error) [44]. While ARIMA and VARMA struggle with noisy data, LSTMs sensitivity to the selection of hyperparameters chosen poses a challenge; in light of this, Random Forests robustness to noise could make it a preferential choice for handling noisy financial data.

Real World implications In economic research, ARIMA is favoured for forecasting economic indicators, such as GDP and Inflation growth [49], due to its ability to capture linear trends; while, LSTM is applied to predict stock prices and apply credit risk assessment, due to its ability to deal with non-linear data which has long term dependencies; however, the modern RNN approach can extend to fields beyond time series data, like speech recognition and natural language processing.

While ARIMAs struggles with sudden market shifts due to its requirement for stationarity, LSTMs emergence has the potential to reshape economic theories, especially in identifying outliers and regime shifts [50], as it can be found to deviate from efficient/traditional market. ARIMA would instead be applied to measure more predictable movements, such as gradual declines in a businesses revenue or their production output.

VARMA performs well when handling multivariate time series data, and assists in analysing complex interactions among economic variables and to forecast macroeconomic indicators - helping policymakers to stabilise the economy [26].

Random Forest models find application in a wide range of fields including portfolio optimisation, credit scoring, and fraud detection, providing valuable feature importance scores for informed, data-driven decision making.

Hybrid models combining ARIMA, LSTM, VARMA, and Random Forest are being explored, aiming for more effective models by leveraging the strengths of each component [51]. These models are useful in finance, combining LSTMs adaptability and ARIMAs stability [52], resulting in more accurate forecasts.

3 Problem Analysis

3.1 Objectives

The goal whilst completing this paper is to compare numerous models capability at forecasting time series data on various index funds. There are multiple methods which have the ability to predict historical data, but for this study those that have been investigated in the prior section will be used. The following objectives can be set:

1. Produce ARIMA, VARMA, LSTM, and Random Forest models to forecast the market close price of selected index funds.

2. Evaluate the accuracy of each model and compare them.

The following sections of this report will describe the measures which were taken in order to fulfill the specified objectives.

3.2 Model Requirements

ARIMA (AutoRegressive Integrated Moving Average) The ARIMA model should be able to forecast the price that each Index will arrive at after a given period of time, based on the amount of data available to it

1. Stationarity - The model should ensure time series data is stationary by applying first-order differencing to it [19], checking that the statistical properties of the data do not change over time, meaning that the average, variance, and autocovariance of the series stay constant.

2. Order selection - Determining the order (p,d,q) of the ARIMA model is often based on ACF and PACF plots, or by doing a "grid search" [53] - evaluating the accuracy of each combination of orders for the model. For the orders: p represents the count of AutoRegressive terms, d represents the differencing required to achieve stationarity, and q represents the number of Moving Average terms.

VARMA (Vector AutoRegressive Moving Average) The VARMA model should be able to forecast the price that each Index will arrive at after a given period of time, using the collection of all index historical data and a base economic indicator (such as gold).

1. Stationarity - The model should process the historical data by applying differencing to ensure stationarity. This step ensures the statistical characteristics remain constant over time, including the mean and variance of the series [19].

2. Order selection - The model should be able to use a determined order (p,q) for its AutoRegressive and Moving Average portions. These values can be selected using AIC (Akaike Information Criterion) or BIC (Bayesian Information Criterion).

LSTM (Long Short-Term Memory) The LSTM model created should be able to forecast the path that the Close Price of the Index will take, after considering the past historical data.

1. Data Preprocessing - Cleaning the data by handling outliers and missing values. Then make sure all features have similar ranges, this helps the model converge to an optimal solution sooner during training.

2. Model Architecture Designing - Design the model architecture by determining the number of layers, units per layer, activation functions, and selecting a loss function and optimiser, along with any additional functions which would be tracked during training the model [54].

Random Forest The Random Forest generated should be able to aggregate to form an accurate prediction of the behaviour of each individual time series data.

1. Data Preprocessing - Clean the data to deal with missing values within the time series data.

2. Lagged Data - It should be ensured that the data possesses lagged features for the model to analyse, as the Random Forest uses feature importance to aid in its forecasts.

3. Terminating Criteria - A parameter and value should be decided on for the tree to terminate, such as maximum tree depth or a minimum node size.

3.3 Evaluation Requirements

Error Metrics The following error metrics are used to represent the difference between the actual values in a dataset and the predicted values produced by the models. These are useful for understanding how well the model is performing.

- MAE - Measures the average absolute difference between predicted and actual values, providing an indication of average error regardless of the direction.
$$MAE = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i| \tag{3.1}$$

- MSE - Calculates the average squared error between predicted and actual values, but particularly penalising larger errors - due to squaring. This causes MSE to be significantly more susceptible to outliers than MAE.
$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{3.2}$$

- RMSE - The square root of the MSE, offering a widely used, easily interpretable measure of error. It is, like MSE, sensitive to large outliers/errors.
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2} \tag{3.3}$$

- MAPE - The average percentage difference between predicted and actual values, its main limitation is it is sensitive to zeros in the actual values.
$$MAPE = \frac{100}{n}\sum_{i=1}^{n}|fracY_i - \hat{Y}_iY_i| \tag{3.4}$$

Statistical Testing When evaluating models, statistical analysis is useful to ascertain whether the different models yield distinct forecasting outcomes. This study compares the performance of different forecasting models using one-way ANOVA (ANalysis Of VAriance). ANOVA is particularly adept at capturing substantial deviations between the performance of groups; its null hypothesis states that there are no significant differences between groups, and an alternate hypothesis says there is a notable difference. In this study, ANOVA will be used with the aim of discerning any significant disparities between the produced forecasts. Additionally, Tukey's Tests will be employed as a post hoc test to determine which pairs of models show significant differences in performance.

4 Design & Implementation

4.1 Data Profiling

The forecasted data originates from various Index Funds, using the "Close" price - the final price that a stock/index traded at during the day. The Python library *yFinance* [55] was used for data collecion, capturing data between 03/01/2017 to 31/12/2019 from *Yahoo! Finance* [56]. This period was chosen due to market stability and upwards trends, due to low interest and the technology sector performing well and adding to overall market sentiment; the selected years also came prior to COVID-19 (which threw the markets into an unpredictable turmoil, as many were unaware of what was occurring) [57]. The library provides data for each day, which includes the variables: "Adj Close", "Close", "Open", and "Volume. The "Adj Close", or "Adjusted Close", accounts for corporate decisions made within the company to reflect the stocks true value; "Close" represents the days final price of a stock/index; "Open" is understood as the first price that the index trades at for the day, after the financial markets open, while "Volume" describes the total number of trades that are made for each selected financial asset throughout the day. Market closing price was selected as it is able to reflect the market sentiment - all relevant information for the whole day's activity - in one value, and also it is less likely to be affected by fluctuations throughout the day [58]. The index funds selected were: Dow Jones Industrial Average, NASDAQ, S&P 500, FTSE 250, Russell 2000, each exhibiting different behaviours due to the diverse range of markets. Most are US focused and listed on the New York Stock Exchange (Dow Jones Industrial Average, NASDAQ, Russel 2000 and S&P 500), whilst the FTSE 250 represents mid-cap companies on the London Stock Exchange. These indices provide a range of publicly traded companies, as Dow Jones Industrial Average, NASDAQ, and S&P 500 focus on large capitalization companies, and FTSE 250 represents mid-cap companies, whilst Russell 2000 represents numerous small-cap stocks. Figure 4.1 and Figure A.1 represent information describing the collected close data for the period 03/01/2017 to 31/12/2019.

Then after collecting the data, and storing it as a *Pandas DataFrame* [59], it is then split into training and test data, which is produced by a 70:30 split. The test data subset was therefore omitted from the training and then used

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 770 entries, 2017-01-03 to 2019-12-31
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   ^DJI    754 non-null    float64
 1   ^FTMC   758 non-null    float64
 2   ^GSPC   754 non-null    float64
 3   ^IXIC   754 non-null    float64
 4   ^RUT    754 non-null    float64
dtypes: float64(5)
memory usage: 36.1 KB
```

Figure 4.1: Information about collected data, after filtering for Close Data

to assess each model's accuracy on the data, in order to assess how close the predictions are to the real market events. Figure 4.2 shows plots of the close market prices, after being split into Training and Test Data.

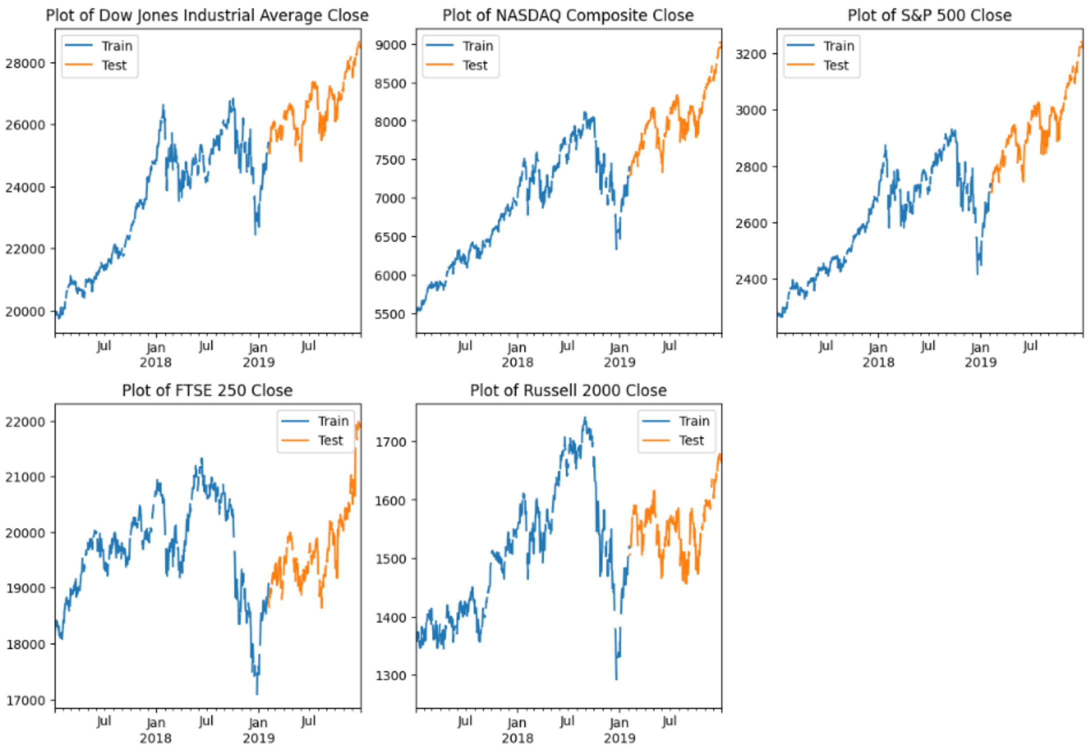

Figure 4.2: Plots of training and testing market close data

4.2 Stationary Data

For ARIMA and VARMA models to function optimally, data must be stationary. A stationary time series is one whose properties do not depend on the time at which the series is observed [20]; to ensure differencing the series can achieve stationarity Augmented Dickey-Fuller (ADF) tests may be used. ADF testing can reveal whether a unit root is present in the data, which would suggest non-stationarity (that the data's mean and variance changes with time). The null hypothesis of ADF tests is the time series has a unit root, and therefore is not stationary; the alternate hypothesis is that the data does not have a unit root and is stationary; a more negative *ADF Test Statistic* favours the alternate hypothesis. After generating the ADF Test Statistic, using the *adfuller* [60] function from the *statsmodels* library, it can be compared to critical values from the Dickey-Fuller Distribution, in this study's case the critical values at the significance level of 10%, 5% and 1% were chosen to observe[61]. If the ADF Statistic is found to be less than the selected critical value then the null hypothesis can be rejected, in favour of the view that the data is stationary. The p-value produced can be used to compare to the significance levels, again if the p-value is less than the chosen significance level then the null hypothesis can be rejected and the conclusion made that the data is stationary.

Table 4.1: ADF Test Statistics

Index ID	ADF Statistics	p-values	Critical Values		
			1%	5%	10%
ˆDJI	1.396	0.584	-3.439	-2.865	-2.569
ˆFTMC	-1.813	0.374	-3.439	-2.865	-2.569
ˆGSPC	-1.033	0.741	-3.439	-2.865	-2.569
ˆIXIC	1.115	0.709	-3.439	-2.865	-2.569
ˆRUT	2.039	0.270	-3.439	-2.865	-2.569

4.3 ARIMA (AutoRegressive Integrated Moving Average)

Model Training In order to train the model, after testing for stationary data and then differencing it, a range of values for the models order (p,d,q) is grid searched [53]. For this study the ranges chosen were: for p the range of 0 to 3 was chosen; for d the range of 0 to 2 was chosen, and for q the range from 0 to 3 was selected. Following this each of the chosen indices are then iterated through, extracting the training and test data for them from the train/test datasets. A nested loop is then utilised to loop over all possible combinations of (p,d,q) - within the chosen ranges. In the nested loop a model is created using the *statsmodels* library's *ARIMA* [62] and give it the current combination of order parameters, then fit this model to the training data. Next, Akaike Information Criterion (AIC) is performed to give an estimation of the prediction's error and an indication of the suitability of the current order. If the produced AIC value is lower, and therefore better, than the current recorded AIC score then the *bestOrder* values are updated to hold the new values. After the nested loop has run its course, bestOrder is then used for the current data to generate a forecast of how the ARIMA model expects the test data to behave. The forecast ranges between the start of the test data, and end of training data, to the end of the test data.

4.4 VARMA (Vector AutoRegressive Moving Average)

Data Manipulation For VARMA models to perform optimally they require stationary input data, so before training the model the data is tested by performing Augmented Dickey-Fuller (ADF) tests. Preprocessing techniques can then be applied based on the results of these tests, such as First-Order differencing.

Additionally, gold price data was incorporated into the dataset - to serve as an economic indicator; gold prices can be an indicator of economic health and inflation [2]. Gold is included in the forecast as it is a leading indicator for economic health; when gold prices rise it can signal potential economic trouble ahead, and the opposite is also true: that falling gold prices can indicate economic growth. It can also help the model understand how changes in inflation/interest rates will affect both gold and stock prices.

Model Training With the use of the *statsmodels* library, an instance of the *VARMAX* [63] model is created with the training data, the combination of the original index data and gold prices. The model is then fit to the training data, through which the VARMAX model estimates suitable parameters for the provided training data. After fitting the model, forecasts are then generated for future time steps equal to the length of the testing data - which is currently unseen data, so that the model was not training on data it already has knowledge of. The default orders (p,q) are used for the VARMAX model, with both p and q being set to 1; these parameters are used to determine the AutoRegressive and Moving Average orders.

4.5 LSTM (Long Short-Term Memory)

Data Transformation To train the LSTM model effectively, data quality is paramount. Missing values are removed from the data, to ensure data consistency and facilitating proper pattern learning. Then the data is scaled, preventing issues like "the exploding gradient problem". Using *MinMaxScaler* [64], from *sklearn*, the data is normalised, within the range -1 to 1, to avoid the risk of certain data features dominating the models training. A 70:30 train-test split is then performed to provide ample training data for the model whilst still maintaining enough to sufficiently test the model. This split avoids having a testing set which is too small to be confident that the model's evaluation is accurate, while minimising overfitting when compared to other splits, and also enhances training times.

Sequential data is key for the LSTM models function, as past values and market trends influence future behaviour of the financial markets. Sequences must be labeled to discern the series of values, whether for use in training or testing, the *"createSequencing"* function uses the next data point as the label, then outputs the original Close Prices data and the assigned labels for the data.

Then each dataset undergoes missing value handling, substituting the missing data for the median value of the respective column to preserve the data distribution. Using a mean value could risk distorting the data if there are significant outliers (which can be the case with unpredictable stocks), while the median is more stable and less easily influenced by changes in the data. *PyTorch Data Loaders* are then used to transform the training and test data into iterable objects, which can be processed by the model for use in mini-batch training [65]. The training data batches are shuffled to ensure the model can witness a diverse range of example values in each epoch of its training, preventing the model from memorising the sequence order, encouraging it to learn patterns in the data. Although the train data is shuffled, the test data isn't shuffled to ensure the evaluation of the models performance on the testing set is consistent across numerous runs.

Hyperparameters Hyperparameters are pivotal in defining the architecture of LSTM models and influencing the way they are trained. They influence key aspects of the model, such as: input and output sizes, hidden layers, sequence length, epochs, batch size, learning rate, optimiser choices, and the criterion used [54]. Each hyperparameter influences the models ability to capture patterns in the data and the computational efficiency of the model. The number of columns the input data had was used to determine the input and output size, along with a hidden size of 64 - to capture complex patterns within the data while remaining efficient.

Two layers were used in this LSTM model, to maintain a balance between performance and complexity. The LSTM model utilises a sequence length of 10, considering the past 10 time steps to forecast the next, allowing the model to capture a balance of short and long term dependencies. The LSTM model trains for 100 epochs, with a batch size of 64 for each epoch. The ADAM (ADAptive Moment estimation) optimiser is used to minimise the loss of the model, with a learning rate defined at 0.001, and the model uses MSE (Mean Squared Error) when aiming to minimise errors 3.3. For further description regarding hyperparameters see Appendix A.2.

Model Design The model is initialised using the previously described hyperparameters to define the architecture of the LSTM model. The Input Size, Hidden Size, and Num Layers are used to specify how the model processes the input data within the *PyTorch LSTM model* [54]. Then a linear layer translates the hidden layer neurons to output layer neurons. When defining the models forward pass method a *PyTorch tensor* is input, this tensor contains the input sequences. The initial states are defined with zeros for the hidden state, which represents all information from the input step to the current time step, and the cell state, which is used to transport information across time steps. After initialising the hidden and cell states the LSTM layer is then applied to these, which is then used to output the final hidden state; then a linear layer translates the LSTM layer to be output.

Model Training In order to train the model, first an empty list is initialised to store the training loss values for each epoch, which is used to track the models progress through epochs. For each epoch a *running_loss* variable is initialised, which tracks the model's loss for the current epoch. After doing this an inner loop is used to iterate over mini-batches of data from the earlier defined *train_loader*. Then the recorded optimiser gradients are cleared, ensuring that only the gradients for the current batch are being considered whilst updating the model parameters. Then the model is fed the input data so it produces the predicted output values. The loss between the predictions and the ground truth is then calculated - using the MSE criterion, described in Evaluation Requirements 3.3. An optimisation step is made to update the model's parameters, based on computed gradients from a backward pass.

4.6 Random Forest

Data Feature Creation The data used for the Random Forest model would be the same historical data used for all preceding models surrounding the index funds, consisting of the Close Price and the Date associated with this price. After cleaning the data of all non-values, it must be ensured that the data has lagged features. Lagged features can be created by looping through each of the indices and creating additional columns in the *Pandas DataFrame* containing the historical data, these additional columns can be assigned to each lag, i.e. if there were three lags then there would be three more columns than there originally were for one index fund. The values in each of these columns are the lagged values for the original index, shifted up by the lag count. The inclusion of lagged features allows the model to learn from historical patterns and relationships between the past and present index funds

Model Design The Random Forest model that will be being used for this study is imported from the library *sklearn*, as *RandomForestRegressor* [66]. This model was chosen as it is specifically designed for performing regression tasks, rather than including the alternative for Random Forest (classification). The model is provided with the hyperparameters for the number of estimators and a random state. A Random State is used as it allows for reproducible, consistent results from the model, which prevents the models performance from varying when evaluating and measuring the results after multiple runs; it also allows for this study to be re-run and produce the expected results. A Random State of *1* was provided to the model in this instance. The number of estimators hyperparameter dictates the number of decision trees used in the Random Forest, with a higher number of estimators generally improving the performance of the model (up to a point). With less estimators being used a Random Forest would prove to be less computationally expensive and hence able to train faster, although a Random Forest with more decision trees would be expected to perform better at generalising to analyse unseen data, and also be more robust. After trial and error with different amounts of estimators, a count of 100 estimators was decided on. This value was selected as additional estimators did not significantly improve the predictive power and accuracy of the model, when measuring using MAPE.

Model Training After ensuring the data has the correct features, the historical data is split into training and test sets in a 70:30 split. After splitting the data, it is then again split into X and Y portions, with the X containing the dates associated with the target variable, the Close Price, that is stored in Y. The model is then initialised with the aforementioned hyperparameters and, for each index fund, fit to the train data; then the forecasts for the length of the test period are generated. After that, these can be evaluated using RMSE and MAPE.

5 Results & Evaluation

5.1 Model Forecasts

ARIMA (AutoRegressive Integrated Moving Average) The forecasts that the ARIMA models produced were not accurate, with most remaining at a very similar price from the final value in the training set for the time period of the test set. This indicates that ARIMA may not be suitable for this time period and time series data. Table 5.1 shows the evaluation metrics for the ARIMA model. Figure 5.1 shows plots of the training data, testing data, and the forecasts produced by ARIMA.

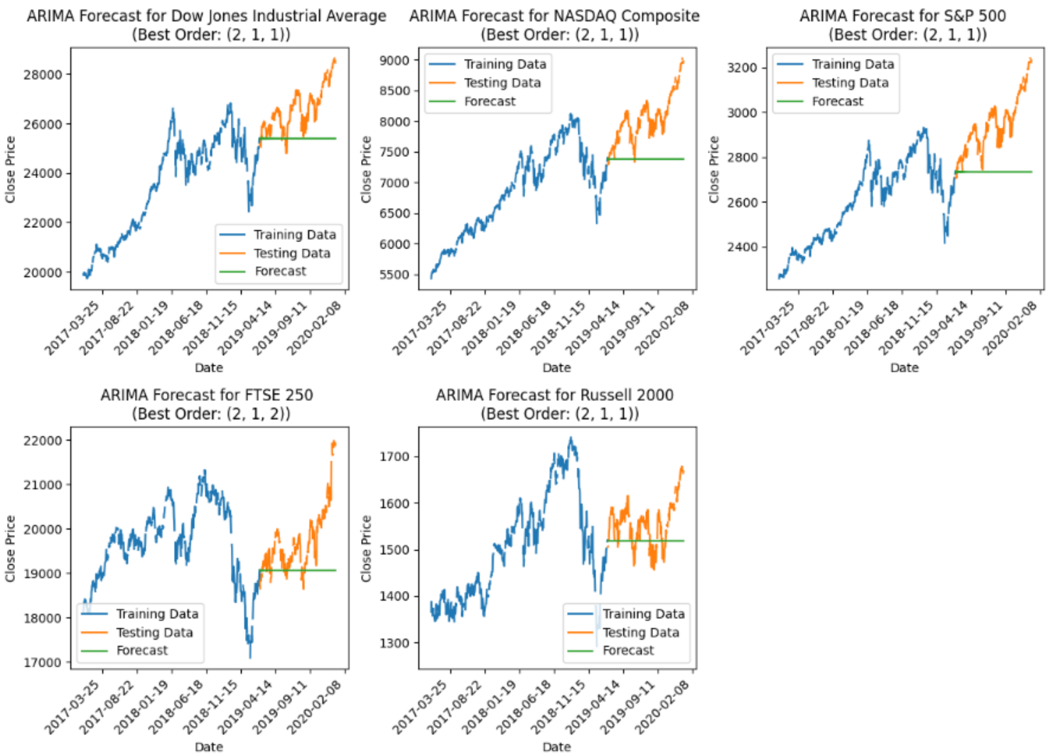

Figure 5.1: ARIMA Forecasts

Table 5.1: ARIMA model average error metrics

Model	Average MAE	Average MSE	Average RMSE	Average MAPE (%)
ARIMA	570.94	749636.27	555.50	5.23

VARMA (Vector AutoRegressive Moving Average) The forecasts that the VARMA model produced were inaccurate, and unexpectedly performed worse than ARIMA. The forecasts showed a smoothed gradient, which could be attributed to the inclusion of Gold data in this model as an economic indicator, but to determine if this is true would require further investigation. VARMA may not be suitable for this time period and time series data. Table 5.2 shows the evaluation metrics for the VARMA model. Figure 5.2 shows plots of the training data, testing data, and the forecasts produced by VARMA.

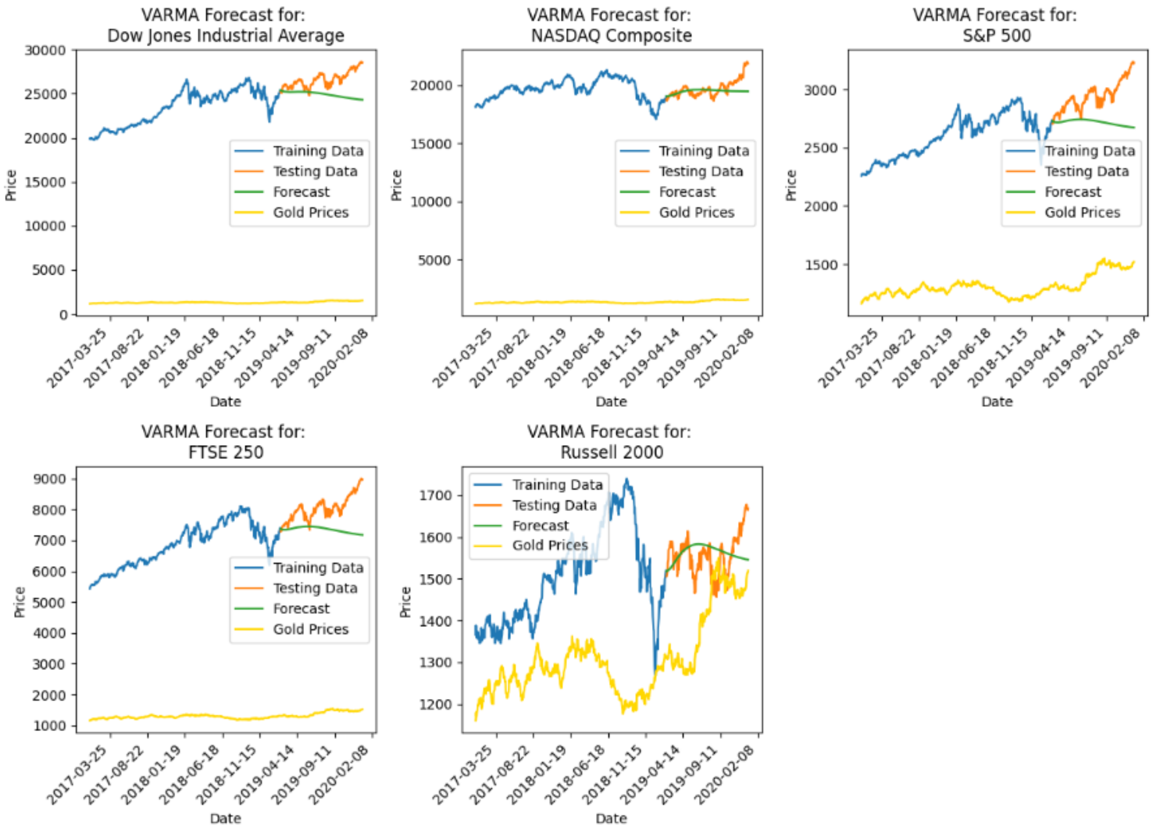

Figure 5.2: VARMA Forecasts

LSTM (Long Short-Term Memory) For some of the indices LSTM proved effective at forecasting the pattern the close price took, however for others

Table 5.2: VARMA model average error metrics

Model	Average MAE	Average MSE	Average RMSE	Average MAPE (%)
VARMA	634.45	1072572.06	777.36	NaN

it exhibited gross inaccuracy - this can be observed in Figure 5.3. The average evaluation metrics for these predictions are shown in Table 5.3.

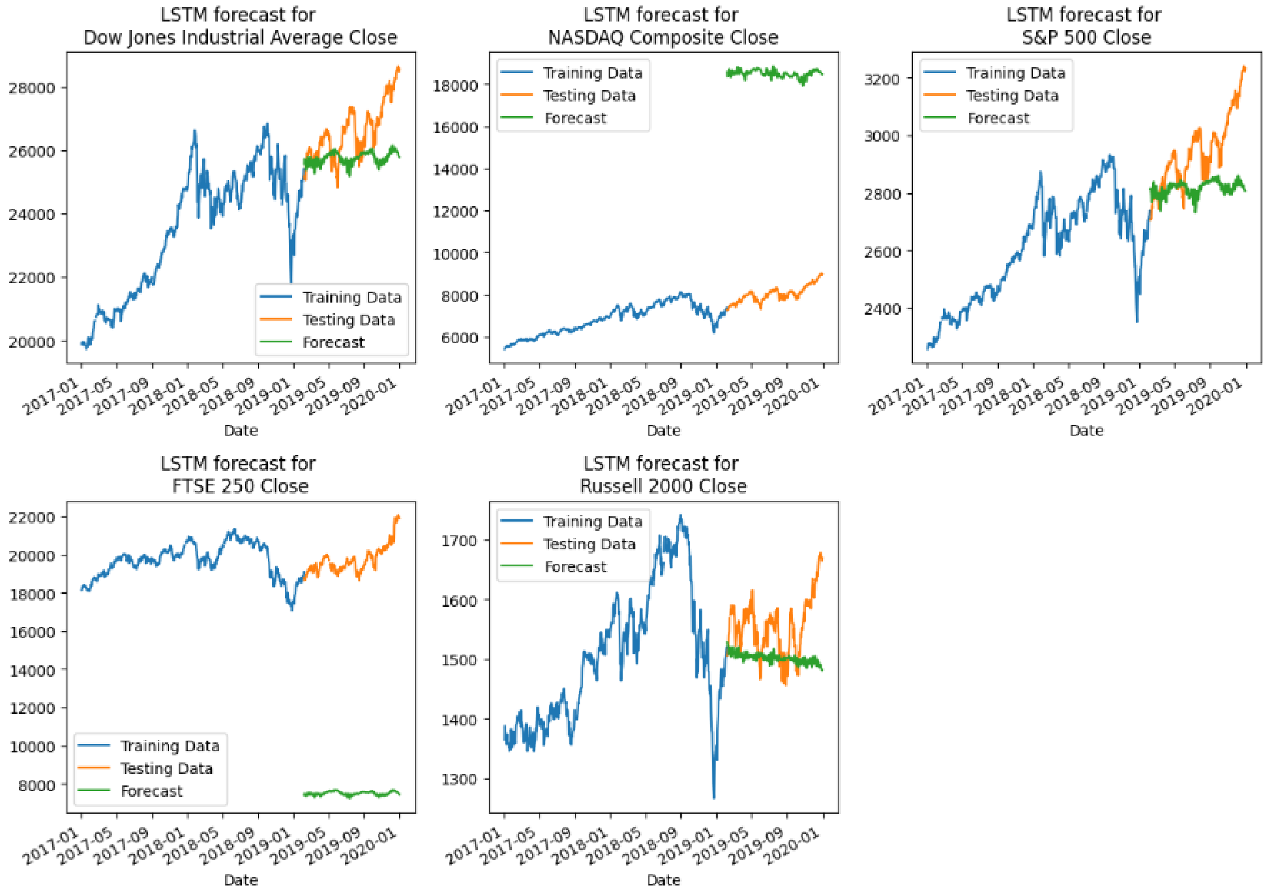

Figure 5.3: LSTM Forecasts

Table 5.3: LSTM model average error metrics

Model	Average MAE	Average MSE	Average RMSE	Average MAPE (%)
LSTM	4916.59	54376342.08	4905.77	42.19

Random Forest Random Forest proved very effective at analysing the patterns the Close Price data would take. The predictions Random Forest made were quite accurate and performed the best of the four models. The evaluation metrics for this model are shown in Table 5.4, along with the plotted forecasts in Figure 5.4.

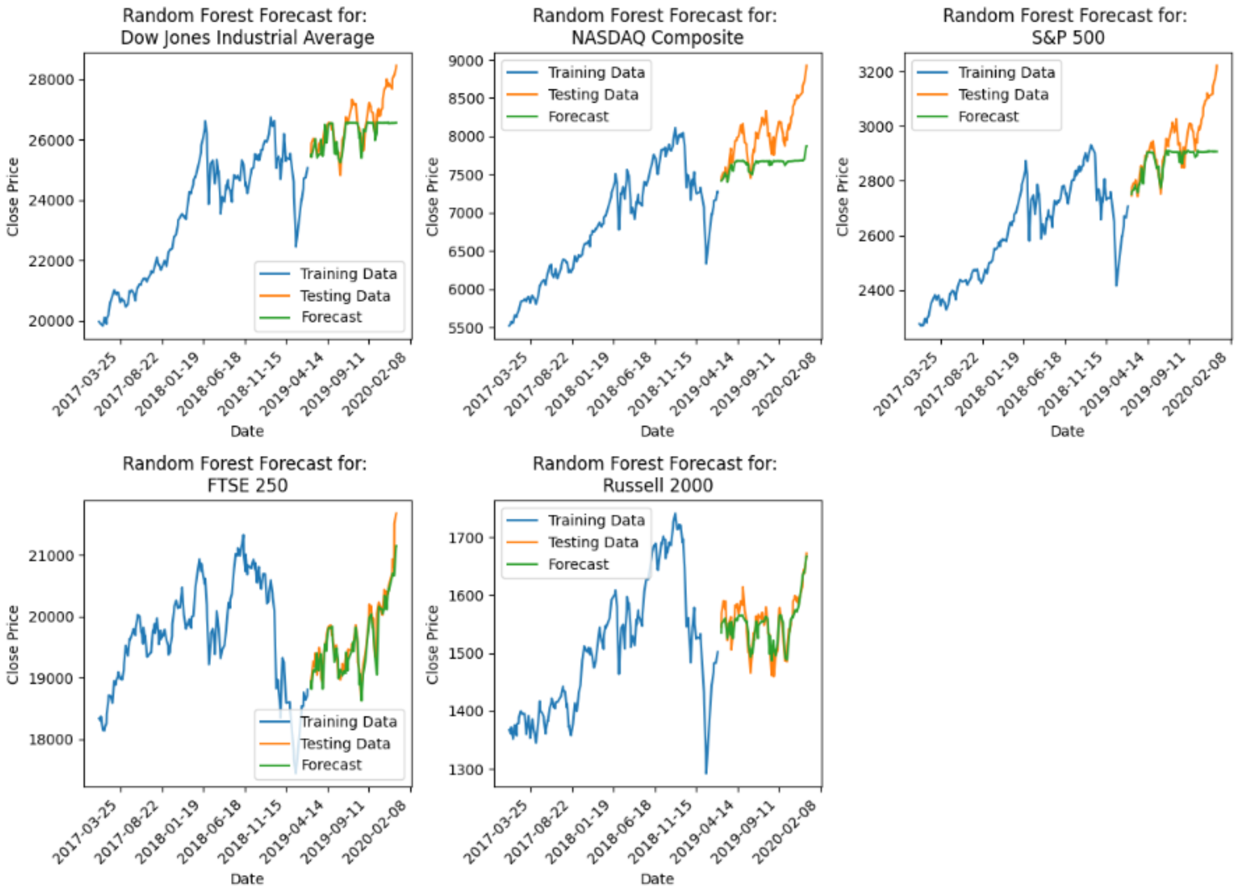

Figure 5.4: Random Forest Forecasts

Table 5.4: Random Forest model average error metrics

Model	Average MAE	Average MSE	Average RMSE	Average MAPE (%)
Random Forest	222.65	145100.34	195.56	2.14

5.2 Model Comparison

After comparing the effectiveness of ARIMA, VARMA, LSTM, and Random Forest at forecasting the market close price of the chosen indices, the average of the collected error metrics can be observed in Table 5.5. These values given indicate that overall, Random Forest is most proficient at forecasting the price of the index funds. While the LSTM model made some accurate predictions, LSTM experienced shortfalls on NASDAQ Composite and FTSE 250 which cause it to, on average, perform worse than the Random Forest model (the individual accuracy's can be seen in Table A.1). ARIMA and VARMA can be seen to perform with similar accuracy to each other, indicating that there is not huge benefit to increasing Computational Cost by including additional data. See below in Table 5.5 the Average RMSE, Average MSE, Average MAE, Average MAPE, for each model. With the accuracy that Modern Techniques produce, the advancement and improvement in mathematics, and modelling since the origination of Traditional Methods like ARIMA, can be observed.

Table 5.5: All average error metrics

Model	Average MAE	Average MSE	Average RMSE	Average MAPE (%)
ARIMA	570.94	749636.27	555.50	5.23
VARMA	634.45	1072572.06	777.36	NaN
LSTM	4916.59	54376342.08	4905.77	42.19
Random Forest	222.65	145100.34	195.56	2.14

Comparison of the error can be achieved by performing ANOVA (ANalysis Of VAriance) on the RMSE values for each model. Table 5.6 shows that the p-value is above the significance level 0.05, which means that there is not sufficient evidence to reject the null hypothesis and conclude that there are significant differences between the RMSE of each model. This means that, in terms of RMSE there is not substantial differences between the performance of the models. The *f_oneway* [67] function was used from the *scipy* library to perform ANOVA.

Table 5.6: F-Statistic and P-Value, for RMSE of all model forecasts

F-Statistic	p-value
2.5484950694296344	0.09232764598308445

The actual testing data was compared against the forecasts from ARIMA, VARMA, LSTM and Random Forest using ANOVA with a significance level of 0.05 was maintained - if the ANOVA tests produce a p-value higher than the significance level then the null hypothesis is rejected and a conclusion is drawn that there is significant variation between the forecasting models.

In table 5.7, all the index funds show extremely small (or zero) p-values; this indicates strong evidence against the null hypothesis and that there

Table 5.7: F-Statistics and P-Values, for forecasts on each index fund

Index	F-Statistics	p-values
Dow Jones Industrial Average	186.28	4.32×10^{-89}
FTSE 250	24243.21	0.0
S&P 500	239.11	1.63×10^{-103}
NASDAQ	106683.30	0.0
Russell 2000	49.57	1.03×10^{-33}

are differences between at least one pair of forecasts in each case of the forecasts produced. Some forecasts of the selected index funds exhibit a very large F-statistic (i.e. NASDAQ), indicating a large performance difference among the forecasting models for these funds.

After establishing that at least one pair of results show significant differences, post hoc Tukey's tests, using the *statsmodels* library's *pairwise_tukeyhsd* [68], are performed on all the forecasts of each index, to produce a multiple comparison of means. These found that when forecasting the Dow Jones Industrial Average, all models significantly differ from each other - except for LSTM and Random Forest. The tests showed that the produced LSTM and VARMA predictions resembled each other, when forecasting the NASDAQ Composite fund. For the S&P 500, it can be found there is no significant difference between LSTM and RF, or between RF and VARMA; although for the S&P 500, ARIMA, LSTM, and VARMA significantly differ from each other. Similarities were found for the predictions of FTSE 250 from RF and VARMA, with all other models showing significant differences. The opposite was true when forecasting the Russell 2000 Index, with RF and VARMA differing significantly, but no further difference between ARIMA, LSTM, and VARMA. Appendix A.4 has more details about Tukey's tests.

5.3 Discussion

The difference between modern and traditional methods of financial forecasting is made evident by their approaches to capturing market dynamics. The selected modern techniques (LSTM and Random Forest) quite accurately trace the fluctuations in the market close data, whereas traditional methods (ARIMA and VARMA) tend to produce a smoother prediction which obscures the market movements - as shown in Figure 5.1 and Figure 5.2. This disparity may be attributed to, the shared characteristic of ARIMA and VARMA, Moving Average components [69], which are designed to minimise short-term fluctuations- but further investigation is required to determine if this is true.

In contrast, LSTMs and Random Forests ability to more accurately capture market behaviour is made clear in their forecasts, seen in Figure 5.3 and Figure 5.4, which exhibit closer predictions. Even though these modern models are more computationally complex than their traditional counterparts, the enhanced predictive power and the ability to assign feature importance justifies modern methods being adopted to forecast financial data. Random Forest excels at identifying and prioritising features based on their assessed importance, enabling a deeper understanding of the underlying drivers of market behaviour [70]. Similarly, LSTM models, after some manipulation, can reveal the assessed importance that each variable within a model holds, further adding to the forecasting process.

Amongst the traditional methods, VARMA showed notable, unexpected, performance over LSTM; this could be attributed to the inclusion of gold prices 5.2. The model would have been able to use the price of gold as an economic indicator, reflecting market sentiment and global economic stability, which may provide it with an advantage over ARIMA when forecasting - due to its multivariate capabilities.

ANOVA tests on the predicted values highlight the significant difference Random Forest possessed over other models5.7. However, when using ANOVA on the RMSE of the forecasts it found the errors produced by each model were not significantly dissimilar5.6. This can suggests that the individual shortcomings of the models were balanced out, after averaging the error across all of the forecasted values.

Despite their computational efficiency, traditional methods like ARIMA and VARMA face limitations, due to their requirement for stationary data. Differencing the data to achieve stationarity can pose the risk of removing important features, which may then compromise the forecasting accuracy of these methods. Hybrid approaches combining multiple models offer promise when addressing the limitations of individual models, by leveraging the strengths of the sole approaches to enhance predictive performance [51].

The overall findings highlight the clear advantages of modern approaches when capturing the complexities of market behaviour, and that the exceptional benefits of modern methods extend beyond strictly their predictive accuracy. Despite the increased computational complexity associated with modern methods, modern techniques ability to provide further insight into the market behaviour, through better tracing trends and identifying feature importance [70], proves invaluable. With the ever increasing complexity of financial markets, the adoption of modern forecasting techniques will become imperative for those seeking to navigate particular uncertainty and benefit from emerging, previously hidden, opportunities.

6 Conclusion

After multiple forecasting models were applied to selected index fund data, the ARIMA, VARMA, LSTM, and Random Forest models were evaulated using MAPE and RMSE, and then compared using ANOVA - to ascertain any significant differences between the predictions they produced. Upon finding that there were significant differences, the forecasts were also subjected to Tukey's Tests to identify which models had differences. These tests revealed that the Random Forest model demonstrated better performance when compared to other models; this would suggest that a modern approach, Random Forest, is more preferential over a traditional method. Further ANOVA investigation comparing the RMSE of the models found that they performed comparably, this discrepancy may be due to the balancing effect of combining individual errors to calculate RMSE. The advantage of Random Forest is shown by its ability to imitate market behaviour.

For future studies, enhancements could be made to improve prediction accuracy, including:

- Hybrid Methods - Combining models, such as ARIMA with LSTM, could yield more accurate predictions. Hypbrid approaches can integrate complementary features to enhance performance.

- Consideration of external factors - Introducing external factors, like news sentiment or economic indicators, could increase the predictive power by providing further insights into market activity.

- Additional Data - More diverse/relevant data can enhance the accuracy of the forecasting models; this can include, but is not limited to, macroeconomic indicators or industry specific data.

- Different error metrics/evaluation methods - Exploring other evaluation metrics could potentially provide a more comprehensive understanding of the models performance and behaviours.

- Market strategies - Investigate market strategies for buying and selling to maximise profit, like "Buy and Hold", or "Pair Trading".

- Feature Importance - Investigating the feature importance in models can lead to a better understanding of the models performance.

A Appendix

A.1 Data collection

Ticker	^DJI	^FTMC	^GSPC	^IXIC	^RUT
count	754.000000	758.000000	754.000000	754.000000	754.000000
mean	24397.198444	19681.232461	2703.161458	7201.443021	1520.138517
std	2234.141749	826.223824	227.623908	825.506462	100.202197
min	19732.400391	17090.500000	2257.830078	5429.080078	1266.920044
25%	22420.784668	19161.099609	2502.632507	6461.864868	1433.012543
50%	24879.439453	19692.950195	2723.530029	7329.705078	1525.225037
75%	26061.084473	20228.350098	2879.137390	7869.020142	1580.892548
max	28645.259766	22059.000000	3240.020020	9022.389648	1740.750000

Figure A.1: Description of collected Close Data

A.2 LSTM Hyperparameters

Input Size: This hyperparameter is used to define how many input dimensions the data, i.e. how many columns the data has. In the case of this study, the number of columns are used - allowing the model to train on all the selected index fund data at once. This is used to determine how many input neurons the LSTM model uses, directly influencing the architecture of the model.

Output Size: For output size the aim is to be able to make predictions for each individual index, so it should be ensured that the model outputs the same number of predictions as the number of inputs given to it. When performing multivariable forecasting the Output Size and Input Size often match. This hyperparameter directly affects the architecture of the model, as it states that the model requires an output neuron for each index.

Hidden Size: The hidden size hyperparameter is used to represent how the data is represented internally, within the LSTM model. When choosing a hidden size it is important to consider the computational cost of the model, as when hidden size is increased so is the cost. However, having a larger hidden size allows for the model to be able to identify more complex patterns within the data. This study uses a hidden size of 64, as it is not overly large but is still sufficient to capture the complex patterns in the data.

Number of Layers: This hyperparameter is labelled very accurately, as it just defines the number of LSTM layers which are applied on top of each other within the model. A higher number of layers causes the model to become a deeper neural network - increasing the models capacity to learn but also causes it to become more computationally expensive. The choice of two LSTM layers allows the model to learn complex representations whilst also staying not excessively computationally expensive, maintaining a balance in complexity and performance. A choice of a lower number of layers also reduces the risk of overfitting, however this is already low due to the use of a large dataset.

Sequence Length: This hyperparameter is used to manipulate the data for effective model training, as previously described in 4.3.1. It represents the number of historical observations which are considered with each prediction. Longer/larger sequences allow the model to identify more long term dependencies within the data, however this again comes with the tradeoff of being more computationally expensive. This study uses a sequence length of 10, meaning it will consider 10 past time steps to predict the next time step, capturing a balance of short and long term dependencies.

Epochs: Epochs specify the number of times that the dataset is passed through the LSTM model whilst training it. With each epoch the model updates its parameters to improve over time, meaning that a higher number of epochs gives the model more opportunities to learn - however also increases the risk of overfitting. There is also the possibility of the model reaching an optimal solution early in training, and then there not being much reduction in loss from more training. 100 epochs is a common choice in training LSTM models and permits the model to undergo enough iterations in training to accurately converge.

Batch Size: In training the model does not analyse each data point individually, but instead selects batches of the given Batch Size and processes multiple data points simultaneously. This is useful in this study's case as large datasets are used and it speeds up the training of the model. It also smooths the optimisation of the model, as each batch provides a different perspective on the data which encourages the model to learn more general patterns. The LSTM model used for this study uses a batch size of 64 data points.

Learning Rate: This hyperparameter defines the step size which the model parameters are updated whilst training, using the selected optimisation algorithm - ADAM. It is important to choose an appropriate learning rate, in order for the model to converge towards an optimal solution whilst both avoiding overshooting and not settling for a less optimal solution by getting stuck in a local minima. The learning rate is defined at 0.001 in this study, providing ADAM with a moderate step size and allowing efficient convergence.

Optimiser: For the optimiser of the models parameters, this study uses ADAM (ADAptive Moment estimation) to minimise the loss of the model. ADAM is an optimiser that is used to update the network weights during training to minimise loss. It combines AdaGrad and RMSProp and this causes faster convergence and less tuning of hyperparameters. It computes adaptive learning rates for parameters by keeping track of the exponentially decaying average of past gradients and their squares. AdaGrad (ADAptive Gradient algorithm) is an optimiser used to adjust the learning rate in training based on parameters. It scales the learning rates inversely proportional to the square roots of sum of the squared gradients for each param. The idea is that it performs larger updates for infrequent parameters and smaller updates for those which are more common - this makes it useful for sparse data (where some parameters may be updated less than others). Faster convergence is caused by a reduced learning rate for frequent parameters and increased for less frequent ones. A drawback of AdaGrad is that if the learning rate becomes too small, which happens over time, then it can cause slow convergence. RMSProp (Root Mean Squared Propagation) is an optimiser that addresses the learning rate issue from AdaGrad - by using the aforementioned exponentially decaying average of past gradients and their squares - in Adam. This lets it focus on more recent gradients instead of all past gradients, lessening the issue of diminishing learning rates. However, like AdaGrad, RMSProp still has the same issue of having a fixed learning rate schedule - which isn't ideal for optimisation.

Criterion: MSE loss is used as the loss function when training and testing the LSTM model, aiming to minimise the MSE. Mean Squared Error is commonly used in financial time series forecasting.

A.3 Model Error Metrics

In Table A.1 are the individual error metrics for each model, on each index. The Index column contains the symbol for each index, Dow Jones Industrial Average (DJI), FTSE 250 (FTMC), S&P 500 (GSPC), NASDAQ (IXIC), and Russell 2000 (RUT).

Model	Index	MAE	MSE	RMSE	MAPE(%)
ARIMA	DJI	1228.76	2180764.22	1205.09	4.53
	FTMC	701.38	927083.80	661.55	3.45
	GSPC	212.12	59205.02	211.49	7.05
	IXIC	664.11	577532.37	661.61	8.07
	RUT	48.34	3595.93	37.76	3.05
VARMA	DJI	1228.76	2180764.22	1205.09	4.53
	FTMC	701.38	927083.80	661.55	3.45
	GSPC	212.12	59205.02	211.49	7.05
	IXIC	664.11	577532.37	661.61	8.07
	RUT	48.34	3595.93	37.76	3.05
LSTM	DJI	1196.16	2064356.45	1159.36	4.41
	FTMC	10751.24	115741098.39	10751.24	134.22
	GSPC	198.87	53345.51	197.02	6.61
	IXIC	12390.32	154019623.46	12390.32	62.78
	RUT	46.34	3286.58	30.91	2.93
Random Forest	DJI	474.65	433415.61	415.33	1.75
	FTMC	154.56	49943.26	97.86	0.78
	GSPC	74.95	11359.28	69.10	2.46
	IXIC	394.49	230425.12	391.30	4.78
	RUT	14.60	358.46	4.21	0.94

Table A.1: All average error metrics

A.4 Tukey's Tests

Table A.2 presents the results of the post hoc Tukey's Tests performed across the models forecasts on selected index funds. The column headings represent a different property found through the testing. The "Index" column contains the ticker for each index which was being forecast. "Model 1" and "Model 2" describe the pair of models which are being examined, and "Mean Difference" displays the mean difference between the two groups being compared. Adj-P contains the adjusted p-values, taking into account multiple comparisons to the original p-value for the Family-Wise Error Rate (FWER); the FWER for each test was set at a significance level of 0.05. "Lower" and "Upper" represent the lower and upper bounds of the confidence interval for the produced "Mean Difference". "Reject H_0" contains a Boolean value which indicates whether the null hypothesis, which indicates a significant difference between "Model 1" and "Model 2", can be rejected; "True" suggests a rejection, whereas "False" means there is no statistically significant difference. The Index column contains the symbol for each index, Dow Jones Industrial Average (DJI), FTSE 250 (FTMC), S&P 500 (GSPC), NASDAQ (IXIC), and Russell 2000 (RUT).

Index	Model 1	Model 2	Mean Difference	Adj-P	Lower	Upper	Reject H_0
DJI	ARIMA	LSTM	-926.109	0.0	-1023.9549	-828.2632	True
	ARIMA	RF	-707.7829	0.0	-805.6288	-609.937	True
	ARIMA	VARMA	-744.4566	0.0	-842.3025	-646.6108	True
	LSTM	RF	218.3261	0.0	120.4803	316.172	True
	LSTM	VARMA	181.6524	0.0	83.8065	279.4983	True
	RF	VARMA	-36.6737	0.7677	-134.5196	61.1721	False
IXIC	ARIMA	LSTM	-407.1179	0.0	-467.7932	-346.4425	True
	ARIMA	RF	11015.3632	0.0	10954.6879	11076.0386	True
	ARIMA	VARMA	-427.0295	0.0	-487.7049	-366.3542	True
	LSTM	RF	11422.4811	0.0	11361.8058	11483.1565	True
	LSTM	VARMA	-19.9117	0.8316	-80.587	40.7637	False
	RF	VARMA	-11442.3928	0.0	-11503.0681	-11381.7174	True
GSPC	ARIMA	LSTM	-131.2218	0.0	-147.9635	-115.1078	True
	ARIMA	RF	-112.3213	0.0	-128.4352	-96.2073	True
	ARIMA	VARMA	-128.4889	0.0	-144.6029	-112.3749	True
	LSTM	RF	18.9005	0.0141	2.7865	35.0145	True
	LSTM	VARMA	2.7329	0.9718	-13.3811	18.8469	False
	RF	VARMA	-16.1676	0.0489	-32.2816	-0.0536	True
FTMC	ARIMA	LSTM	-53.4536	0.3168	-133.8856	26.9785	False
	ARIMA	RF	-12089.0643	0.0	-12169.4964	-12008.6323	True
	ARIMA	VARMA	-375.4439	0.0	-455.8759	-295.0119	True
	LSTM	RF	-12035.6108	0.0	-12116.0428	-11955.1787	True
	LSTM	VARMA	-321.9903	0.0	-402.4224	-241.5583	True
	RF	VARMA	11713.6204	0.0	11633.1884	11794.0525	True
RUT	ARIMA	LSTM	0.8507	0.8686	-8.3125	4.3511	False
	ARIMA	RF	-43.9834	0.0	-32.888	-20.2244	True
	ARIMA	VARMA	-39.2443	0.0	-45.5761	-32.9125	True
	LSTM	RF	-24.5755	0.0	-30.9073	-18.2437	True
	LSTM	VARMA	-37.2636	0.0	-43.5954	-30.9318	True
	RF	VARMA	-12.6881	0.0	-19.0199	-6.3563	True

Table A.2: Tukey's Test results

Bibliography

[1] M. von Wallis and C. Klein, 'Ethical requirement and financial interest: A literature review on socially responsible investing,' *Business Research*, vol. 8, 2015. [Online]. Available: https://doi.org/10.1007/s40685-014-0015-7.

[2] W. Yamaka, 'Understanding the nexus between emerging stock market volatility and gold price shocks,' in Springer, 2023.

[3] M. Mallikarjuna and R. Rao, 'Evaluation of forecasting methods from selected stock market returns,' *Financial Innovation*, 2019.

[4] Z. Rezaee, S. Aliabadi, A. Dorestani and N. Rezaee, 'Application of time series models in business research: Correlation, association, causation,' *Sustainability*, 2020.

[5] C. L. Giles, S. Lawrence and A. C. Tsoi, 'Noisy time series prediction using recurrent neural networks and grammatical inference,' *Machine Learning*, 1988.

[6] P. Campbell, 'What is financial forecasting, why is it important, and how to properly conduct financial planning and forecasting,' 2021. [Online]. Available: https://www.paddle.com/resources/financial-forecasting.

[7] A. Ruparelia, 'What is financial forecasting, and how to conduct it?' *Financial Management*, 2023. [Online]. Available: https://agicap.com/en/article/financial-forecasting/.

[8] A. Tuovila, 'Forecasting: What it is, how it's used in business and investing,' *Technical Analysis Basic Education*, 2024. [Online]. Available: https://agicap.com/en/article/financial-forecasting/.

[9] H. Wasserbacher and M. Spindler, 'Machine learning for financial forecasting, planning and analysis: Recent developments and pitfalls,' *Digit Finance*, 2022.

[10] T. C. Mills, 'Forecasting financial time series,' in *The Oxford Handbook of Economic Forecasting*, Oxford Handbooks, 2011.

[11] M. Zellner, A. E. Abbas, D. V. Budescu and A. Galstyan, 'A survey of human judgement and quantitative forecasting methods,' *R. Soc. Open Sci.*, 2021.

[12] S. Tenny, J. M. Brannan and G. D. Brannan, 'Qualitative study,' *StatPearls [Internet]*, 2022. [Online]. Available: https://ncbi.nlm.nih.gov/books/NBK470395/.

[13] L. A. Maggio, J. L. Sewell and J. Artino A. R., 'The literature review: A foundation for high-quality medical education research,' *Journal of Graduate Medical Education*, 2016.

[14] J. McCambridge, J. Witton and D. R. Elbourne, 'Systematic review of the hawthorne effect: New concepts are needed to study research participation effects,' *Journal of Clinical Epidemiology*, 2014.

[15] W. Kenton, 'Quantitative analysis (QA): What it is and how it's used in finance,' *Quantative Analysis*, 2023. [Online]. Available: https://www.investopedia.com/terms/q/quantitativeanalysis.asp.

[16] J. Smith and H. Noble, 'Bias in research,' *Evidence-Based Nursing*, 2014.

[17] K. Nikolopoulos, 'Forecasting with quantitative methods: The impact of special events in time series,' *Applied Economics*, 2010.

[18] M. Vogl, 'Quantitative modelling frontiers: A literature review on the evolution in financial and risk modelling after the financial crisis (2008-2019),' *SN Business Economics*, 2022.

[19] T. Alghamdi, K. Elgazzar, M. Bayoumi, T. Sharaf and S. Shah, 'Forecasting traffic congestion using ARIMA modeling,' in *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2019, pp. 1227–1232.

[20] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*. OTexts, 2018. [Online]. Available: https://otexts.com/fpp2/.

[21] A. Hayes, 'Autoregressive integrated moving average (ARIMA) prediction model,' *Advanced Technical Analysis Concepts*, 2024. [Online]. Available: https://www.investopedia.com/terms/a/autoregressive-integrated-moving-average-arima.asp.

[22] S. Liu, J. McGree, Z. Ge and Y. Xie, 'Big data from mobile devices,' in *Computational and Statistical Methods for Analysing Big Data with Applications*, Academic Press, 2016.

[23] D. Abugaber, *A Language, not a Letter: Learning Statistics in R*. University Of Illinois Chicago, 2014, ch. 23.

[24] X. Hu and M. Yu, 'Based on ODE and ARIMA modeling for the population of china,' *Technology and Investment*, 2013.

[25] I. U. Moffat and E. A. Akpan, 'White noise analysis: A measure of time series model adequacy,' *Applied Mathematics*, 2019.

[26] H. Lütkepohl, 'Chapter 6 forecasting with VARMA models,' in *Handbook of Economic Forecasting*, G. Elliott, C. Granger and A. Timmermann, Eds., Elsevier, 2006.

[27] Penn State Eberly College of Science, '11.2 vector autoregressive models VAR(p) models,' 2024. [Online]. Available: https://online.stat.psu.edu/stat510/lesson/11/11.2.

[28] C. James and et al., 'A VARMA analysis of the causal relations among stock returns, real output, and nominal interest rates,' *The Journal of Finance*, vol. 40, 1985.

[29] S. Claris and N. Peter, 'ARIMA model in predicting of COVID-19 epidemic for the southern africa region,' *African Journal of Infectious Diseases*, 2022.

[30] S. Gharat, 'Solution for deep learning without large dataset? data augmentation!!!,' 2018. [Online]. Available: https://medium.com/@snaily16/is-deep-learning-useless-if-there-is-no-large-dataset-available-no-eac26f898968.

[31] R. Cahuantzi, X. Chen and S. Güttel, 'A comparison of LSTM and GRU networks for learning symbolic sequences,' in *Intelligent Computing*. Springer Nature, 2023.

[32] H. Wang and B. Raj, 'On the origin of deep learning,' *arXiv preprint arXiv:1702.07800*, 2017.

[33] F. A. Gers, J. Schmidhuber and F. Cummins, 'Learning to forget: Continual prediction with LSTM,' *Neural Computation*, 2000.

[34] C.-L. Hung, 'Deep learning in biomedical informatics,' in *Materials Today*, Elsevier, 2023.

[35] V. Rainardi, 'Recurrent neural network (RNN) and LSTM,' 2021. [Online]. Available: https://dwbi1.wordpress.com/2021/08/07/recurrent-neural-network-rnn-and-lstm/.

[36] MathWorks, 'Long short-term memory neural networks,' *Deep Learning Toolbox*, 2024. [Online]. Available: https://uk.mathworks.com/help/deeplearning/ug/long-short-term-memory-networks.html.

[37] S. Wang and R. M. Summers, 'Machine learning and radiology,' *Medical Image Analysis*, vol. 16, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1361841512000333.

[38] IBM, 'What is bagging?' *Think*, 2024. [Online]. Available: https://www.ibm.com/topics/bagging.

[39] M. Kumar and M. Thenmozhi, 'Forecasting stock index movement: A comparison of support vector machines and random forest,' *Indian Institute of Capital Markets 9th Capital Markets Conference Paper*, 2006.

[40] T. Hastie, R. Tibshirani and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Second. Springer, 2009.

[41] IBM, 'What is random forest?' *Think*, 2024. [Online]. Available: https://www.ibm.com/topics/random-forest.

[42] A. Saeed, 'Improving the interpretability of the random forest classifier,' *Towards Data Science*, 2020. [Online]. Available: https://towardsdatascience.com/representing-a-random-forest-with-few-decision-trees-11283e433dbb.

[43] S. Siami-Namini, N. Tavakoli and A. Siami Namin, 'A comparison of ARIMA and LSTM in forecasting time series,' in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2018.

[44] D. Kobiela, D. Krefta, W. Król and P. Weichbroth, 'ARIMA vs LSTM on NASDAQ stock exchange data,' *Procedia Computer Science*, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050922013382.

[45] M. Elsaraiti and A. Merabet, 'A comparative analysis of the ARIMA and LSTM predictive models and their effectiveness for predicting wind speed,' *Energies*, 2021.

[46] E. Cauer, W. Mathis and R. Pauli, 'Life and work of wilhelm cauer (1900 – 1945),' in *Proceedings of the Fourteenth International Symposium of Mathematical Theory of Networks and Systems (MTNS2000)*, 2000.

[47] H.-Y. Kim, 'Analysis of variance (ANOVA) comparing means of more than two groups,' *Restorative dentistry & endodontics*, vol. 39, 2014.

[48] H.-Y. Kim, 'Statistical notes for clinical researchers: Post-hoc multiple comparisons,' *Restorative dentistry & endodontics*, vol. 40, 2015.

[49] S. E. Alnaa and F. Ahiakpor, 'ARIMA (autoregressive integrated moving average) approach to predicting inflation in ghana,' *Journal of Economics and International Finance*, 2011.

[50] C. R. Ko and H. T. Chang, 'Lstm-based sentiment analysis for stock price forecast,' *PeerJ Computer Science*, 2021.

[51] C. H. Fajardo-Toro, J. Mula and R. Poler, 'Adaptive and hybrid forecasting models—a review,' in *Engineering Digital Transformation*, Springer, 2018.

[52] J. P. S. Kumar, R. Sundar and A. Ravi, 'Comparison of stock market prediction performance of ARIMA and RNN-LSTM model – a case study on indian stock exchange,' *AIP Conference Proceedings*, 2023.

[53] Y. Ensafi, S. H. Amin, G. Zhang and B. Shah, 'Time-series forecasting of seasonal items sales using machine learning–a comparative analysis,' *International Journal of Information Management Data Insights*, vol. 2, 2022.

[54] *PyTorch LSTM*. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.nn.LSTM.html.

[55] R. Aroussi, *Yfinance 0.2.38*. [Online]. Available: https://pypi.org/project/yfinance/.

[56] Y. Inc, *Yahoo! finance uk*. [Online]. Available: https://uk.finance.yahoo.com/.

[57] S. Baek, S. K. Mohanty and M. Glambosky, 'COVID-19 and stock market volatility: An industry level analysis,' *Finance Research Letters*, vol. 37, 2020.

[58] A. Hayes, 'What is closing price? definition, how it's used, and example,' 2023. [Online]. Available: https://www.investopedia.com/terms/c/closingprice.asp.

[59] *Pandas.DataFrame*. [Online]. Available: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html.

[60] *Statsmodels.tsa.stattools.adfuller*. [Online]. Available: https://www.statsmodels.org/dev/generated/statsmodels.tsa.stattools.adfuller.html.

[61] R. Harris, 'Testing for unit roots using the augmented dickey-fuller test: Some issues relating to the size, power and the lag structure of the test,' *Economics Letters*, vol. 38, 1992.

[62] *Statsmodels.tsa.arima.model.ARIMA*. [Online]. Available: https://www.statsmodels.org/stable/generated/statsmodels.tsa.arima.model.ARIMA.html.

[63] *Statsmodels.tsa.statespace.varmax.VARMAX*. [Online]. Available: https://www.statsmodels.org/stable/generated/statsmodels.tsa.statespace.varmax.VARMAX.html.

[64] *Sklearn.preprocessing.MinMaxScaler*. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html.

[65] *PyTorch dataloaders*. [Online]. Available: https://pytorch.org/tutorials/beginner/basics/data_tutorial.html#datasets-dataloaders.

[66] *Sklearn.ensemble.RandomForestRegressor*. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html.

[67] *Scipy.stats.f_oneway*. [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.f_oneway.html.

[68] *Statsmodels.stats.multicomp.pairwise*$_tukeyhsd$. [Online]. Available: https://www.statsmodels.org/dev/generated/statsmodels.stats.multicomp.pairwise_tukeyhsd.html.

[69] A. James and V. Tripathi, 'Time series data analysis and arima modeling to forecast the short-term trajectory of the acceleration of fatalities in brazil caused by the coronavirus (COVID-19),' *PeerJ*, vol. 9, 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8286711/.

[70] J. Rogers and S. Gunn, 'Identifying feature relevance using a random forest,' in *Subspace, Latent Structure and Feature Selection*, C. Saunders, M. Grobelnik, S. Gunn and J. Shawe-Taylor, Eds., vol. 3940, Springer, 2006. [Online]. Available: https://doi.org/10.1007/11752790_12.