Table of Contents

[Abstract: 2](#_gjdgxs)

[Introduction: 2](#_30j0zll)

[Background 2](#_1fob9te)

[Problem Statement 3](#_3znysh7)

[Justification of the study 3](#_2et92p0)

[Research Questions 3](#_tyjcwt)

[Aims and Objectives 3](#_3dy6vkm)

[Literature Review 4](#_1t3h5sf)

[Methodology 8](#_4d34og8)

[Overview 8](#_2s8eyo1)

[Long Short-Term Memory (LSTM): 9](#_17dp8vu)

[Architecture 9](#_3rdcrjn)

[Why use LSTM for predicting the stock market? 12](#_26in1rg)

[Autoregressive Integrated Moving Average (ARIMA) 13](#_lnxbz9)

[Architecture 13](#_35nkun2)

[Why use ARIMA for predicting the stock market? 15](#_1ksv4uv)

[Data Preparation 15](#_44sinio)

[Overview: 15](#_2jxsxqh)

[Data Collection 15](#_z337ya)

[Description of the dataset: 16](#_3j2qqm3)

[Data Preprocessing 16](#_1y810tw)

[Model Training and Analysis: 18](#_4i7ojhp)

[LSTM: 18](#_2xcytpi)

[ARIMA: 20](#_1ci93xb)

[Results And Analysis: 22](#_3whwml4)

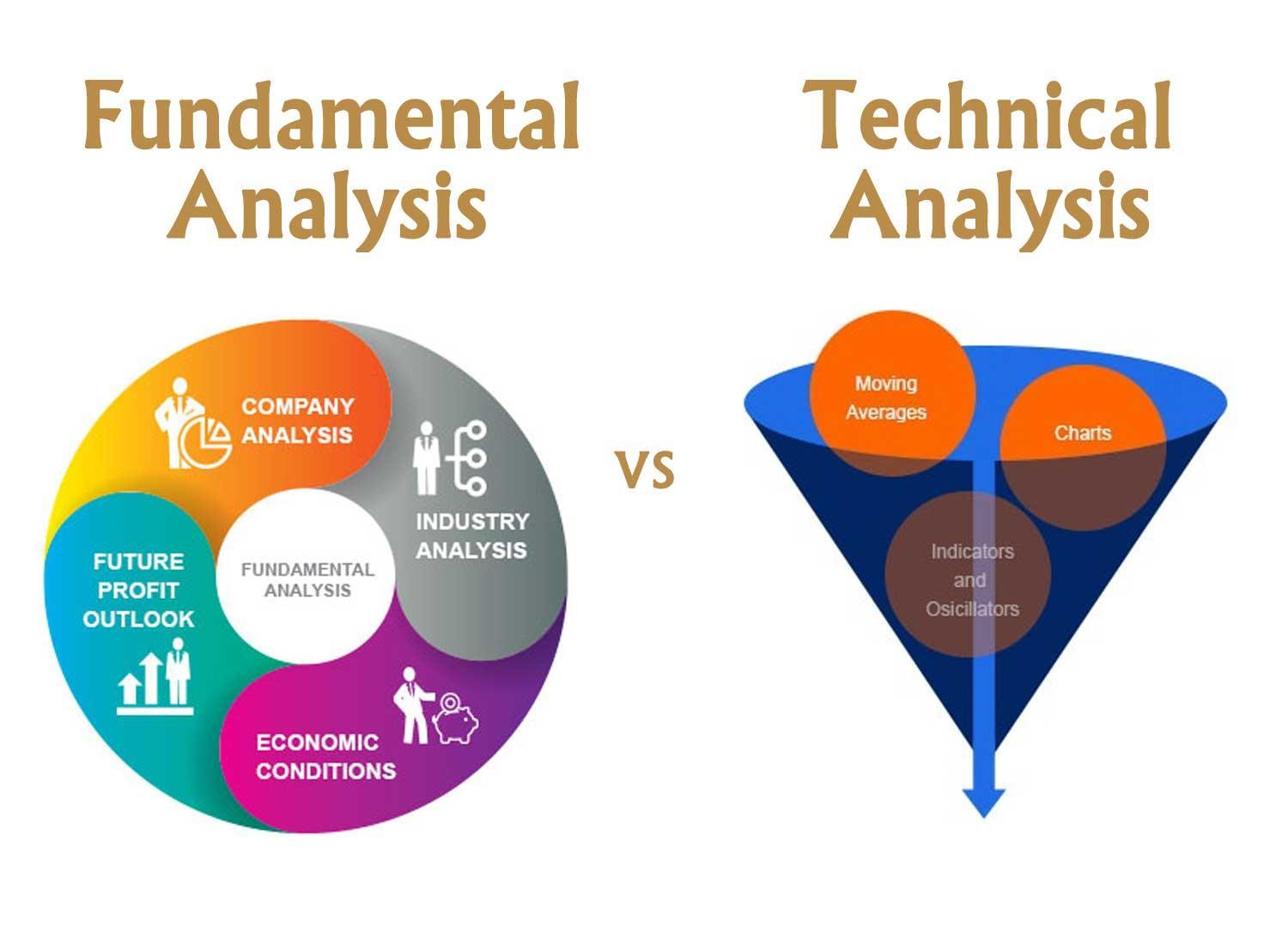
# Abstract:

Making predictions about the stock market is challenging since many factors influence how the market behaves. In this project, we will use the Istanbul Stock Exchange (ISE) using the data from the UCI machine learning repository. The goal of this project is to compare how well two models forecast the ISE index: Long Short-Term Memory (LSTM) and Autoregressive Integrated Moving Average (ARIMA). Finding the model with the highest accuracy while utilizing the least amount of data input is the project's main aim. By creating the model architecture, assessing the model performance, and figuring out the best model for short-term stock prediction, this study will greatly enhance the understanding of the stock exchange trends and the effectiveness of models in financial forecasting.

# Introduction:

## Background

Forecasting the stock market’s future using past data and other influencing factors is referred to as stock market prediction. There are two popular approaches to stock price forecasting. First and foremost is “fundamental analysis”, which can be divided into top-down and bottom-up approaches. Price/earnings (P/E) ratios and other metrics are used in bottom-up analysis to assess a company’s performance, whereas top-down analysis begins with the state of the economy and forecasts how it will impact specific stocks. The second method is “technical analysis”, which looks for patterns in charts and projects future price movement by analyzing historical price trends. A few investors utilize both approaches: technical analysis indicates the trend’s entry point, while fundamental analysis offers them multiple points of view on a stock (Figure 1).



*Figure 1: Fundamental Analysis and Technical Analysis Comparison*

The ability to make better decisions and control risk is one of the main advantages that offers investors, financial analysts, and traders to reduce risk and make smarter judgments. Because of several factors that may impact market behavior in addition to the regular fluctuations in prices, predicting stock market values is a difficult and complex undertaking.

One important financial market that represents Turkey's economic performance and investment opportunities is the Istanbul Stock Exchange (ISE). Both domestic and foreign investors can benefit from understanding and forecasting the ISE attribute. In this study, I made use of the UCI machine learning repository’s dataset “Istanbul Stock Exchange (ISE)”. With so many variables involved in predicting stock values, it is difficult to make predictions of values with great accuracy, which is where machine learning comes in. The applications of machine learning and statistical models in financial forecasting have been developed to improve the accuracy and confidence of investing predictions.

Features include the value of the Istanbul Stock Exchange (ISE) index, the Standard & Poor’s 500 index (SP), which tracks the performance of 500 large US companies, the German stock market index for 30 major companies (DAX), the London Stock Exchange’s 100 largest companies (FTSE), the Tokyo Stock Exchange’s 225 large companies (NIKKEI), and the Brazilian stock market index (BOVESPA). EU, an index that monitors stock performance among EU member nations. Emerging Markets Index (EM), which represents stocks in developing nations. Collectively, these characteristics offer a thorough understanding of the regional (ISE) and worldwide market environments, assisting in the development of a strong predictive and full understanding of the connections between the targeted variable and other financial metrics.

Machine learning research is used to establish how much historical data the model must study to anticipate the stock market. Each pricing attribute is given a weight by the machine learning model. Recurrent neural networks (RNNs) and long short-term memory (LSTMs) are the common machine learning models used to predict time series data, including stock prices, weather forecasts, and housing prices. Assessing how important each factor is of recent versus historical data helps to identify the factors that have the greatest influence on present- or next-day prices.

## Problem Statement

This study’s main goal is to use historical financial data to anticipate the Istanbul Stock Exchange (ISE) with accuracy. Choosing the right model for prediction is important because of the complexity of financial markets. Even though they are widely used, traditional time series models like ARIMA might not be able to properly represent complex dependencies on time and non-linear relationships. On the other hand, advanced deep learning models such as the LSTM network have shown progress in processing sequential data; however, they require sufficient training data and appropriate tuning. The objective of this research is to evaluate and compare the ISE indices forecasting performance of the LSTM and ARIMA models to identify which model requires less data to provide more accurate forecasts.

## Justification of the study

This study is justified by the potential benefits of improving investment forecasting accuracy. Accurate models may enhance financial decision-making, investment strategies, and risk minimization. By evaluating the performance of LSTM and ARIMA models, and identifying the benefits and drawbacks of each model, this project contributes to the ongoing research in financial projections. The findings will be useful to investors, financial analysts, and researchers who are interested in applying machine learning and statistical models in the financial market. And in cases where data is very limited, understanding which model performs best with less data might be extremely beneficial.

## Research Questions

The research aims to explain the following questions:

1. When predicting the stock market with limited amounts of data input, which model LSTM or ARIMA provides the highest accuracy?
2. By studying market movements, daily returns, moving averages, and correlations between stocks, how can we forecast future stock behavior?

## Aims and Objectives

This research aims to find out whether a model like LSTM or ARIMA predicts the Istanbul Stock Exchange (ISE) index more accurately. The objective is to:

1. Explain how well the ARIMA and LSTM model predicts the ISE index.
2. Determine which model works best with minimum data input.
3. Describes some efficient financial forecasting techniques.

# Literature Review

Qian et al. in their case study of “Stock Predicting based on LSTM and ARIMA” [1] using the LSTM and ARIMA model 95% of the total data for each stock is applied as the training set data, which is used to train the model parameters and rest is used as the test set data. The data used is Google Stock Price from April 2017 to April 2022 taken from Yahoo Finance. The model’s predicted outcomes on the test dataset validate the model’s benefits and drawbacks. To avoid overfitting, several dropout layers and six LSTM layers are incorporated into the LSTM model architecture. The output’s dimensions are 64 units, and the return sequences parameter represent stock. Using LSTM and ARIMA 487 to predict whether to return the entire series or only the last output in the output sequence, with input\_shape serving as the shape of the training set. The dense layer with a specified output of one unit will be added, and the drain layer with an output value of 0.2 will be eliminated. Lastly, the model works with a batch size of 32 and can be used for 100 iterations. However, the study doesn’t train RNN on the entire observation sequence, instead, it uses a batch of small subsequences randomly selected from the training data. According to the prediction findings, the LSTM algorithm outperforms the alternative ARIMA in terms of MSE, MAE, and RMSE. When projecting future prices, investors in the capital market may find the findings of this study useful. 97% of the data was collected, which was highly promising for this research project.

Ma et al. examined the three models in their research [2]. Through an analysis of the three model’s underlying assumptions and forecast outcomes, this paper particularly contrasts the three models. Although the LSTM model is heavily influenced by the data processing, it is considered to have the best prediction performance in the end. In terms of performance, the ANN model outperforms the ARIMA model. Also, Ma, Q. mentioned the ANN may be more responsible for the LSTM model’s performance. Additionally, by strengthening the white noise sequence, ARIMA-GARCH can raise the ARIMA model’s accuracy even further. In comparison with the other two models, the LSTM model adds additional variables to differentiate between abrupt changes and sudden fluctuations in markets. The dataset is “DELL's stock price” from 2010 used to analyze the models in this study.

Sunki et al. concluded that time series forecasting in the stock market is a challenging task requiring technical methods and careful research [3]. Though no forecasting technique can accurately predict stock prices, time series forecasting offers helpful insights and helps investors make viable choices. These models incorporate a variety of factors, including trend, seasonality, and autocorrelation, to generate projections. As a result, it can be concluded that the ARIMA model provides a better match to the data than both LSTM and FBProphet models.

Based on these three models, ARIMA has the lowest RMSE (root mean square error) value 7.8919253, suggesting that it has the highest predicted accuracy for this dataset. Followed by LSTM model has a 10.33765 RMSE value and FBProphet 9.11863 also had a lower RMSE value which appears to be the least accurate in predicting the target variable. The dataset used in this paper is “Netflix”

Kobiela et al. in their case study of using ARIMA and LSTM[4], which model works better in terms of the selected input data, parameters, and the feature count has been shown in this research paper. Mean Square Error (MSE) and mean absolute percentage error (MAPE) were the relative metrics used to compare the selected models. In regression problems usually use selected measures. By comparing the selected metrics in various models, the study reveals which model performs better.

The prediction of the stock prices performed over a range of periods, from one day to nine months. Overall, ARIMA is better at processing single-feature (price) data, and it generally performs better than LSTM, especially over a long period. Due to the absence of more features, LSTM struggled with learning and only performed better for one-day forecasts. Future study involves adding more features and exploring hybrid models which potentially increase LSTM performance. The study concluded that for single-feature stock price prediction on “NASDAQ data”, ARIMA is now more successful.

Varshney et al. this study compares and calculates the stock price forecasting using ARIMA and ANN [5]. The outcomes showed that the ANN approach to stock price forecasting is more accurate than ARIMA modeling. The NIFTY 50 index closing price from the NSE India served as the source of data for this research. The “Levenberg–Marquardt” algorithm was used to simulate and train the network. Given the forecast error arising from these two models is rather small, it is evident that they reached nearly forecast performance. These results are comparable to those of Adebiyi et al. (2014) ANN results are predicted more accurately than ARIMA. The ANN model’s predicted line and the NSE index actual values nearly overlap. On the other hand, the ARIMA model’s prediction appears to diverge rather than exactly overlap. Towards the end of the study period, the ARIMA results show rising deviation as it shows a linear pattern that is directed based on the ARIMA model results.

Talati et al. suggest that LSTM can manage the sequential and nonlinear structure of financial data and the model is good at predicting stock prices [6]. The author believes that feature selection and data preparation can improve the model’s performance. The author has trained the model on three different datasets such as “Infosys (1996 to 2022), Microsoft (1986 to 2022), and TCS (2002 to 2022)”.

Huang et al. concludes that the LSTM model [7] can learn from the sequential patterns of stock data and include long-term dependencies, as they are quite predictive at predicting the prices of the stock. The study shows that the proposed model performs much better than the earlier model by Xu, and Cohen & Ullah, and Qasim, with improvements of 35.18% and 5.86% respectively. The model achieves a high accuracy of 86.77% and shows a significant reduction in errors, with a mean squared error (MSE) and root mean squared error (RMSE) much lower than the original models. The model consists of four layers (100, 50, 100, and 30 units) with dropout used to prevent overfitting. These findings demonstrate the LSTM model’s efficiency and significance of social media variables for accurate stock movement forecasting. The author uses the historical stock price data of “Google stocks “from 2014 to 2019.

Adebiyi et al. In this study the understanding of ARIMA and ANN [8] models for predicting “Dell stock prices” were examined. Based on the minimal forecast errors and a close match to actual prices, ARIMA (1,0,0) was found to be the optimal configuration. Additionally, the ANN model with a 10-17-1 architecture demonstrated great accuracy and minimal mistakes. Although both models worked well, test results showed that the ANN model typically produces forecasts that are more accurate than those made by the ARIMA model. Though the ANN model performed somewhat better overall, statistical tests revealed little variance between the actual and predicted values for either model.

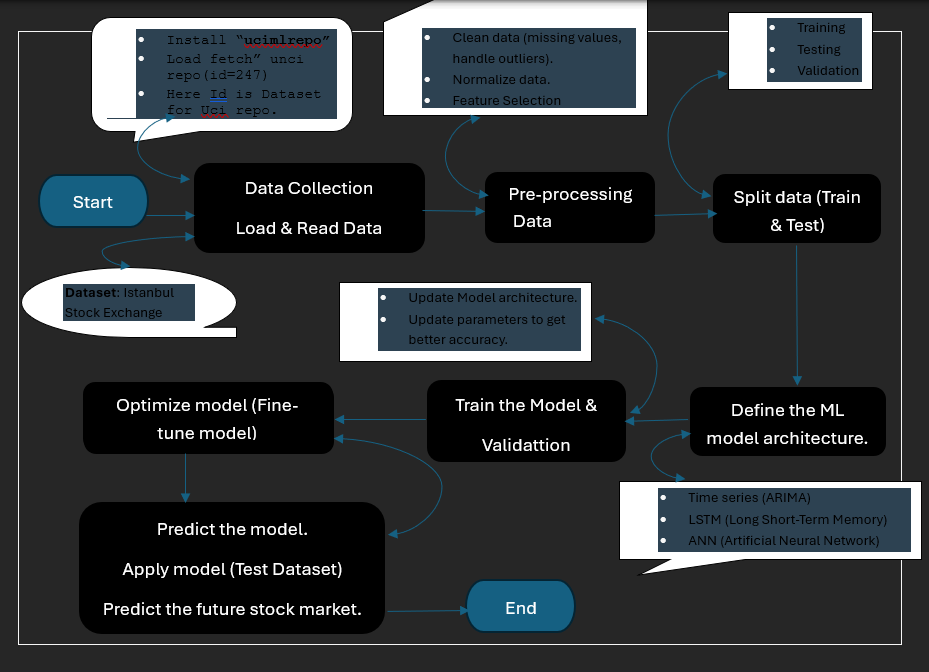
Whereas the forecasts of the ANN model closely matched the actual values, the ARIMA model had a directional pattern. To increase forecast accuracy even further using new market indexes and recent stock data, future research may investigate hybrid models.

|  |  |  |  |
| --- | --- | --- | --- |
| **Paper** | **Machine-Learning models** | **Datasets** | **Results** |
| Stock Predicting based on LSTM and ARIMA | LSTM, ARIMA | Google Stock Price |  |
| Comparison of ARIMA, ANN, and LSTM for Stock Price Prediction | LSTM, ARIMA, ANN | DELL's stock price |  |
| Time series forecasting of the stock market using ARIMA, LSTM, and FB prophet | LSTM, ARIMA, FB prophet | Netflix data |  |
| ARIMA vs LSTM on NASDAQ stock exchange data | LSTM, ARIMA | NASDAQ data |  |
| A Comparative Study of Future Stock Price Prediction Through Artificial Neural Network and ARIMA Modelling | ANN, ARIMA | NSE India data |  |
| Stock Market Prediction Using LSTM Technique | LSTM | Infosys Microsoft & TCS |  |
| Enhancing Stock Market Prediction Through LSTM Modeling and Analysis | LSTM | Google stocks |  |
| Comparison of ARIMA and Artificial Neural Networks Models for Stock Price Prediction | ARIMA, ANN | Dell stock prices |  |
|  |  |  |  |
|  |  |  |  |

# Methodology

## Overview

The goal of this project is to use machine learning techniques to predict the Istanbul Stock Exchange. This will be executed in a few steps (Figure 2). Collecting and reading the data is the first step, and then comes data preprocessing, to handle the missing values and identify the features from the dataset. Then data is split into training and testing and trained using LSTM and ARIMA models. Subsequently, we will test the models, and in the end, we’ll assess model performance by tuning the parameters to see which model forecasts the future stocks more accurately.



**Figure 2: Data management plan flow chart**

## Long Short-Term Memory (LSTM):

Long Short-Term Memory Networks are a type of Recurrent Neural Network (RNN) that process sequential data and collect long-term dependencies. “Hochreiter” and “Schmidhuber” created LSTM to address the issues identified by conventional RNN and machine learning methods. When handling time-series data, like stock prices, which are generally non-stationary and show trends and changing circumstances, LSTM proves to be particularly useful.

### Architecture

By understanding the LSTM’s architecture, we shall see in the following section how it solves this issue. LSTM functions quite similarly to an RNN cell at a high level, and this is how the LSTM network operates internally. As seen in the diagram below, the LSTM network design is divided into three components, each of which carries out a distinct task.

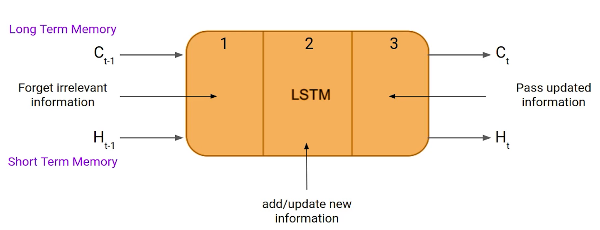


Figure 3: LSTM Architecture

From the above figure [Figure 3] as shown, an LSTM unit consists of three components referred to as gates. They regulate the information that enters and exits the memory cell, also known as the LSTM cell. The first gate is known as the “Forget gate”, where data from the preceding timestamp is either significant and should be remembered or can be ignored. Further, the cell attempts to learn new information from the input to this cell at the second gate, which is referred to as the “input gate”. The cell transfers the modified data from the current timestamp to the subsequent timestamp in the final gate, which is called the “output gate”. This LSTM cycle is considered a single-time step.

An LSTM has a hidden state, just like a basic RNN, where “H(t-1)” is the hidden state of the timestamp that was previously recorded, and “Ht” is the hidden state of the timestamp that is currently recorded. Furthermore, the cell state of an LSTM is denoted by “C(t-1)” for the past timestamp and “C(t)” for the present timestamp, respectively. In this case, long-term memory refers to the cell state, and short-term memory to the hidden state.

Let’s use the equation to better understand how these gates function in the LSTM architecture [Figure 4].

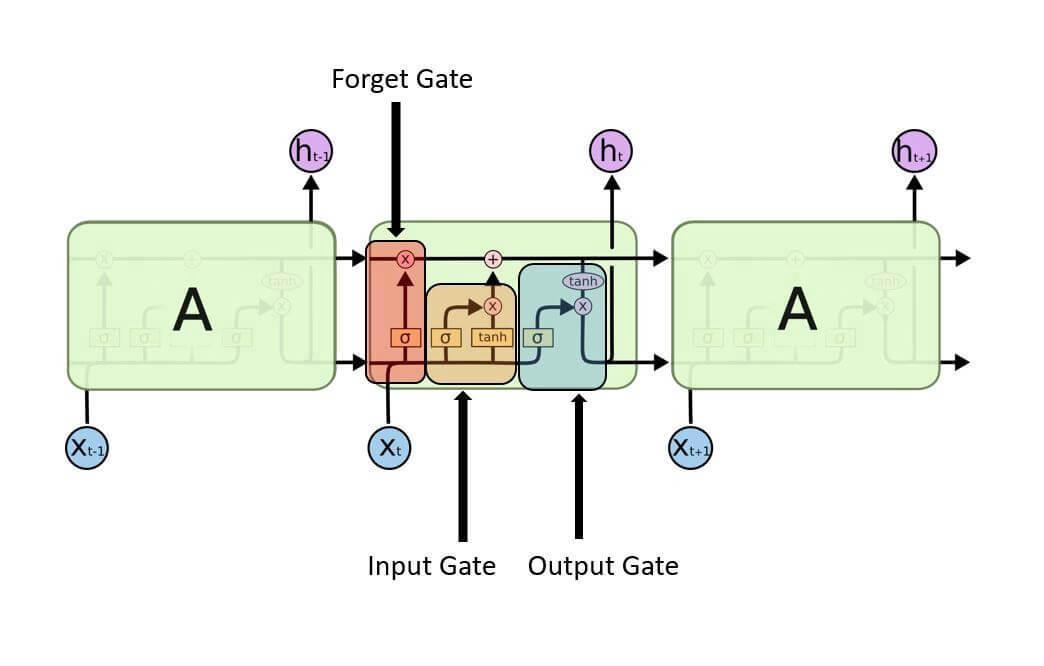


Figure 4: Long Short-Term Memory Network

Forget Gate:

In this gate, we must figure out whether to keep or remove the data from the previous time step. Below is the forget gate equation.

)

Here, Let’s understand the working of the equation,

Xt: It’s the timestamp’s current input.

Uf: The input’s associated weight.

Ht-1: The previous timestamp’s hidden state

Wf: The weight matrix related to the hidden state.

It then undergoes the use of a sigmoid function. As a result, f and t will become a numbers 0 and 1. And then the cell state of the earlier timestamp is then multiplied by this f and t.

The input and output gate equation is essentially the same as that of the forget gate. Likewise, because of the sigmoid function, its value will also range between 0 and 1. The equations below can be used to feed new data into the input gate and extract the output from the output gate.

Input gate new information:

To handle new information, a sigmoid function is used to modify the existing data, and that’s how the neural networks operate rather than classifying information to be important or not. As a result, the data has been updated completely. LSTM, on the other hand, uses a mechanism to transmit the data known as cell states while executing minimal addition and multiplication changes to the data.

Here is the equation to calculate the new information.

The function of a hidden state at timestamp t-1 and input x at timestamp t now identifies the new information that has to be provided to the cell state. And “tanh” is the activation function in this case. The value of new information will range from -1 to 1 as a result of the “tanh” function. Information is added to the cell state at the current timestamp if the value of Nt is positive, and subtracted from the cell state if it is negative. The cell state at the current timestamp is represented by Ct-1 in this case, and the other values are those we previously calculated.

Output gate:

In this gate we will use Ot and tanh of the updated cell information to compute the current hidden state as shown below.

It turns out that the present output and long-term memory (Ct) determine the hidden state. Simply use the “SoftMax” activation on the hidden state Ht if you need to extract the current timestamp output. Finally, the token with the highest score in the output is used to make the prediction.

**Essential elements of LSTM:**

* **Memory Capability**: Because LSTMs can remember information over time, they are perfect for capturing long-term dependencies in sequential data.
* **Gated mechanism**: LSTMs use gates to manage information flow, enabling them to store important data and eliminate unnecessary information.

### Why use LSTM for predicting the stock market?

With the recent breakthroughs that have been happening in data science, it is found that for almost all these sequence prediction problems, long short-term Memory networks have been observed as the most effective solution (Talati et al.). The basic idea of traditional statistical models such as linear regression and Autoregressive Integrated Moving Average (ARIMA) is that the data is stationary and that its statistical features such as variance and mean don’t change over time.

Because LSTM has feedback connections, as compared to standard neural networks, it can handle complete data sequences as an alternative to only handling single data points. In addition to this, it is very good at identifying and forecasting patterns in sequential data, such as speech, text, and time series. By extracting important insight from sequential data, LSTM has developed into an effective tool in deep learning and artificial intelligence that has assisted advances across a range of industries.

Stock prices, on the other hand, show seasonality, patterns, and trends despite not being stationary. LSTMs are ideally suited for stock market prediction because they can manage this non-linear correlation within the data.

## Autoregressive Integrated Moving Average (ARIMA)

A statistical analysis model known as an autoregressive integrated moving average (ARIMA) makes use of time series data to forecast future trends by using historical data or to get a deeper understanding of the data set. One type of regression analysis that evaluates the strength of a single dependent variable to other changing variables is the ARIMA model. Rather than using actual values, the model looks at differences among values in the data set to forecast future movements in securities or the financial markets.

### Architecture

An understanding of an ARIMA model can be gained by describing each of its components as follows:

1. Autoregressive (AR): A model that displays a variable that is evolving and evaluates on its own lagged, or prior, results is known as an autoregression model.
2. Integrated (I): To enable the time series to become stationary, the difference between the current and previous values is substituted for the data values, which are represented by integrated.
3. Moving average (MA): Considers a data point’s connection with a residual error by applying a moving average model to observations that are lagged.

With a standard notation, every component in ARIMA operates as a parameter. Standard notations for ARIMA models are with p, d, and q, where the parameters are replaced by integer values to denote the type of ARIMA model that is being utilized. One way to define the parameters is as follows:

* AR (P): the model’s autoregressive terms or the number of lag observations.
* I (d): the variation in the observation that is not seasonal, or the number of differencing cycles applied to the raw observations.
* MA (q): the moving average window size, sometimes referred to as the moving average order.

Computer methods and machine learning approaches are used to compute the ARIMA model, which is complex and performs best on very large data sets. To make the data stationary in an ARIMA model, they are different. A model that demonstrates consistency proves the data stability across time. Since the majority of market and economic data exhibits trends, the goal of differentiation is to eliminate any seasonal patterns or trends.

Regression models may be adversely affected by seasonality, which is the occurrence of regular and predictable patterns in data that reoccur over a year. Many calculations made during the process cannot be completed and the desired outcomes can’t be obtained if a trend develops, and stability is not obvious.

To start creating an ARIMA model for a stock, we first download the maximum amount of price data. After determining the data trends, use the autocorrelation to determine the lowest level of differencing (d). A series is considered to be different if the “lag-1” correlation is either zero or negative. If the “lag-1” is greater than zero, you might need to vary the series more. Next, compare the self-correlation and partial autocorrelation to figure out the order of regression (p) and the order of moving average (q).

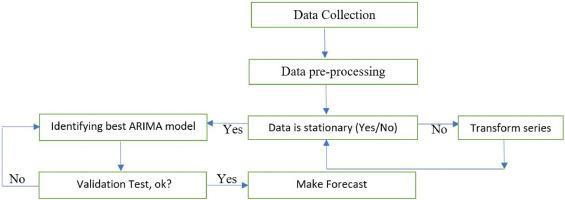


Figure 5: Methodology to apply the ARIMA model for forecasting

To understand the working of the ARIMA model in basic terms [Figure 5]:

* Collect data: Gather the data that will be studied or used to forecast trends.
* Review data: Ensure that there are no major long-term changes to the data. If so, you may need to make sure that data is stationary through small adjustments.
* Identify model: Examine the data to determine how much, if any, adjustment is necessary and how historical data influences current data. Python libraries can be used for the time series data.
* Check model accuracy: Compare the actual data with the predictions generated by your Python ARIMA model to see if the model accurately describes the data.
* Predict future: After the model is well developed, utilize it to project future events based on the prediction of your model.
* Improve model: If the predicted outcomes don’t look great, tune the parameters of the model until the projections look great.
* And finally, run the model on the testing dataset, verify the predictions, and compare the predicted and actual values.

### Why use ARIMA for predicting the stock market?

Since ARIMA captures patterns, seasonality, and correlation between past and future values, it is a useful time series data model for stock market forecasting. The model’s adaptability to different market conditions can be achieved by fine-tuning it to certain data sets. ARIMA stabilizes non-stationary data by differencing the series, an important characteristic of stock prices. ARIMA’s historical accuracy and statistical basis make it a useful tool in financial forecasting, however, it is best suited for short-term forecasts.

## Data Preparation

### Overview:

Using advanced machine learning techniques, namely “Long Short-Term Memory (LSTM) networks” and “ARIMA” model, the main goal of this research is to accurately forecast stock values for the “Istanbul Stock Exchange (ISE)”. To create and assess prediction models, the approach consists of multiple phases, such as feature selection, data preprocessing, data-collecting, and detailed data analysis. This section describes how the dataset was prepared and analyzed for accurate stock market forecasting, along with the considerations and actions that were involved.

### Data Collection

The UCI machine learning repository, which offers a large dataset for predicting stock market research, is the source of the data used in this work. Daily data from several global market indexes, including the ISE index, FTSE, DAX, NIKKEI, BOVESPA, and Standard & Poor (SP), EU, EM, is included in the dataset. Together, these indexes reflect a range of economic situations and offer a solid basis for simulating the movements of the Istanbul Stock Exchange.

A screenshot of a computer

Description automatically generated

Figure 6: Dataset

### Description of the dataset:

The primary target variable is the Istanbul Stock Exchange (ISE) index, which displays the value of an index of stocks and bonds traded there. Furthermore, the dataset contains other attributes that map to other important global indices, including:

* Standard & Poor’s 500 index (SP): It is a key metric for both the US stock market and the world economy since it represents the economic growth of 500 large-cap US corporations.
* DAX: over 30 significant German firms that trade on the “Frankfurt Stock Exchange” are tracked by DAX, which offers market insights throughout Europe.
* FTSE: The FTSE, which tracks the top 100 firms listed on the “London Stock Exchange”, is an essential indicator of the state of the UK economy.
* NIKKEI: Considering the state of the Japanese economy, it consists of 225 business that are listed on the “Tokyo Stock Exchange”.
* BOVESPA: Brazil’s main stock market index, which shows how the country’s economy is doing.
* EU: A composite indicator that reflects the European Union’s economic activity.
* EM: The Emerging market index provides a perspective on the state of the developing countries’ economies by representing stocks from these markets.

### Data Preprocessing

To ensure that the data is clear, consistent, and appropriate for analysis, preprocessing the data is an essential step in getting the dataset ready for modeling. The steps needed to preprocess the data are described in this section.

Handling missing values: Machine learning model performance can be greatly impacted by missing data. Therefore, handling null values, and missing data is the first step in preprocessing.

#### Scaling and Normalization:

All numerical features were and scaled to fall into a comparable range since any machine learning model, particularly LSTM, ARIMA, works better with normalized data. This enables greater convergence of the model and boosts the learning process.

MinMax Scaling: Features were scaled using the “MinMaxScaler” method. By modifying the values according to the lowest and highest values of each feature, this scaler converts the attributes to a specified range, usually [0,1]. To maintain the strength of each feature’s relationship, scaling was done to each feature separately.

#### Data Feature Selection

Feature engineering is the process of adding new features or changing pre-existing ones to improve the model’s ability to predict the future.

Feature selection: In addition to the ISE index, the main features used for the analysis are the DAX, SP, FTSE, NIKKEI, BOVESPA, EU, and EM. These characteristics were selected considering their potential impact on the target variable and its importance.

Below, find the correlation matrix (heat map) for the data used for the model performance.

A screenshot of a color chart

Description automatically generated A screenshot of a color chart

Description automatically generated

Fig 7 [a]: Matrix before feature extraction Fig 7 [b]: Matrix after feature extraction

The first heatmap shows the ISE, SP, DAX, FTSE, NIKKEI, BOVESPA, EU, and EM among its seven indices. With values like 0.87 between DAX and FTSE and 0.95 between EU and FTSE, the heatmap demonstrates the strong correlation between European indices like the FTSE, Eu, and DAX. They have a great deal in common, indicating a strong regional connection. The EU (0.72) and BOVESPA (0.69) have moderate correlations with the Emerging Markets (EM) index, suggesting that there is some global trend alignment. Conversely, the NIKKEI exhibits poor correlations with the majority of indices, particularly with the SP (0.13) and DAX (0.26), indicating that it may be useful for feature extraction diversification since it gathers distinct market data.

ISE, SP, DAX, NIKKEI, and BOVESPA are among the five indices in the second heatmap, which focuses on a smaller number of more specific indexes. Certain correlations hold in this instance, such as the robust correlation of 0.72 between SP and BOVESPA and the high correlation of 0.69 between DAX and SP. There are fewer market perspectives available without FTSE, EU, and EM.

#### Transforming data:

The data must be transformed into a 3D array format, including the features, time steps, and samples, to be ready for the LSTM model. Whereas, for ARIMA models data must be stationary, which means that statistical parameters like mean, and variance shouldn’t vary with time. Trends and seasonality are eliminated using transformations like differencing, which subtracts old data points from current ones, to achieve consistency.

#### Splitting the dataset:

The dataset was divided into training and testing data sets to assess how well the predictive models performed. Generally, a higher percentage of the data about 70-80% is used for training to feed the model with sufficient knowledge to learn, and the remaining data is used for testing.

## Model Training and Analysis:

### LSTM :

A screenshot of a computer

Description automatically generatedFig8[a]: Model with all features

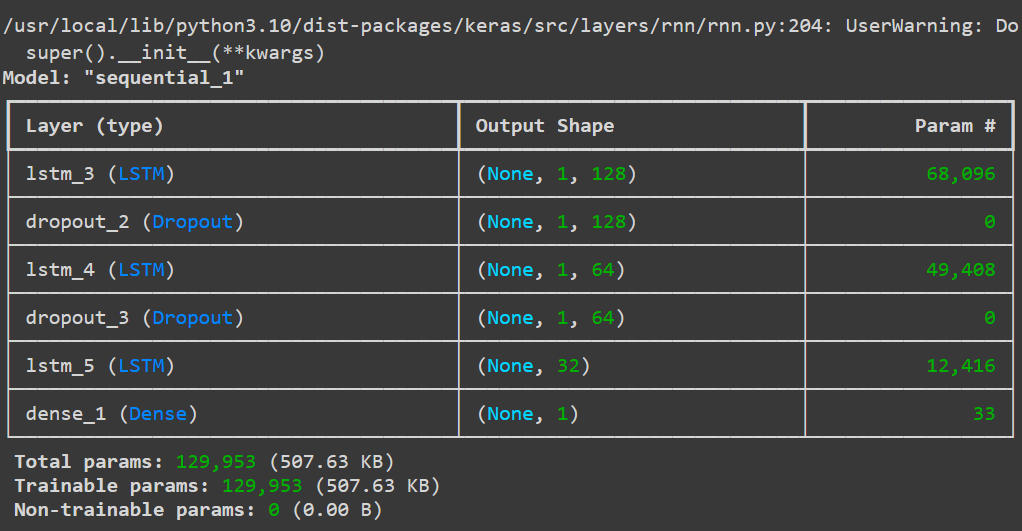


Fig8[b]: Model after feature extraction

The figures above describe the creation of the sequential model using keras for sequence-based applications such as time series prediction. The first layer is an LSTM layer with 128 units that processes the input sequence and outputs the entire sequence. After that, a dropout layer randomly removes 30% of the units during training to prevent overfitting. A second LSTM layer with 64 units returns the entire sequence, followed by another dropout layer. When the output of the model is a single value, the third LSTM layer, which has 32 units is different such that it only returns the last output of the sequence. Since the last layer is dense and has only one unit, the model will only produce one continuous value. The model is compiled using the “Adam optimizer” with a learning rate of 0.0003, and the “Mean Squared Error (MSE) as the loss function, followed by “Mean Absolute Error (MAE) as the performance metric. Regression activities involving sequential data, which means time series forecasting, frequently use this structure.

Training curve:

A graph of training and validation

Description automatically generated A graph of training and validation

Description automatically generated

Fig: Training curve with all features Fig: Training curve after feature extraction

A graph showing a blue and red line

Description automatically generated

Figure 9

The model is trained on all features that are available in the above image. The blue line, which represents the training data, has a higher degree of volatility, suggesting that the model fits the data including the noise in the dataset quite closely. The model obtains a lower training loss of 0.100 despite this overfitting, indicating that it is well-tuned to the training set, maybe at the cost of its capacity to generalize to new data. While the black line shows the model’s prediction, closely matching the training data, the red line reflects the original target data. This could result in a minimal training loss but a worse performance on unknown data.

A graph showing a graph of a training

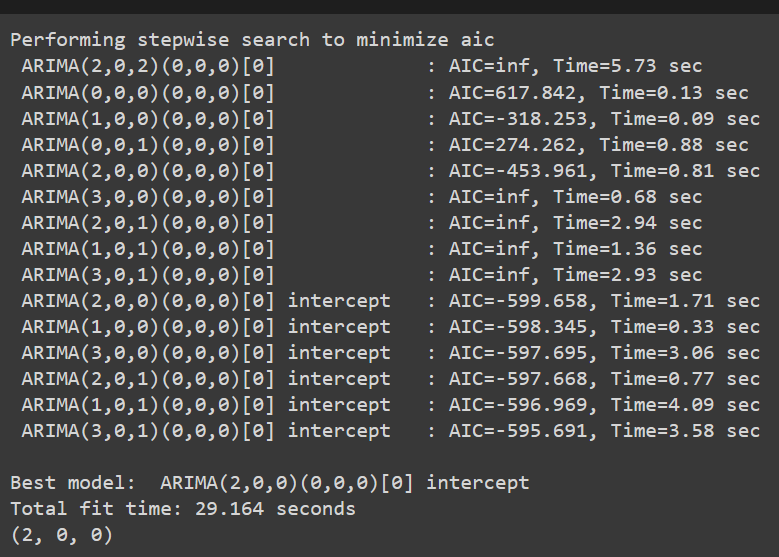
Description automatically generated with medium confidence

Fewer features were used to train the model in this image. The training data after feature extraction is represented by the brown line, which exhibits less fluctuation than the first image and may indicate that noise has less of an impact on the model. The model’s predictions, which closely match the actual data, are displayed on the yellow line, while the blue line depicts the original target data. Because it is not overfitting to noise, the model with fewer features is probably better at generalizing to new data, even though its training loss is 0.0127 larger than in the image [FIGURE].

### 

### ARIMA:

A code is designed to develop a better time series forecasting model by adding exogenous (independent) variables to the ARIMA technique. A function included in the “pmdarima” package, “auto\_arima” iterates over various combinations of the ARIMA components (p, q, d) to automatically determine the optimal ARIMA model parameters (FIGURE). Then, a model is fitted using the “SARIMAX” function from the statsmodels.

 A screenshot of a computer

Description automatically generated

Fig 11: Auto-arima order Fig 12: Sarimax model summary

Model summary description:

The auto\_arima function defines SARIMAX as the model with an order of (2,0,0) that is utilized to forecast the target variable ‘y’. There were 428 data points used in the construction of this model. The model’s log-likelihood is 381.715, which is a good fit because higher values suggest a better match. Several criteria are used to compare this model’s quality to others: the Bayesian Information Criterion (BIC), which penalizes more complex models more heavily than the Akaike Information Criterion (AIC), which is -749.429; the Hannan-Quinn Information Criterion (HQIC), which balances model complexity and fit, is -738.207; and finally, AIC has the lower value indicates a better model.

A well-fitted and statistically reliable model is produced by the SARIMAX (2, 0, 0) model with four exogenous variables. While the AR component, particularly the second lag, reflects some of the auto-regressive behavior in the time series, the significant exogenous variable implies that external factors are essential in forecasting the target variable. The model assessment is typically good despite some variance from normality, which makes it a trustworthy forecasting tool for the provided data.

A graph of a graph

Description automatically generated with medium confidence

Figure 13:

The predicted data in the above graph [], which displays the ARIMA model trained with all features, closely resembles the training data, suggesting a possibility of overfitting. As seen by the variable projected data, the model performs well on training data but produces less trustworthy adaptation for future predictions because it captures even changes in the data.

A graph of a graph

Description automatically generated with medium confidence

Figure 14:

Following feature extraction, the model in this graph becomes less sensitive to noise and wider based on key patterns. The model provides more stable, consistent, and better-generalized forecasts with a better balance between accuracy and robustness, even though the prediction lines are not as closely matched with the training data.

# Results And Analysis:

Several metrics, including (MAE), (MAPE), (MSE), (RMSE), are frequently used to assess the performance of statistical or machine learning models. The model’s prediction reliability and accuracy are revealed by these indicators. The LSTM model exhibits comparatively low error rates before feature extraction. The model’s average variation from the actual values is 0.0785 units, as indicated by the MAE of 0.0785. The average percentage variation from the actual data is estimated to be approximately 18% based on the MAPE of 17.97%. The model is producing predictions that are reasonably near to the actual values, with fewer squared errors, according to the MSE of 0.01078 and RMSE of 0.1038. It makes sense that RMSE would be somewhat higher than MAE because it assigns greater weight to larger errors.

The performance of the LSTM model marginally declined after feature extraction. The mean absolute error (MAE) went up to 0.0838, suggesting a modest increase. Additionally, the MAPE rose to 19.51%, indicating that the average percentage error is now greater than it was before feature extraction at roughly 19.51%. But both the MSE and the RMSE increased to 0.01231 and 0.1109, accordingly, suggesting that the model’s predictions are now less accurate when the extracted features are used. The LSTM model’s performance did not increase with feature extraction. Rather, all error metrics experienced a minor rise, suggesting that the feature extraction process may not have been successful or may have added noise or redundant features that negatively impacted model performance.

Lower error rates were obtained when the ARIMA model was first trained will all the features, resulting in a mean Absolute Error (MAE) of 0.080 and RMSE of 0.106. These reduced error values show how well the model matches the training set, picking up on even the smallest differences and trends. This, however, may indicate overfitting, a condition in which the model gets overly adapted to the training set, hence impairing its capacity to generalize effectively to fresh, untested data. Next, to feature extraction, which involved removing unnecessary or less significant features, the model’s error rates marginally jumped, with an RMSE of 0.114 and an MAE of 0.085. These larger error values point to a more generalized model that is less noise-sensitive and more suited for real-world forecasting, even though they also reflect a less accurate fit on the training set. By striking a balance between model complexity and robustness, the feature-extracted model trades off some accuracy for better generalization and stability.

Numerous tests were carried out by adjusting the hyperparameters to determine the ideal model structure after an LSTM model was defined and developed to estimate a single constant value. The number of timesteps, the number of units in LSTM layers, the dropout rate, and the learning rate were the main hyperparameters that were adjusted during these studies. The model was trained for 50 epochs in the first experiment using a single timestep, 100 units, a 0.2 dropout rate, and a learning rate of 0.001. The validation MAE of 0.0998 showed a decent result for short-term prediction. Later tests found that while the training length got higher, performance was somewhat improved, with a validation MAE of 0.1039, when the timestep and units were increased to 8 and 120, respectively, while the learning rate was lowered to 0.0001. On the other hand, trails with a timestep of 10 and 20 demonstrated that more errors appeared by a combination of a reduced learning rate and additional increases in timestep and units, indicating problems with overfitting and challenging convergence.

The model performed poorly in subsequent trials with smaller unit counts and extremely low learning rates, as demonstrated by larger MAE values and slower training, suggesting that the model had difficulty learning under these circumstances. When the learning rate was slightly increased to 0.00001 and 0.0001, the model performed better, especially when a timestep of 5 and 100 units was used. However, the model did not stabilize and produce consistent results until a learning rate of 0.0003. The model performed exceptionally well, which involved configuring the model with a timestep of 1, a batch size of 16, 128 LSTM units, a dropout rate of 0.3, and a learning rate of 0.0003. The model was trained for 100 epochs, and the results showed that the training process produced a low loss of 0.0100 and an MAE of 0.0784 on the training data. The validation results, on the other hand, were extremely promising, showing a validation loss of 0.0081 and a validation MAE of 0.0726. These figures suggest that the models were well-tuned and capable of producing precise predictions, effectively striking a balance between underfitting and overfitting. Out of all the configurations examined, this one proved to be the most efficient, offering the best performance.

# Conclusion

Reference:

1. Qian, H., 2022. [Stock predicting based on LSTM and ARIMA.](https://www.atlantis-press.com/proceedings/icedbc-22/125983558) In: *Proceedings of the International Conference on Economic Development and Business Culture (ICEDBC 2022).* Atlantis Press.
2. Ma, Q., 2020. [Comparison of ARIMA, ANN, and LSTM for stock price prediction.](https://www.researchgate.net/publication/347611169_Comparison_of_ARIMA_ANN_and_LSTM_for_Stock_Price_Prediction)
3. Sunki, A., Satyakumar, C., Surya Narayana, G., Koppera, V. and Hakeem, M., 2024. [Time series forecasting of the stock market using ARIMA, LSTM, and FB Prophet.](https://www.matec-conferences.org/articles/matecconf/pdf/2024/04/matecconf_icmed2024_01163.pdf) *MATEC Web of Conferences*.
4. Kobiela, D., Krefta, D., Król, W. and Weichbroth, P., 2022. [ARIMA vs LSTM on NASDAQ stock exchange data.](https://www.sciencedirect.com/science/article/pii/S1877050922013382) *Procedia Computer Science.*
5. Varshney, S. and Srivastava, P., 2024[. A comparative study of future stock price prediction through artificial neural network and ARIMA modelling.](https://journals.sagepub.com/doi/10.1177/09711023241230367?icid=int.sj-full-text.similar-articles.1) *Journal of the Institution of Engineers (India): Series C.*
6. Talati, D., Patel, M. and Patel, B., 2024. [Stock market prediction using LSTM technique.](https://www.ijraset.com/research-paper/stock-market-prediction-using-lstm-technique) International Journal for Research in Applied Science and Engineering Technology (IJRASET).
7. Huang, W., 2023. [Enhancing stock market prediction through LSTM modeling and analysis.](https://eudl.eu/pdf/10.4108/eai.2-6-2023.2334692) School of Software, South China Normal University, Guangzhou, China.
8. Adebiyi, A., Aderemi, A. and Ayo, C.K., 2014. [Comparison of ARIMA and artificial neural networks models for stock price prediction.](https://onlinelibrary.wiley.com/doi/full/10.1155/2014/614342) *Journal of Data Science*.
9. Ouattara, A. (2023). [Understanding the difference between technical and fundamental analysis in investment.](https://www.linkedin.com/pulse/understand-difference-between-technical-fundamental-trading-ouattara) LinkedIn.
10. Analytics Vidhya. (2021). [Introduction to Long Short-Term Memory (LSTM).](:https:/www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/)
11. Analytics Vidhya. (2022). [An overview on Long Short-Term Memory (LSTM).](https://www.analyticsvidhya.com/blog/2022/03/an-overview-on-long-short-term-memory-lstm/)
12. Khan, F.M. and Gupta, R., 2020. [ARIMA and NAR based prediction model for time series analysis of COVID-19 cases in India.](https://www.sciencedirect.com/science/article/pii/S2666449620300074) Journal of Safety Science and Resilience.

Prophet model

Facebook created prophet, an open-source forecasting tool, to handle timeseries data with significant seasonal trends and the possibility of missing data or variations. It is easy to use, requires little adjustments, and works especially well for tracking seasonality on a daily, weekly, and annual basis. Prophet automatically manages holidays, specially used to forecast activities like traffic, trading, and sales. With only few manual modifications, the model is reliable and applicable to both basic and complicated datasets.

A green and blue graph

Description automatically generated

The graph shows a comparison of the prophet model predicted values (in blue) and the ISE index’s actual historical values (in green). When compared to the projected data, the actual data exhibits substantial volatility, with frequent variations and a missing trend. The gap implies that the prophet model is having trouble explaining the ISE index’s underlying volatility and complexity. As time passes, there is an increase amount of uncertainty in the model’s predictions, as indicated by the confidence interval (shown by the gray-shaded area) surrounding the forecast. But even within this range, the forecast is unable to match the significant fluctuations observed in the real data. The result suggests that prophet may not be the best choice for high-risk financial data, such as the ISE index, even though it is an effective method for time series with distinct trends and seasonality.

Prophet Error metrics

The error measurements of the prophet model show that the RMSE is roughly 0.013 and the MAE is about 0.01. The RMSE is marginally higher than the MAE, indicating that although the model performs rather well on average, its capacity to capture significant deviations is limited. This suggests that forecasting the more erratic components of the data may be difficult.