STOCK PRICE PREDICTION

Session Monsoon 2020

Submitted By

ADITYA PURANIK (1710110025)

RISHIKA CHAUDHARY (1710110276)

VAIBHAV SACHDEVA (1710110369)

Under Supervision

of

DR. S.K. NEOGY

(Professor, INDIAN STATISTICAL INSTITUTE)

ABSTRACT

The prediction of a stock market direction may serve as an early recommendation system for short-term investors and as an early financial distress warning system for long-term shareholders. Forecasting accuracy is the most important factor in selecting any forecasting methods. Research efforts in improving the accuracy of forecasting models are increasing since the last decade. The appropriate stock selection, those are suitable for investment is a very difficult task. The key factor for each investor is to earn maximum profits on their investments with minimum risk. In our project we have employed several methods/algorithms for forecasting stock prices, which are ARIMA, GARCH, Prophet, K-Nearest Neighbours regression, Feed Forward Neural Networks and ETS. Further, we have compared the respective accuracies to determine the best model. These methods are applied on Amazon Stock Price data, spanning 5 years, retrieved from Yahoo Finance API. The results will be used to analyze the stock prices and their prediction in depth, in future research efforts. The project serves as a foundation for democratizing machine learning technologies to the general public in the context of discovering investment opportunities. It paves the way for extending and testing out new models, and developing AutoML in the financial context, in the future.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Stock price is the price of a single stock among the number of stocks sold by a company listed in public offering. Having stocks of a public company allows you to own a portion of it. Original owners of the company initially sell the stocks to get additional investment to help the company grow. This initial offering of stocks to the public is called Initial Public Offering (IPO).
Stock prices change because of the supply and demand. Suppose, if many people are willing to buy a stock, then the price goes up as there is more demand. If more people are willing to sell the stock, the price goes down as there is more supply than the demand. Though understanding supply and the demand is relatively easy, it is hard to derive what factors exactly contribute to the increase in demand or supply. These factors would generally boil down to socioeconomic factors like market behavior, inflation, trends and more importantly, what is positive about the company in the news and what's negative.
Predicting the accurate stock price has been the aim of investors ever since the beginning of the stock market. Millions of dollars worth of trading happens every single day, and every trader hopes to earn profit from his/her investments. Investors who can make right buy and sell decisions will end up in profits. To make right decisions, investors have to judge based on technical analysis, such as company's charts, stock market indices and information from newspapers and microblogs. However, it is difficult for investors to analyze and forecast the market by churning all this information.Some of the first research in prediction of stock prices dates back to 1994, in which a comparative study with machine learning regression models was performed. Since then, many researchers were investing resources to devise strategies for forecasting the price of the stock.

1.2 Motivation

Efficient Market Hypothesis is one of the popular theories in financial economics. Prices of the securities reflect all the information that is already available and it is impossible to outperform the market consistently. There are three variants of Efficient Market Hypothesis [1] (EMH);

namely weak form, semi-strong form and the strong form. Weak form states that the securities reflect all the information that is publicly available in the past. Semi Strong form states that the price reflects all the publicly available data and also, they change instantly to reflect the newly available information. The strong form would include even the insider or private information. But this theory is often disputed and highly controversial.

The best example would be investors such as Warren Buffet, who have earned huge profits over long period of time by consistently outperforming the market. Even though predicting the trend of the stock price by manually interpreting the chaotic market data is a tedious task, with the advent of artificial intelligence, big data and increased computational capabilities, automated methods of forecasting the stock prices are becoming feasible. Machine learning models are capable of learning a function by looking at the data without explicitly being programmed. But unfortunately, the time series of a stock is not a function that can be easily mapped. It can be best described more as a random walk, which makes the feature engineering and prediction much harder. With Deep Learning, a branch of machine learning, one can start training using the raw data and the features will be automatically created when neural network learns. Deep Learning techniques are among those popular methods that have been employed, to identify the stock trend from large amounts of data but until now there is no such algorithm or model which could consistently predict the price of future stock value correctly. Lot of research is going on both in academia and industry on this challenging problem.

Chapter 2

Methodologies

2.1 Components of Time Series

- **Trend component** : The trend is the long term pattern of a time series. A trend can be positive or negative depending on whether the time series exhibits an increasing long term pattern or a decreasing long term pattern. If a time series does not show an increasing or decreasing pattern then the series is stationary in the mean.
- **Cyclical Component** : Any pattern showing an up and down movement around a given trend is identified as a cyclical pattern. The duration of a cycle depends on the type of business or industry being analyzed.
- **Seasonal Component** : Seasonality occurs when the time series exhibits regular fluctuations during the same month (or months) every year, or during the same quarter every year.
- **Irregular Component** : This component is unpredictable. Every time series has some unpredictable component that makes it a random variable. In prediction, the objective is to "model" all the components to the point that the only component that remains unexplained is the random component.

2.2 ARIMA Modeling

ARIMA models are generally used in the forecasting a time series which can be made "stationary" by differencing, whenever necessary. A time series is called *stationary* if:

1. it has no trend
2. its variations around its mean have a constant amplitude.
3. its short-term random time patterns always look the same in a statistical sense.

In terms of power spectrum representation, an ARIMA model can be viewed as a "filter" that tries to separate the signal from the noise, and then extrapolate that signal into the future to obtain forecasts. Basically, the ARIMA model is a linear regression model modified to track linear trends in stationary time series results. [2] The forecasting equation of ARIMA is represented as ARIMA(p,d,q). This equation depends on *lags of the dependent variable* and/or *lags of the forecast errors*. That is:

Predicted value of Y = a constant and/or a weighted sum of one or more recent values of Y and/or a weighted sum of one or more recent values of the errors.

The acronym **ARIMA** stands for **Auto-Regressive Integrated Moving Average**. Lags of the stationarized series in the forecasting equation are called "*autoregressive*" terms, lags of the forecast errors are called "*moving average*" terms, and a time series which needs to be differenced to be made stationary is said to be an "*integrated*" version of a stationary series. In ARIMA(p,d,q), p, q is each is the order of the AR model and MA model, and d is the order of differencing applied to the data. These all values are integers. In terms of y, the general forecasting equation is:

$$\hat{y}_t = \mu + \phi_1 y_{t-1} + .. + \phi_p y_{t-p} - \theta_1 e_{t-1} - .. - \theta_q e_{t-q}$$

The term μ is a constant; ϕ_k and θ_k are coefficients value of AR model variable y_{t-k}, and MA model variable e_{t-l}. The signs of the moving average parameters $\theta's$ are negative, following the convention introduced by Box and Jenkins We have used the Alkaline Information Criterion(AIC) for the paramter estimation step.

$$AIC = -2ln(\hat{L}) + 2k$$

The $ln(\hat{L})$ notation is the value of the likelihood function, and k is the degree of freedom, that is, the number of parameters used. A model that has a small AIC value is generally considered a better model. There are different ways to compute the likelihood function, $ln(\hat{L})$. We use the maximum likelihood estimator for the computation. This method tends to be slow, but produces accurate results.

2.3 GARCH Modeling

Autoregressive models can be developed for univariate time series data that is stationary (AR), has a trend (ARIMA), and has a seasonal component (SARIMA). One aspect of a univariate time series that these autoregressive models do not model is a change in the variance over time. Classically, a time series with modest changes in variance can sometimes be adjusted using a power transform, such as by taking the Log or using a Box-Cox transform. There are some

time series where the variance changes consistently over time. In the context of a time series in the financial domain, this would be called increasing and decreasing volatility.

2.3.1 GARCH

Generalized Autoregressive Conditional Heteroskedasticity, or GARCH, is an extension of the ARCH model that incorporates a moving average component together with the autoregressive component. Specifically, the model includes lag variance terms (e.g. the observations if modeling the white noise residual errors of another process), together with lag residual errors from a mean process. The introduction of a moving average component allows the model to both model the conditional change in variance over time as well as changes in the time-dependent variance. As such, the model introduces a new parameter "p" that describes the number of lag variance terms:

- p: The number of lag variances to include in the GARCH model.
- q: The number of lag residual errors to include in the GARCH model.

A generally accepted notation for a GARCH model is to specify the GARCH() function with the p and q parameters GARCH(p, q); for example GARCH(1, 1) would be a first order GARCH model. A GARCH(p,q) process may be written as follows:

$$\varepsilon_t = \sigma_t \eta_t$$

$$\sigma_t^2 = \omega + \sum_{i=1}^{q} \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2$$

Here, ε_t is usually the disturbance term of a conditional mean equation and $\eta_t \approx i.i.d(0,1)$. The conditional volatility process is determined linearly by it's own lagged values σ_{t-j}^2 and the lagged squared observations. It may be useful to think about the simple GARCH (1,1) specification model as a model in which the conditional variance is specified as a weighted average of the long run variance $\frac{\omega}{1-\alpha-\beta}$, the last predicted variance σ_{t-1}^2 and the new information ε_{t-1}^2.

2.3.2 EGARCH

Despite the apparent success of these simple parameterizations, the ARCH and GARCH models cannot capture some important features of the data. The most interesting feature not addressed by these models is the leverage or asymmetric effect discovered by Black (1976), and confirmed by the findings of French, Schwert, and Stambaugh (1987), Nelson (1990), and Schwert (1990), among others.1 Statistically, this effect occurs when an unexpected drop in price (bad news) increases predictable volatility more than an unexpected increase in price (good news) of similar magnitude. This effect suggests that a symmetry constraint on the conditional variance function in past ?'s is inappropriate. One method proposed to capture such

asymmetric effects is Nelson's (1990) exponential GARCH or EGARCH model. This approach directly models the algorithm of the conditional volatility :

$$\varepsilon_t = \sigma_t \eta_t$$

$$\log \sigma_t^2 = \omega + \sum_{i=1}^{q} (\alpha_i \eta_{t-i} + \gamma(|\eta_{t-i}| - E|\eta_{t-i}|)) + \sum_{j=1}^{q} \beta_j \log \sigma_{t-j}^2$$

where, E is the expectation operator.This model formulation allows multiplicative dynamics in evolving the volatility process. Asymmetry is captured by the α_i parameter, a negative value indicates that the process react more to negative shocks, as observable in real data sets.

2.3.3 TGARCH

Another volatility model commonly used to handle leverage effects is the threshold GARCH (or TGARCH) model; see Glosten, Jagannathan, and Runkle (1993) and Zakoian (1994). The TGARCH specification involves an explicit distinction of model parameters above and below a certain threshold. The TGARCH model can be formulated as follows :

$$\varepsilon_t = \sigma_t \eta_t$$

$$\sigma_t^2 = \omega + \sum_{i=1}^{q} (\alpha_i + \gamma_i I_{t-i}) \varepsilon_{t-i}^2 + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2$$

, where

$$I_{t-i} = \begin{cases} 1 & \text{if } \varepsilon_{t-i} < 0 \\ 0 & \text{if } \varepsilon_{t-i} > 0 \end{cases}$$

The interpretation is straightforward; the ARCH coefficient depends on the sign of the previous error term; if γ_1 is positive, a negative error term will have a higher impact on the conditional volatility, just as we seen in the leverage effect before.

2.4 Prophet Forecasting

The origin of prophet comes from the application of a forecasting model into supply chain management, sales and economics. This model helps with a statistical approach in shaping business decisions.The Prophet model has been developed by Facebook's Core Data Science team and it is an open-source tool for business forecasting. The important idea in Prophet is that by doing a better job of fitting the trend component very flexibly, we more accurately model seasonality and the result is a more accurate forecast. We prefer to use a very flexible regression model (somewhat like curve-fitting) instead of a traditional time series model for this task because it gives us more modeling flexibility, makes it easier to fit the model, and handles missing data or outliers more gracefully.

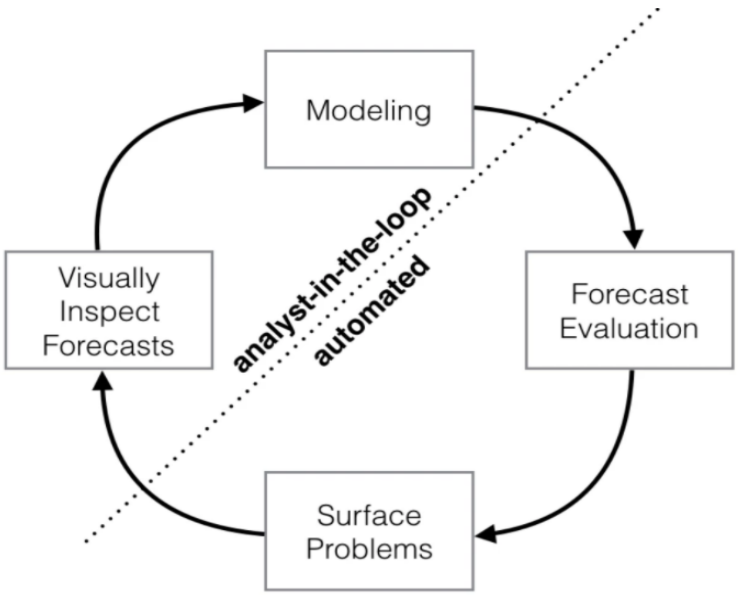

Figure 2.1: Prophet Forecast Flow

2.5 KNN Regression Time Series Forecasting

Time series forecasting has been performed traditionally using statistical methods such as ARIMA models or exponential smoothing. However, the last decades have witnessed the use of computational intelligence techniques to forecast time series. Although artificial neural networks is the most prominent machine learning technique used in time series forecasting, other approaches, such as Gaussian Process or KNN, have also been applied. Compared with classical statistical models, computational intelligence methods exhibit interesting features, such as their nonlinearity or the lack of an underlying model, that is, they are non-parametric. KNN is a very popular algorithm used in classification and regression. This algorithm simply stores a collection of examples. Each example consists of a vector of features (describing the example) and its associated class (for classification) or numeric value (for prediction). Given a new example, KNN finds its k most similar examples (called nearest neighbors), according to a distance metric (such as the Euclidean distance), and predicts its class as the majority class of its nearest neighbors or, in the case of regression, as an aggregation of the target values associated with its nearest neighbors. In this paper we describe the tsfknn R package for univariate time series forecasting using KNN regression.

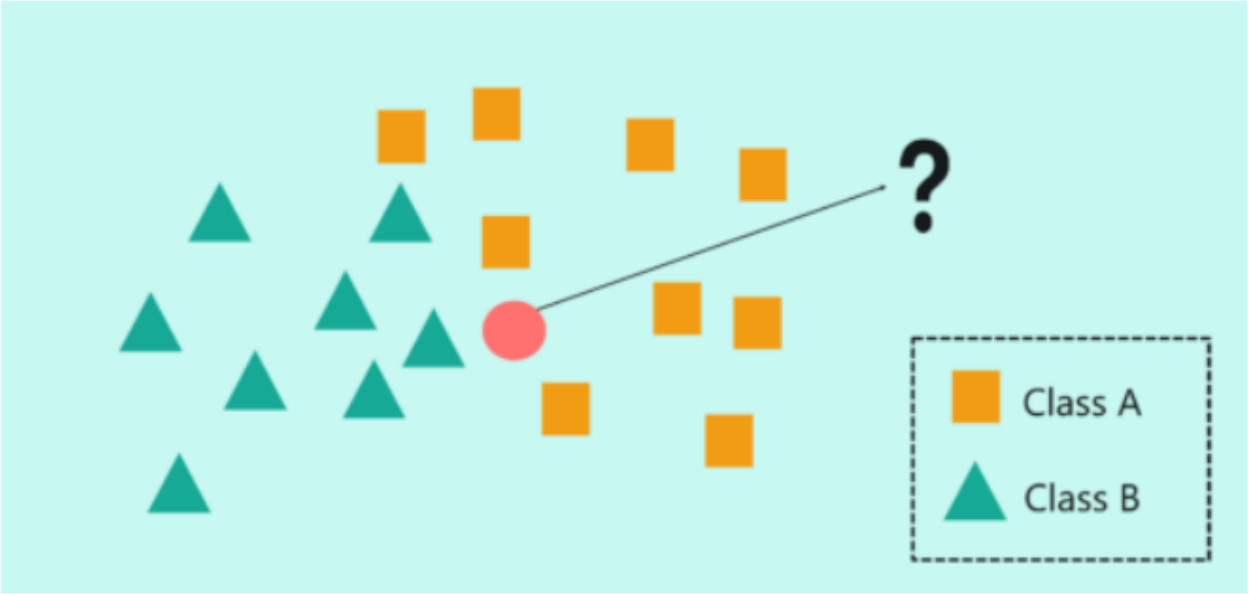

Figure 2.2: KNN model

2.6 Feed Forward Neural Network

A feed-forward neural network (typically multi-layer) is a type of supervised learner that will adjust the network weights on its input and internal nodes, in an iterative manner, in order to minimize errors between predicted and actual target variables. It commonly uses stochastic gradient descent (sometimes called error back propagation) over many iterations in order to find a local minimum of the error response and optimize the network weights accordingly. The basic idea behind stochastic gradient descent is to start by randomizing the weights, then adjust them by iterating through several passes and updating the weights in a direction that moves the total error between target and predicted errors towards the local minimum error of the gradient surface. In practice, a tradeoff is found between optimizing a training set against a validation set, in order to reduce the problem of over-fitting[3]. Researching deeply in new machine learning models we have reached some new neural network function in the forecast package called nnetar. A single hidden layer neural network is the most simple neural networks form. In this single hidden layer form there is only one layer of input nodes that send weighted inputs to a subsequent layer of receiving nodes. This nnetar function in the forecast package fits a single hidden layer neural network model to a timeseries. The function model approach is to use lagged values of the time series as input data, reaching to a non-linear autoregressive model. We use Box-Cox Transformation to ensure the residuals will be roughly homoscedastic. The model can be written as :

$$y_t = f(y_{t-1}) + \varepsilon_t$$

where, y_{t-1} is a vector containing lagged values of the series and f is the neural network. The error series ε_t is assumed to be homoscedastic (and possibly also normally distributed).

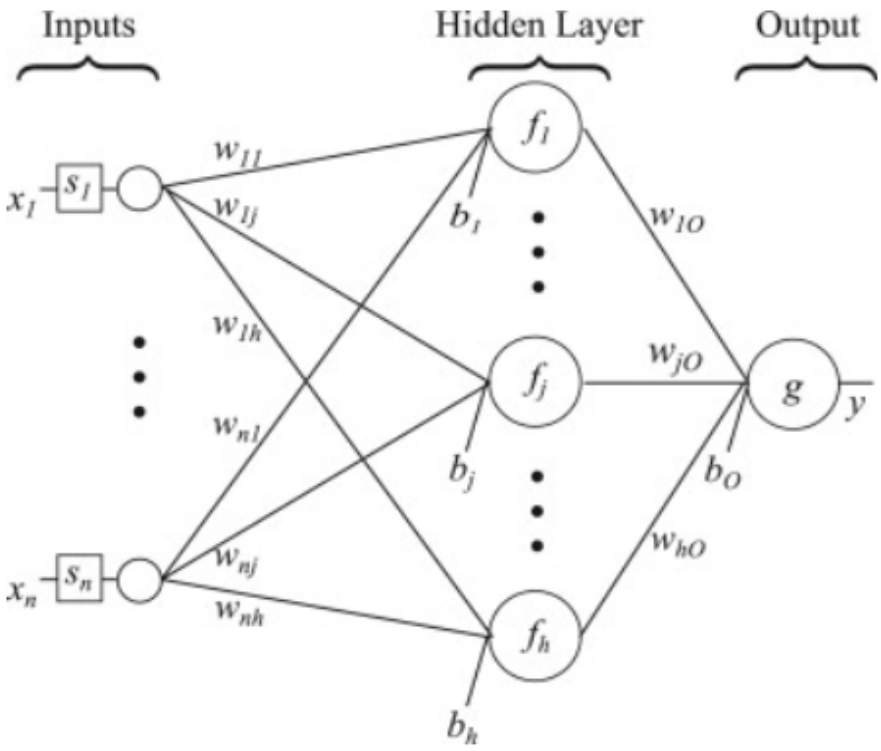

Figure 2.3: Neural Network Structure

To calculate the number of hidden nodes in the hidden layer, the following formula has been used:

$$N_h = \frac{N_s}{(\alpha * (N_i + N_o))}$$

$$N_i = number.of.input.neurons.$$

$$N_o = number.of.output.neurons.$$

$$N_s = number.of.train.samples$$

$$\alpha = 1.5^{-10}$$

Figure 2.4: Hidden Nodes Calculation

2.7 ETS Modeling

ETS (Error, Trend, Seasonal) method is an approach used for forecasting time series univariate time series. It focuses on the trend and seasonality in the data and thus defines how these

unobserved components (error, trend and seasonality) change over time[4]. The flexibility of the ETS model lies in its ability to trend and seasonal components of different traits. By considering variations in the combinations of the trend and seasonal components, nine exponential smoothing methods are possible. Each method is labelled by a pair of letters (T,S) defining the type of 'Trend' and 'Seasonal' components. For each method there exist two models: one with additive errors and one with multiplicative errors. To distinguish between a model with additive errors and one with multiplicative errors (and also to distinguish the models from the methods), we add a third letter to the classification. The possibilities for each component are: Error = (A,M) , Trend = (N,A,Ad) and Seasonal= (N,A,M). Hence, we have a total of 18 ETS models that can be used for forecasting. A great advantage of the ETS statistical framework is that information criteria can be used for model selection. The AIC (Akaike Information Criterion), AICc (Akaike Information Criterion Corrected) and BIC (Bayesian Information Criterion) can be used here to determine which of the ETS models is most appropriate for a given time series.

It is a commonly held myth that ARIMA models are more general than exponential smoothing. While linear exponential smoothing models are all special cases of ARIMA models, the non-linear exponential smoothing models have no equivalent ARIMA counterparts. On the other hand, there are also many ARIMA models that have no exponential smoothing counterparts. In particular, all ETS models are non-stationary, while some ARIMA models are stationary. ETS package in R determines the optimum model by itself taking AIC, AICc and BIC into consideration. Three optimum models, suggested by the ETS package, used for forecasting are.

We have employed 3 ETS models:

- **MNN** : a model with multiplicative errors (M), but no overall trend (N) or seasonality (N) assumed.
- **MAN** : a model with multiplicative errors (M), additive trend (A) and no seasonality assumed.
- **MAdN** : a model with multiplicative errors (M), additive and damped trend but no seasonality assumed.

Chapter 3

Analysis and Results

3.1 Initial Analysis

3.1.1 Visualisation

The figure 3.1 shows the continuous line graph from the Amazon Stock Closing Price. We see a sharp rise in the graph after October 2017 which indicates that the company blossomed after October 2017, since its share price increased.

Figure 3.1: Closing Prices AMAZON 2010-2020

To understand and visualise the sharp increase in the above graph, we plotted the curve from 2018. In the year 2020, there was a sharp increase further, after July.

Figure 3.2: Closing Prices AMAZON 2018-2020

The second chart series (Figure3.7) show the Bollinger Band chart, Bollinger change, Volume Traded and Moving Average Convergence Diverence. The moving average is important to understanding Amazon(AMZN)'s technical charts. It smoothes out daily price fluctuations by averaging stock prices and is effective in identifying potential trends. The Bollinger Band chart plots two standard deviations away from the moving average and is used to measure the stock's volatiliy[5]. The Volume chart shows how its stocks are traded on the daily. The Moving Average Convergence Divergence gives technical analysts buy/sell signals. The rule of thumb is: If it falls below the line, it is time to sell. If it rises above the line, it is experiencing an upward momentum.

Figure 3.3: Bollinger Band chart

3.1.2 Additive and Multiplicative Decomposition

The components of a time series can be an additive or multiplicative. The multiplicative model is preferred when the magnitude of the seasonal pattern increases or decreases with the increase or decrease in the data values. The additive model is preferred when the magnitude of the seasonal pattern does not correlate with the data values.

The Additive Model : $T_t + S_t + X_t + Z_t$

The Multiplicative Model : $T_t * S_t * X_t * Z_t$

Here, we have chosen to use the multiplicative model with respect to our dataset.

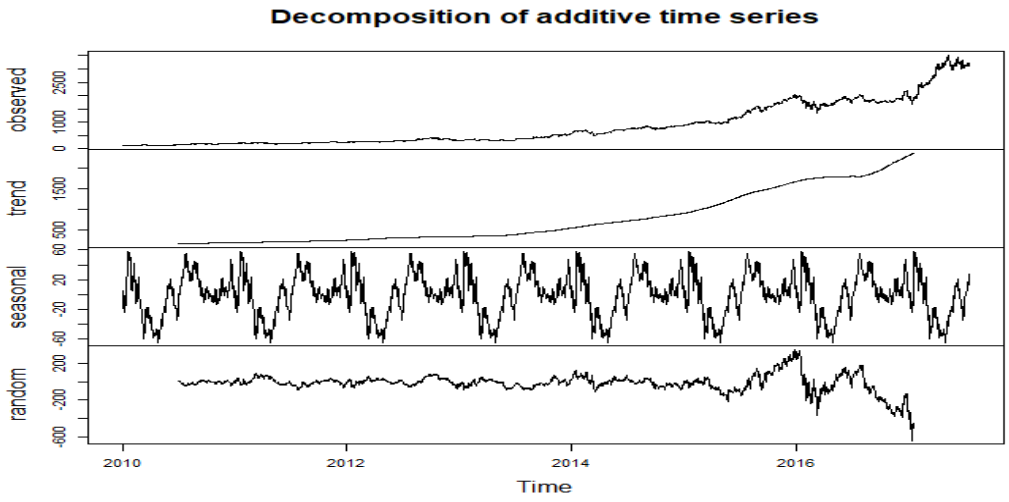

Figure 3.4: Additive Decomposition of Time Series

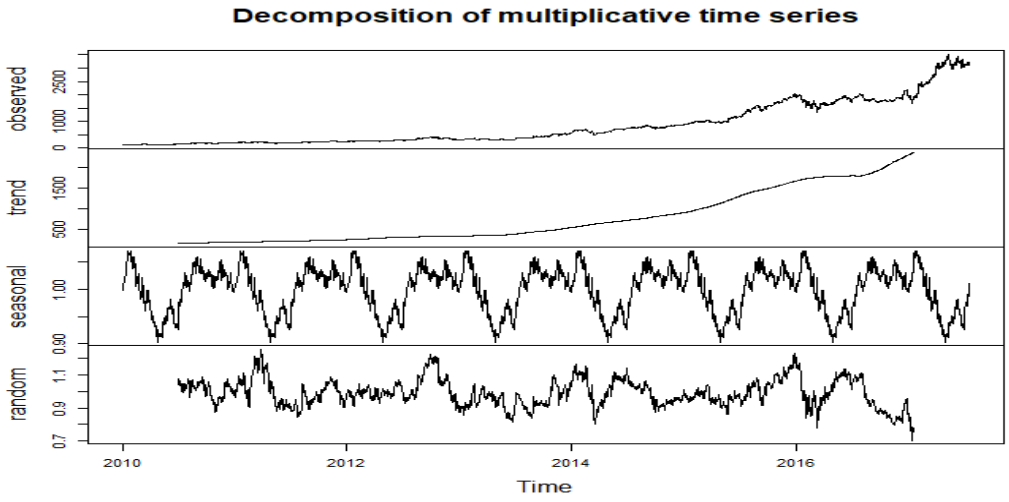

Figure 3.5: Multiplicative Decomposition of Time series

3.1.3 Augmented Dickey Fuller Test

The Augmented Dickey Fuller Test (ADF) is a unit root test for stationarity. Unit roots can cause unpredictable results in the time series analysis. The Augmented Dickey-Fuller test can be used with serial correlation. The ADF test can handle more complex models than the Dickey-Fuller test, and is also more powerful. The hypothesis for the test:

- The null hypothesis for this test is that there is a unit root. (The null hypothesis is that the data is non-stationary.)
- The alternate hypothesis differs slightly according to which equation you're using. The basic alternate is that the time series is stationary.

We perform the ADF test on our dataset and observe our p-value to be very high which denotes non-stationarity, so to make it stationary we perform a log transformation.

```
> print(adf.test(AMZN_Close_Prices))

        Augmented Dickey-Fuller Test

data:  AMZN_Close_Prices
Dickey-Fuller = 0.038888, Lag order = 14, p-value = 0.99
alternative hypothesis: stationary
```

Figure 3.6: ADF before Log Transformation

```
> print(adf.test(logs))

        Augmented Dickey-Fuller Test

data:  logs
Dickey-Fuller = -15.46, Lag order = 13, p-value = 0.01
alternative hypothesis: stationary
```

Figure 3.7: ADF after Log Transformation

The p values should be made as low as possible. It is clearly observable that after taking the log of the dataset, the p value becomes significantly less (=0.01), hence denoting a stationary time series.

3.1.4 ACF-PACF Plots

Autocorrelation refers to how correlated a time series is with its past values. As we know in AR models, the ACF will dampen exponentially. The ACF is the plot used to see the correlation between the points, up to and including the lag unit. We can see that the autocorrelations are significant for a large number of lags, but perhaps the autocorrelations at posterior lags are merely due to the propagation of the autocorrelation at the first lags. For identifying the (p)

order of the AR model we use the PACF plot. For MA models we will use ACF plot to identify the (q) order and the PACF will dampen exponentially. If we look the PACF plot in figure 3.20, we can note that the it has a significant spike only at first lags, meaning that all the higher-order autocorrelations are effectively explained by the first lag autocorrelation.

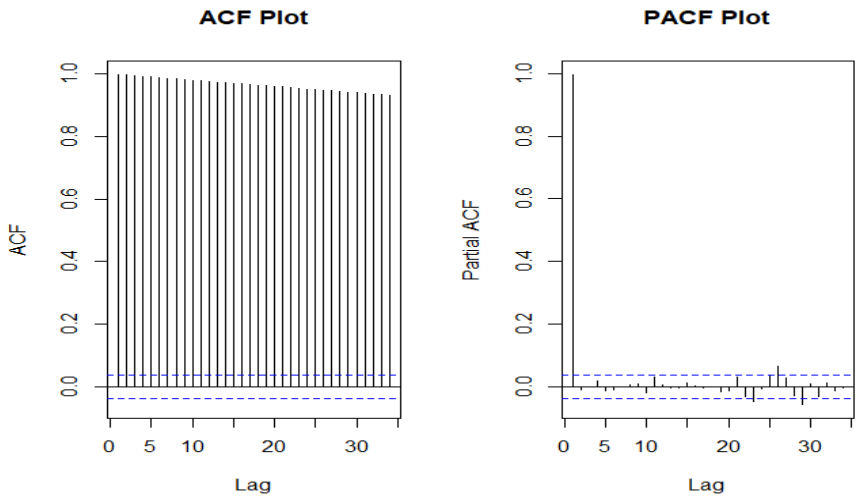

Figure 3.8: ACF-PACF Plot

3.1.5 Log Transformation

As we saw earlier in the ADF test, our dataset is not stationary, so to make it stationary we have taken the log transformation. The transformation eliminates the area which is non-stationary. Following is the graph 3.9 for the log returns plot :

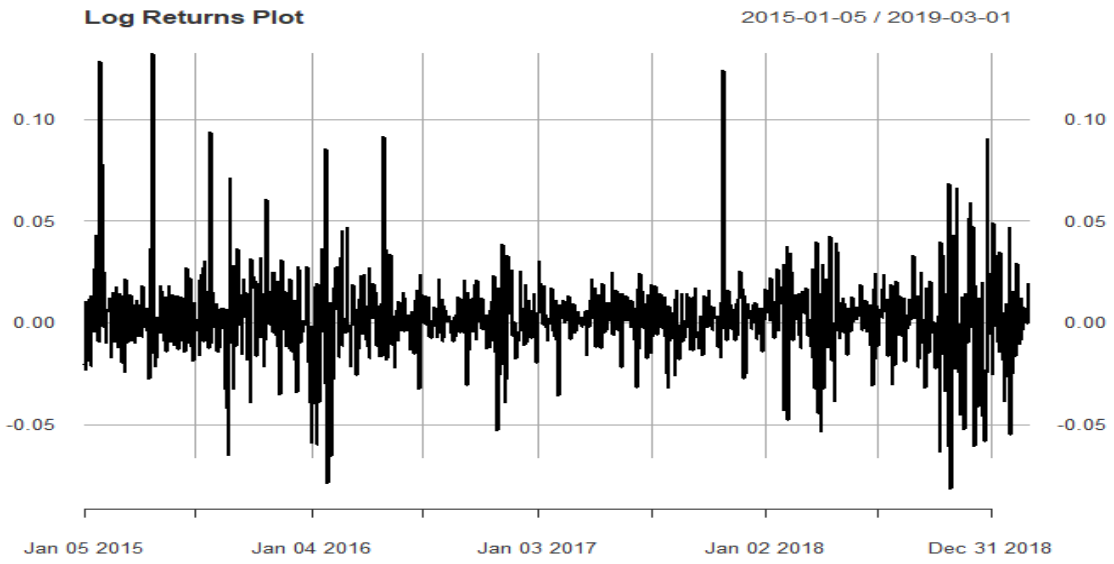

Figure 3.9: Log Returns Plot

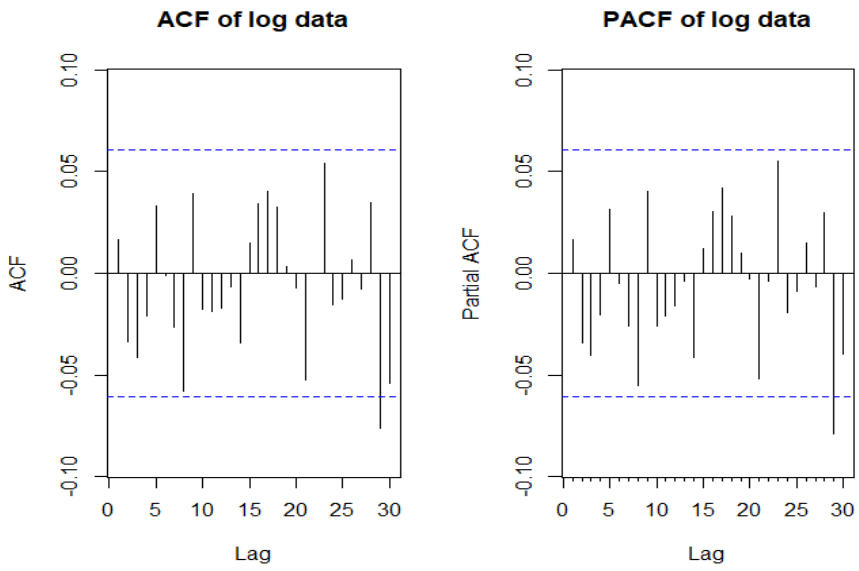

Figure 3.10: ACF-PACF Plot of Logs

3.1.6 Histogram and Empirical Distribution

Because the data set over a five-year period of time is relatively large, the histogram is most effective for this purpose. The histogram is used to determine whether or not the data are approximately normally distributed. A bell shaped histogram gives an indication that the data are normally distributed. The histogram of the Closing Price of the AMAZON STOCK have a unimodal form. They also have a bell shaped form, symmetry around the mean and is normally distributed.

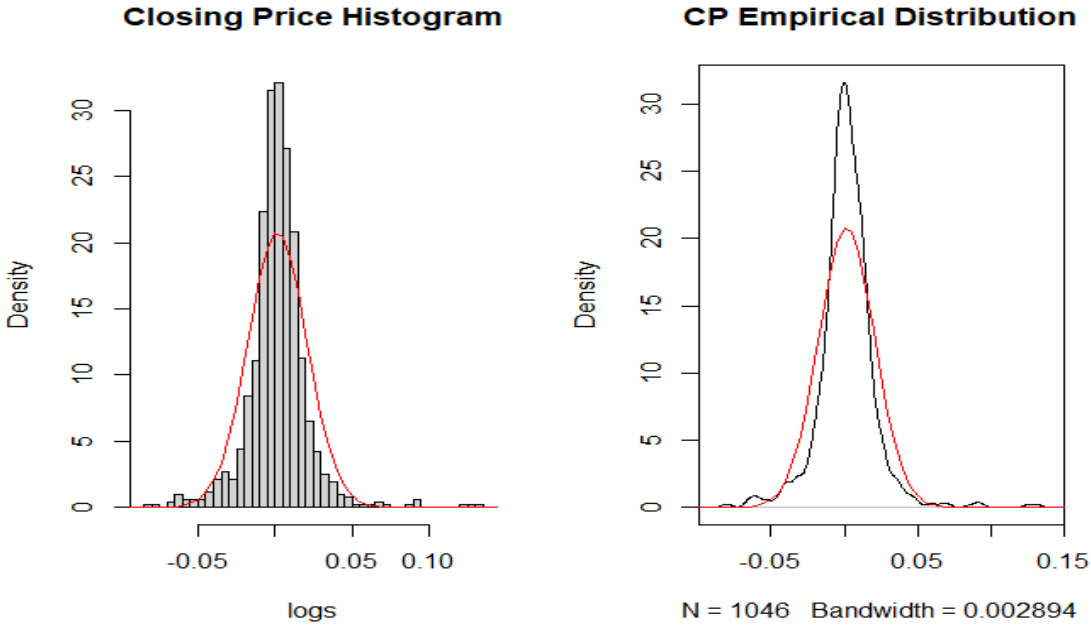

Figure 3.11: Empirical Distribution Plot

3.2 ARIMA Forecasting

Before starting to fit the models, we have performed a train-test split on the AMAZON Closing Price Stock Data. Train consists of 70 per-cent of the total data which the test consists of 30 per-cent. We are using AUTO-ARIMA function that gives us the better approach to the dataset, we will not deep the analysis on finding model parameters.

```
> auto.arima(AMZN_CP, seasonal=FALSE)
Series: AMZN_CP
ARIMA(1,1,1) with drift

Coefficients:
          ar1      ma1   drift
      -0.5901   0.5218  1.8603
s.e.   0.1248   0.1303  0.8033

sigma^2 estimated as 1056:  log likelihood=-7329.06
AIC=14666.12   AICc=14666.15   BIC=14687.37
```

Figure 3.12: ARIMA(1,1,1) Fitting

The next step would be to predict the future stock prices. We have done the forecasting of this model for the next 30 days, refer to image 3.13.

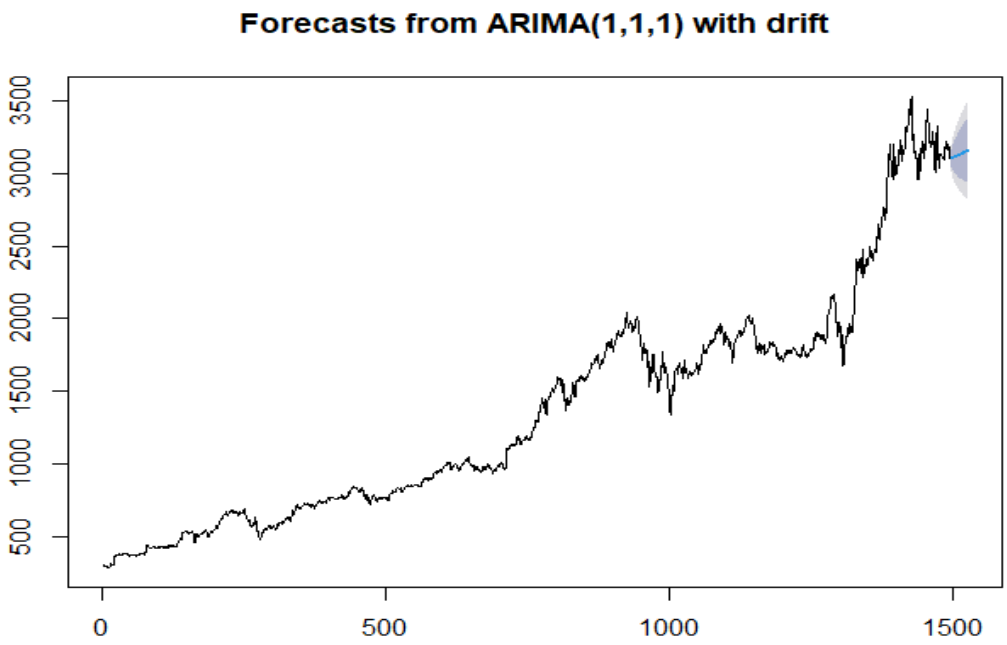

Figure 3.13: Forecast Plot of ARIMA(1,1,1)

Following is the figure indicating the accuracy of the above model, which is ARIMA(1,1,1):

```
                     ME     RMSE      MAE          MPE     MAPE     MASE
Training set 0.003663059 32.4436 18.9587 -0.06277335 1.320833 0.994797
```

Figure 3.14: Accuracy of ARIMA(1,1,1) model

Next, we have fitted another ARIMA model with the parameter lambda=auto, which is as follows:

```
> auto.arima(AMZN_CP, lambda = "auto")
Series: AMZN_CP
ARIMA(2,1,2) with drift
Box Cox transformation: lambda= -0.1976275

Coefficients:
          ar1      ar2     ma1     ma2  drift
      -0.3073  -0.8811  0.3201  0.9114  4e-04
s.e.   0.1380   0.1063  0.1240  0.0902  1e-04

sigma^2 estimated as 2.384e-05:  log likelihood=5841.86
AIC=-11671.71   AICc=-11671.66   BIC=-11639.85
```

Figure 3.15: ARIMA (2,1,2) Fitting

The next step would be to predict the forecast. We have done the forecasting of this model for the next 30 days.

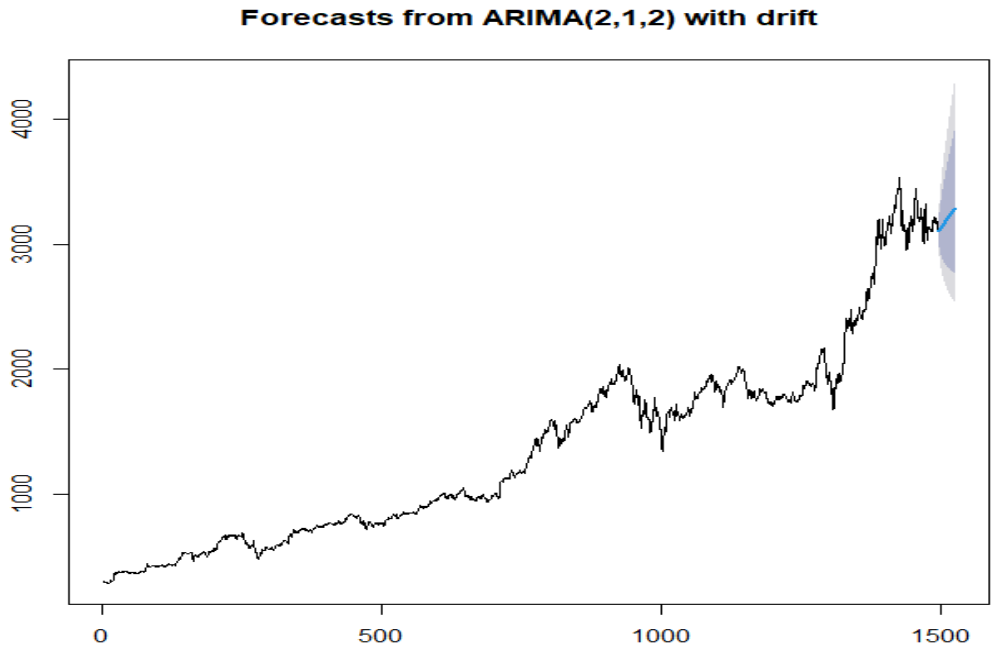

Figure 3.16: Forecast Plot of ARIMA(2,1,2)

Following is the figure for the accuracy of the above model, which is ARIMA(2,1,2) fitting:

```
                     ME      RMSE       MAE          MPE     MAPE      MASE
Training set -0.44928 32.55721 19.00797 -0.02381798 1.316943 0.9969176
                    ACF1
Training set -0.07713581
```

Figure 3.17: Accuracy of ARIMA(2,1,2) model

After studying the significant lags with the help of ACF and PACF model we further optimised the ARIMA model. The optimum model came out to be ARIMA(8,2,8). Following is the time series display (Fig. 3.18) of model residuals calculated using the above ARIMA model.

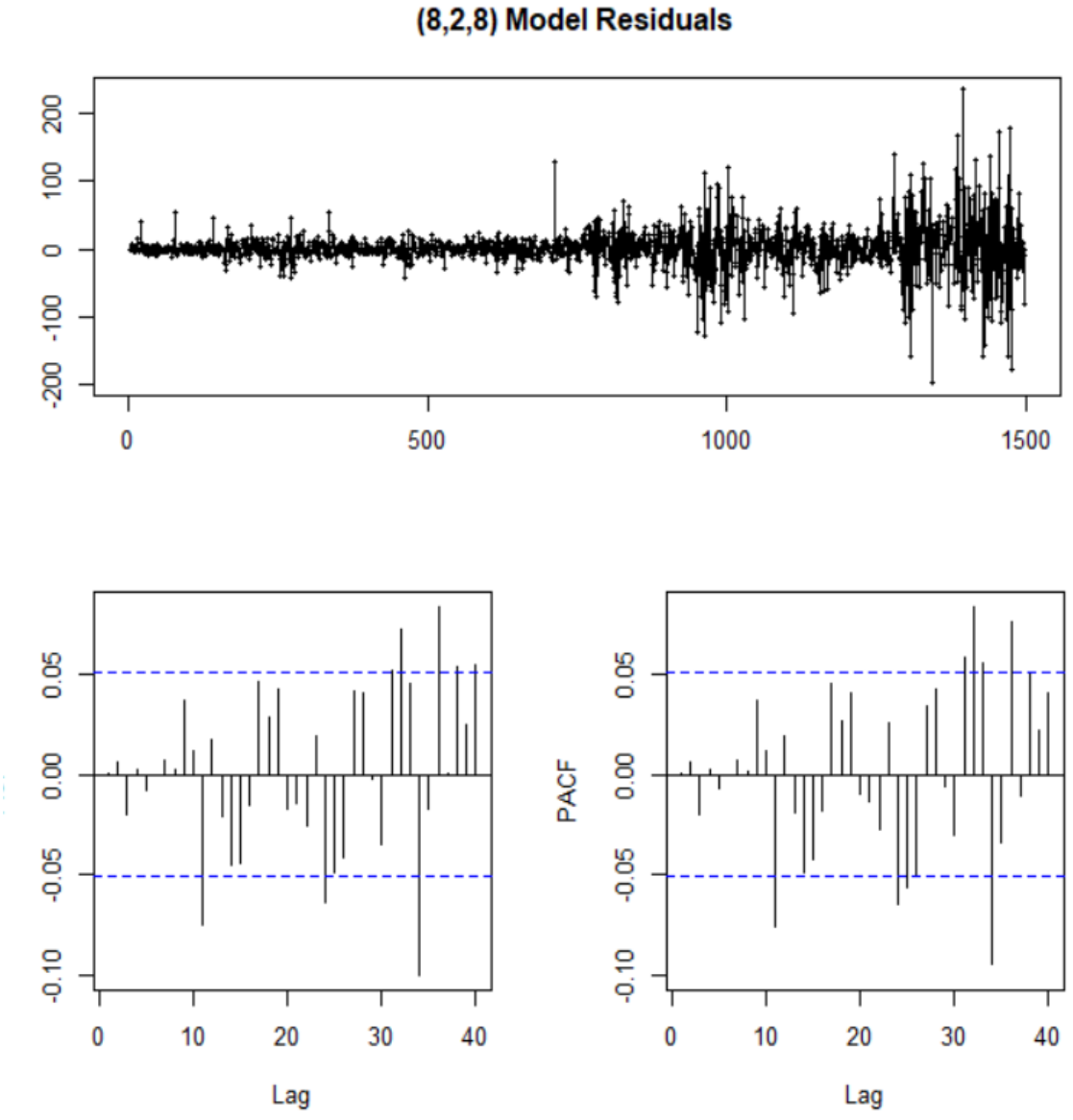

Figure 3.18: ARIMA (8,2,8) Residuals Plot

Using the above model, following is the forecast for next 30 days:

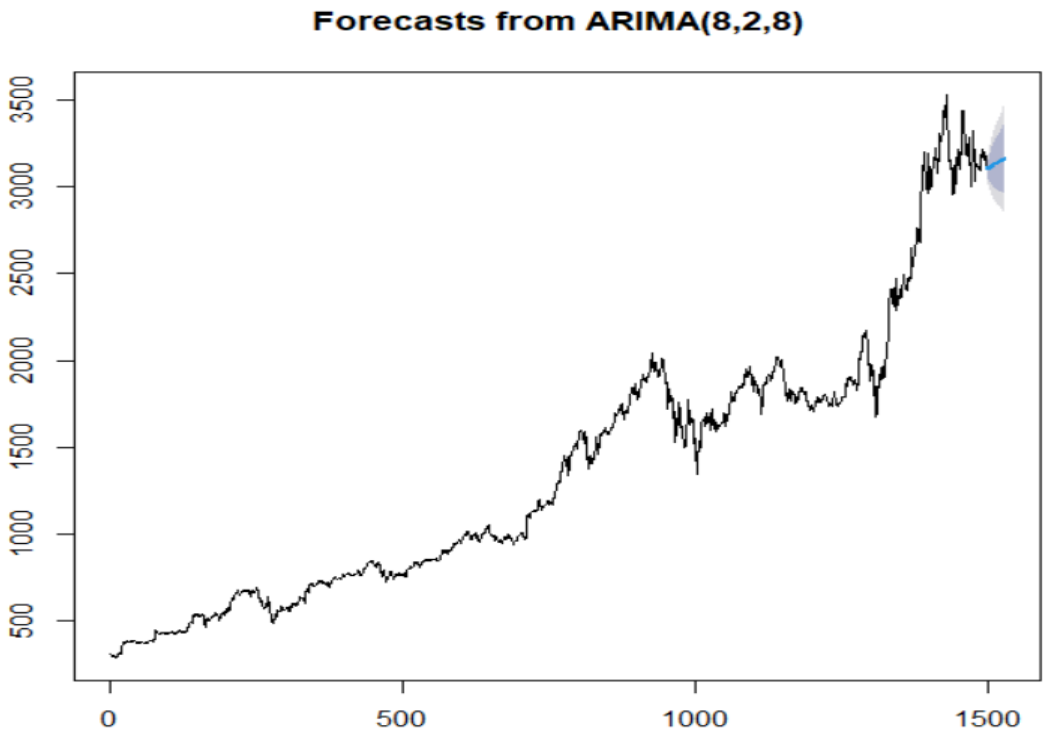

Figure 3.19: Forecasting Plot for ARIMA(8,2,8)

Following is the figure for the accuracy of the above model, which is ARIMA(8,2,8) fitting:

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
Training set	0.6519538	31.90567	18.77074	0.01761751	1.312195	0.9849348	0.0006107156

Figure 3.20: Accuracy of ARIMA(8,2,8) model

3.3 GARCH Forecasting

It is a well-known and commonly accepted stylized fact in empirical finance that the volatility of financial time series varies over time.From evaluating GARCH models implementation we will take the normal residuals and then square them. By doing this residuals plots, any volatile values will visually appear. We try to apply a standard GARCH(1,1) model over ARMA(2,2), looking if we have improved our accuracy and model parameters. Firstly, we run the AUTO-ARFIMA function to find the ARFIMA parameters, which are as follows :

```
*----------------------------------*
*          ARFIMA Model Fit        *
*----------------------------------*
Mean Model       : ARFIMA(2,0,2)
Distribution     : norm

Optimal Parameters
------------------------------------
       Estimate  Std. Error    t value Pr(>|t|)
mu    309.25978    0.001480 208954.88        0
ar1     0.00000          NA        NA       NA
ar2     0.98026    0.000011  92045.14        0
ma1     1.10947    0.000019  58346.34        0
ma2     0.10267    0.000001 131656.42        0
sigma  23.07094    0.036396    633.88        0
```

Figure 3.21: ARFIMA Fitting Parameters

With the parameters collected we choose ARFIMA (2,0,2) and incorporate the parameters to a garch model. After fitting the model, we make the conditional volatility plot. As we can see, the last years of our data have higher peaks, explained by the economic inestability in markets this last years.

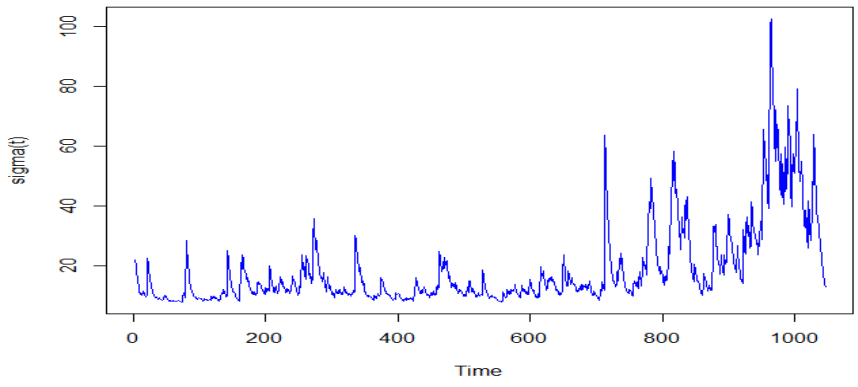

Figure 3.22: Conditional Volatility Plot

The criteria for estimating the order of Autoregressive Process are given as follows:

```
Akaike         8.359802
Bayes          8.392896
Shibata        8.359713
Hannan-Quinn   8.372351
```

Figure 3.23: Info criteria of AR

We now proceed to plot the residuals, we plot the normal residuals and the standardised residuals. By doing these residuals plots, any volatile values will visually appear.

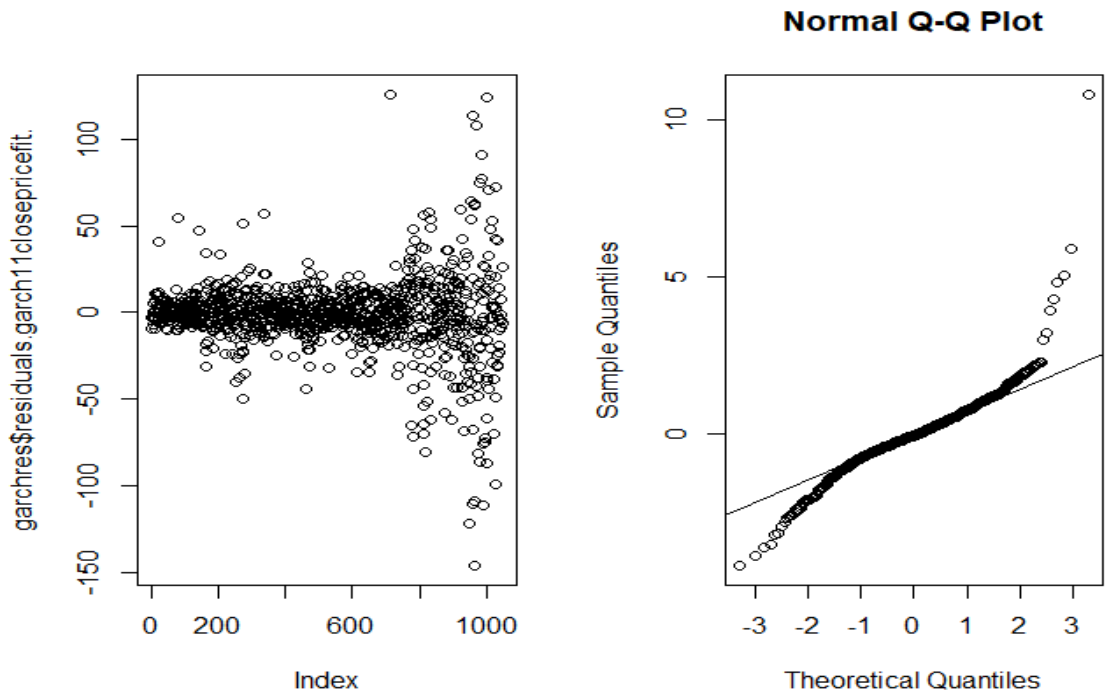

Figure 3.24: Normal and Standardised Residual plots

We see that we have some extreme values that are out of the normal distribution, but the majority of the data points are centered in the line. Having the normality plot of our standardized residuals we make a Ljung-Box test to the squared standardized residuals.

```
        Box-Ljung test

data:  garchres$residuals.garch11closepricefit..standardize...TRUE..2
X-squared = 0.23212, df = 1, p-value = 0.63
```

Figure 3.25: Ljung-Box test

With Ljung Box test we can see that our standardized squared residuals does not reject the null hypothesis, confirming that we are not having autocorrelation between them. As we ex-

plained before with our model volatility, we can see that our model residuals are bigger in the last years data. This can be caused by higher data volatility in 2018 and 2019. As we found our volatility and residuals behavior we can proceed forecasting our next 30 days and compare to the other models.

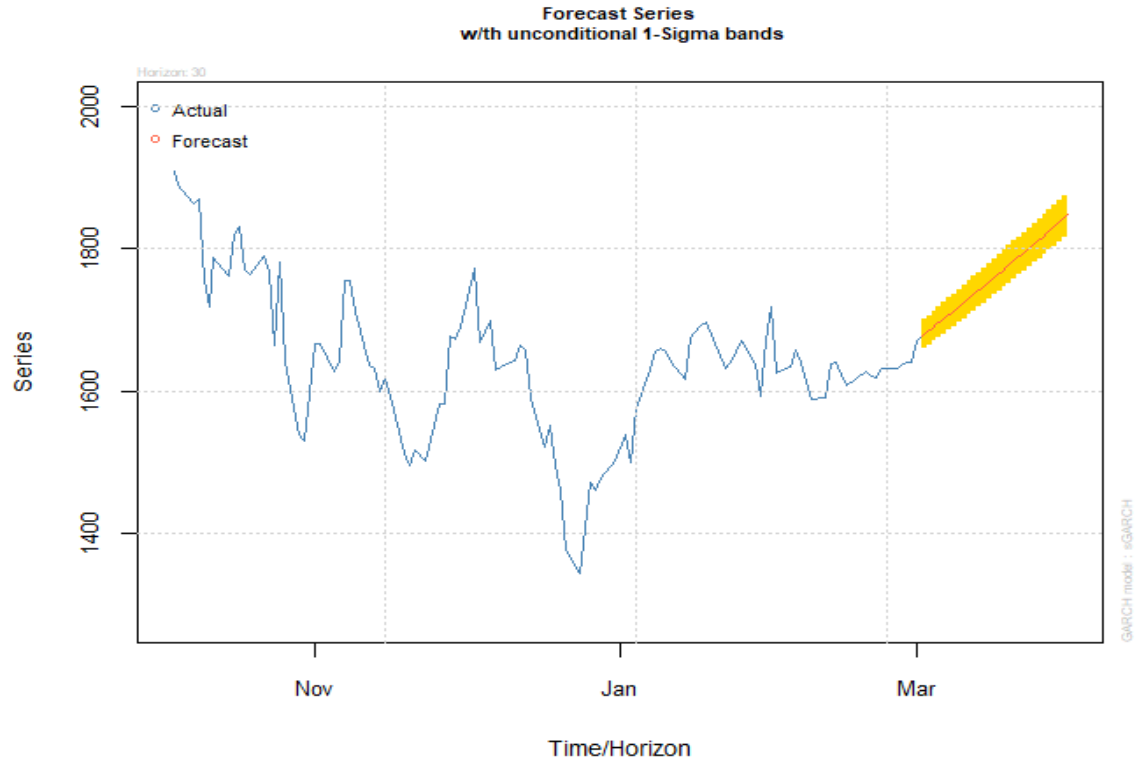

Figure 3.26: Forecasting using GARCH

3.3.1 News Impact Curve

GARCH - The news impact curve relates past return shocks (news) to current volatility. This curve measures how new information is incorporated into volatility estimates. No asymmetries are present in response to positive and negative shocks. Now, we turn to models to be able to incorporate asymmetric effects as well.

EGARCH - The EGARCH model allows good news and bad news to have a different impact on volatility, while the standard GARCH model does not, and the EGARCH model allows big news to have a greater impact on volatility than the standard GARCH model. News impact curve reflects the strong asymmetry in response of conditional volatility to shocks and confirms the necessity of asymmetric models.

TGARCH - The news impact curve for a Threshold-GARCH is less flexible in representing different responses, there is a kink at the zero point.

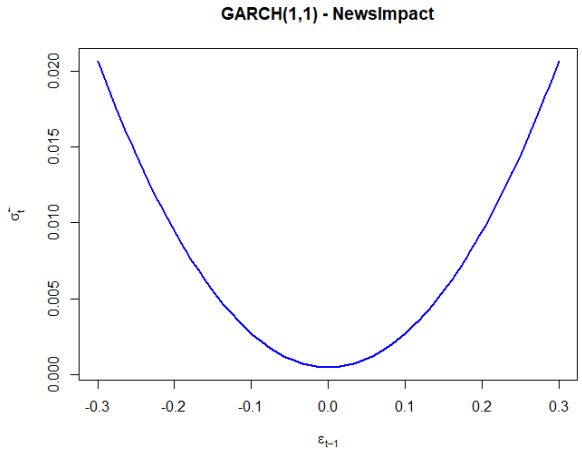

Figure 3.27: News Impact curve of GARCH

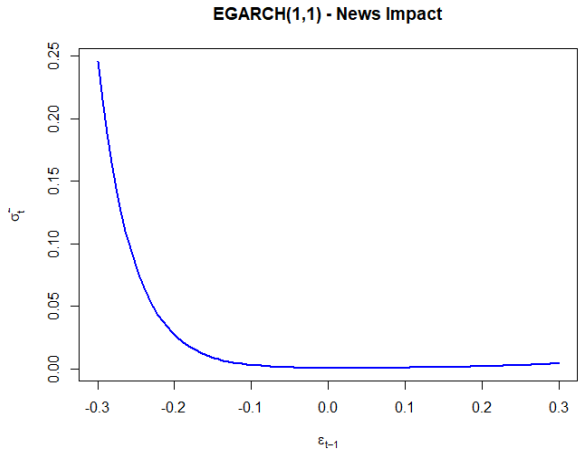

Figure 3.28: News Impact curve of EGARCH

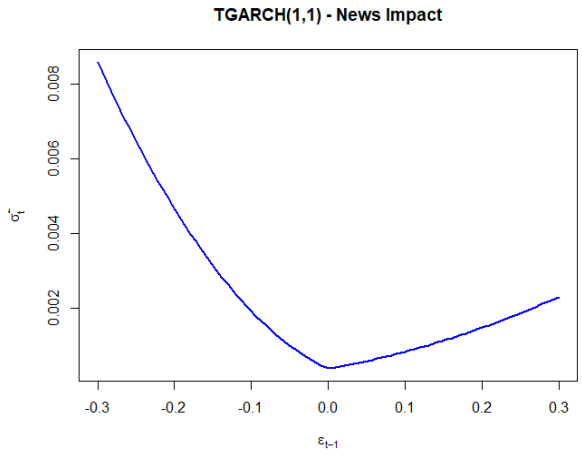

Figure 3.29: News Impact curve of TGARCH

3.4 Prophet Forecasting

The fist step before forecasting using prophet is to convert the dataset into the format that Prophet input requires.Once we have converted the data we can proceed to apply the model with the dataset and predict future values.

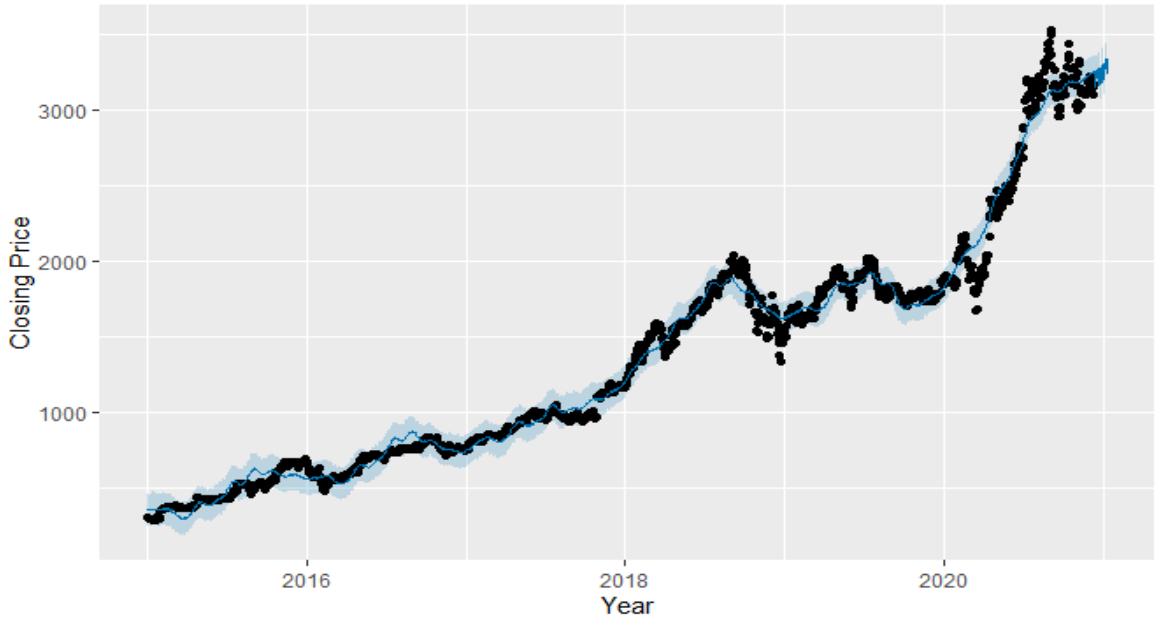

Figure 3.30: Prophet Forecasting

With the model applied and the forecast plotted we proceed to calculate the model performance. As we are new into this model application we will use the accuracy function to compare the real values against the estimated values of the train set. The correct approach for doing this in Prophet is to create a cross-validation process and analyse the model performance metrics, but we are trying to compare the arima vs the other models with the same approach.

	ME	RMSE	MAE	MPE	MAPE
Test set	-0.003406316	82.43328	60.35623	-0.2890445	5.387829

Figure 3.31: Prophet Forecasting Accuracy

Finally for a better understanding of the dataset we can plot our prophet components divided by a trend component, weekly seasonality and yearly seasonality.

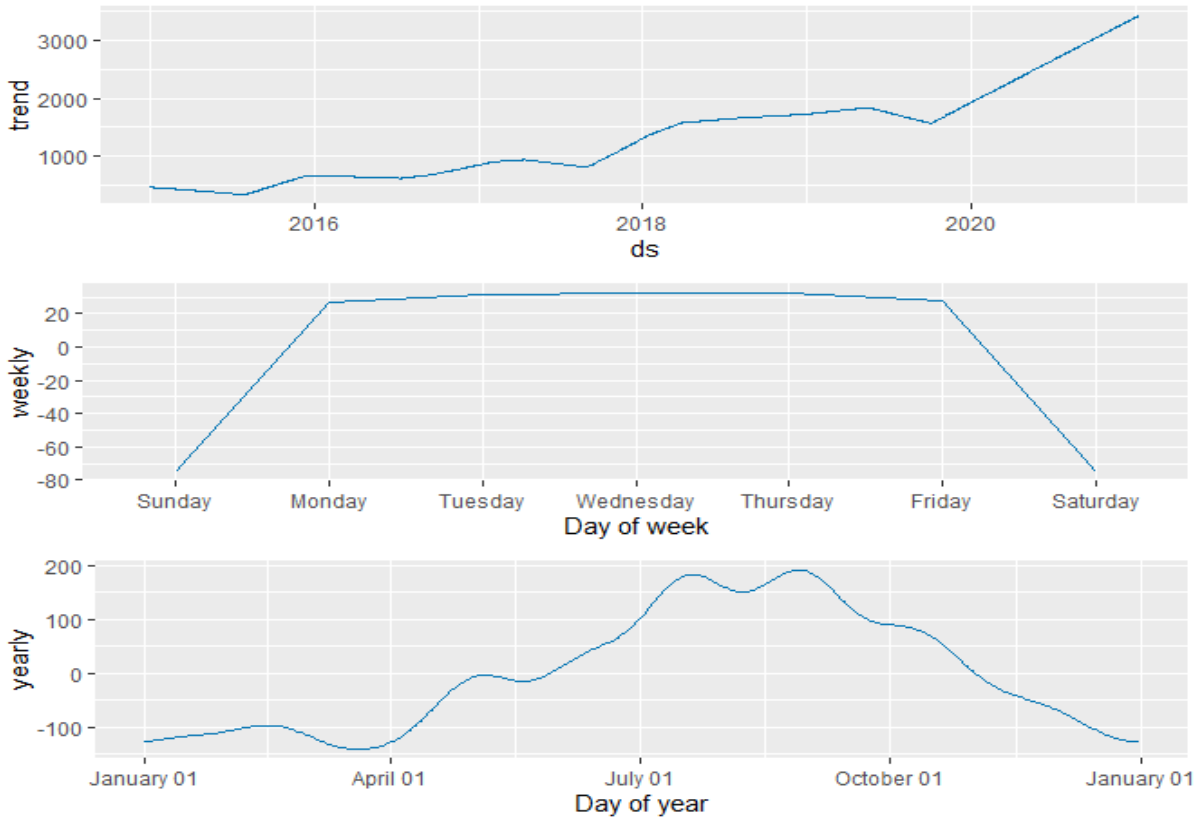

Figure 3.32: Prophet Forecasting Decomposition

3.5 KNN Regression Forecasting

Our main objective is trying to forecast new values of our stock price using a KNN experimental approach.Once we have loaded our package we proceed to forecast the 30 next daily values for the close price dataset converted to dataframe.For this prediction we will use a k equals to 40 as an experimental value because we did an heuristic approach trying to find the best k value. KNN algorithms require tunning experiments but as this study is to show the different forecasting tools we will aim deeply on the application than the tunning of the model for getting the best accuracy.

RMSE	MAE	MAPE
88.314747	77.733127	2.485393

Figure 3.33: KNN Accuracy

Using the rolling origin function we use the model and the time series asociated to asses the forecasting accuracy of the model.Once we have studied our model we can plot our predictions in the following graph.

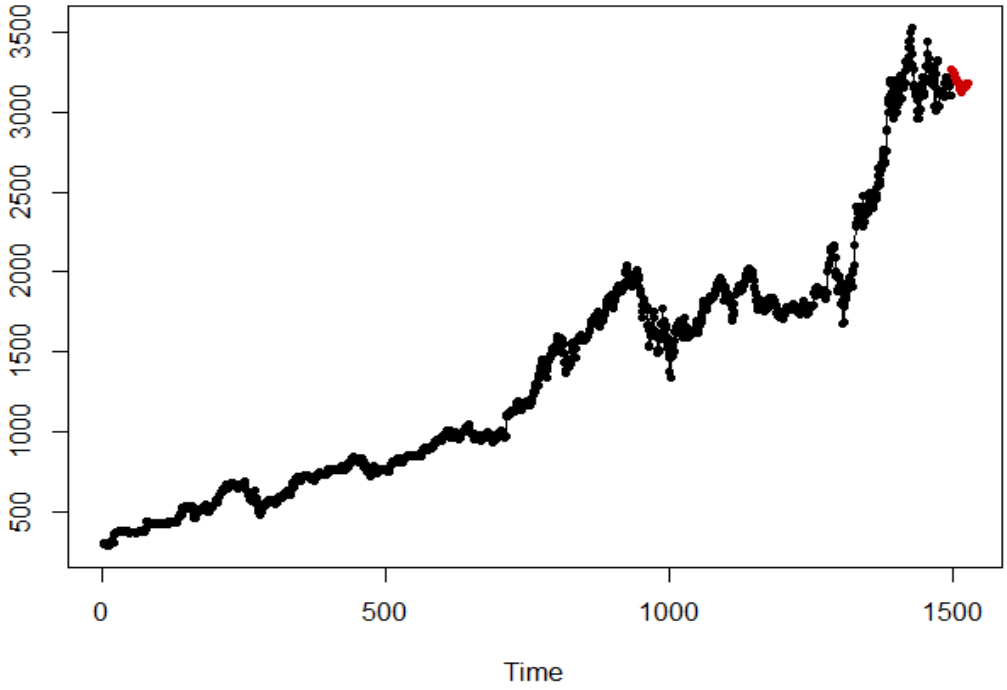

Figure 3.34: KNN Forecast

3.6 Feed Forward Neural Network

We use the NNETAR package in R to carry out the neural network implementation,The nnetar function in the forecast package for R fits a neural network model to a time series with lagged values of the time series as inputs (and possibly some other exogenous inputs). So it is a non-linear autogressive model, and it is not possible to analytically derive prediction intervals.

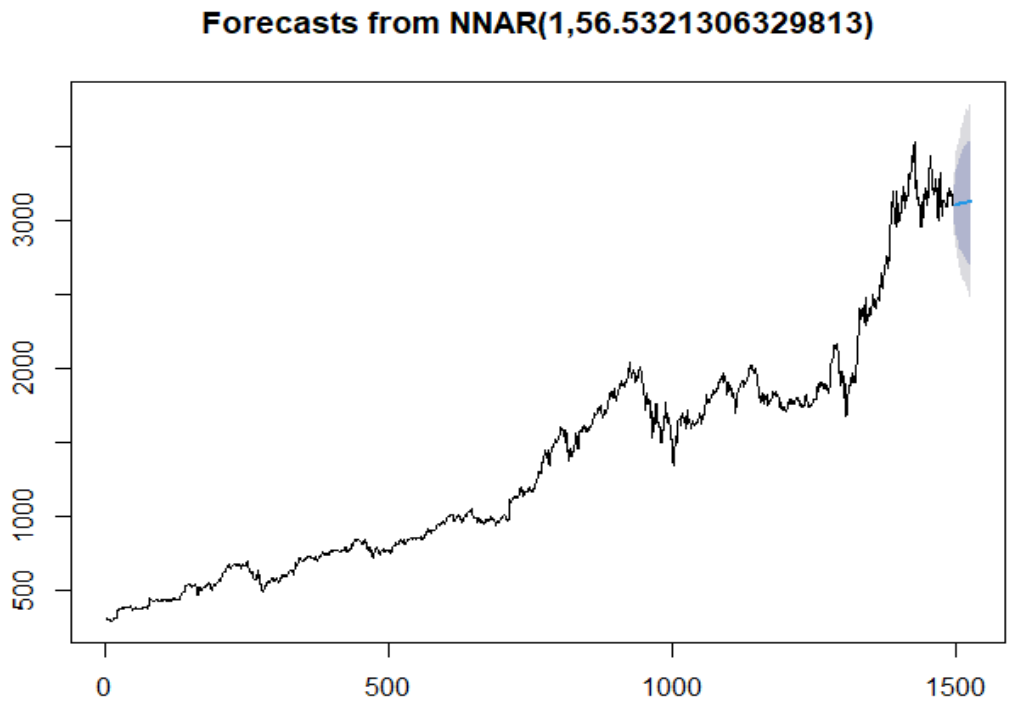

Figure 3.35: Neural Network Forecast

Following image shows the accuracy of the feed forward neural network implemented.

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
Training set	0.2384642	31.99291	18.70472	-0.01964071	1.303475	0.9814702	-0.05985411

Figure 3.36: Neural Network Accuracy

3.7 ETS Modeling

For this type of modeling, the ETS package automatically selects the ETS variant based on AIC, AICc and BIC. The following werre the models prescribed by the ETS package:

- On the basis of AIC: ETS(M,A,N)
- On the basis of BIC: ETS(M,N,N)
- On the basis of AIC and Damping : ETS(M,Ad,N)
- On the basis of BIC and Damping : ETS(M,Ad,N)

ETS(M,N,N)

Forecast for this model :

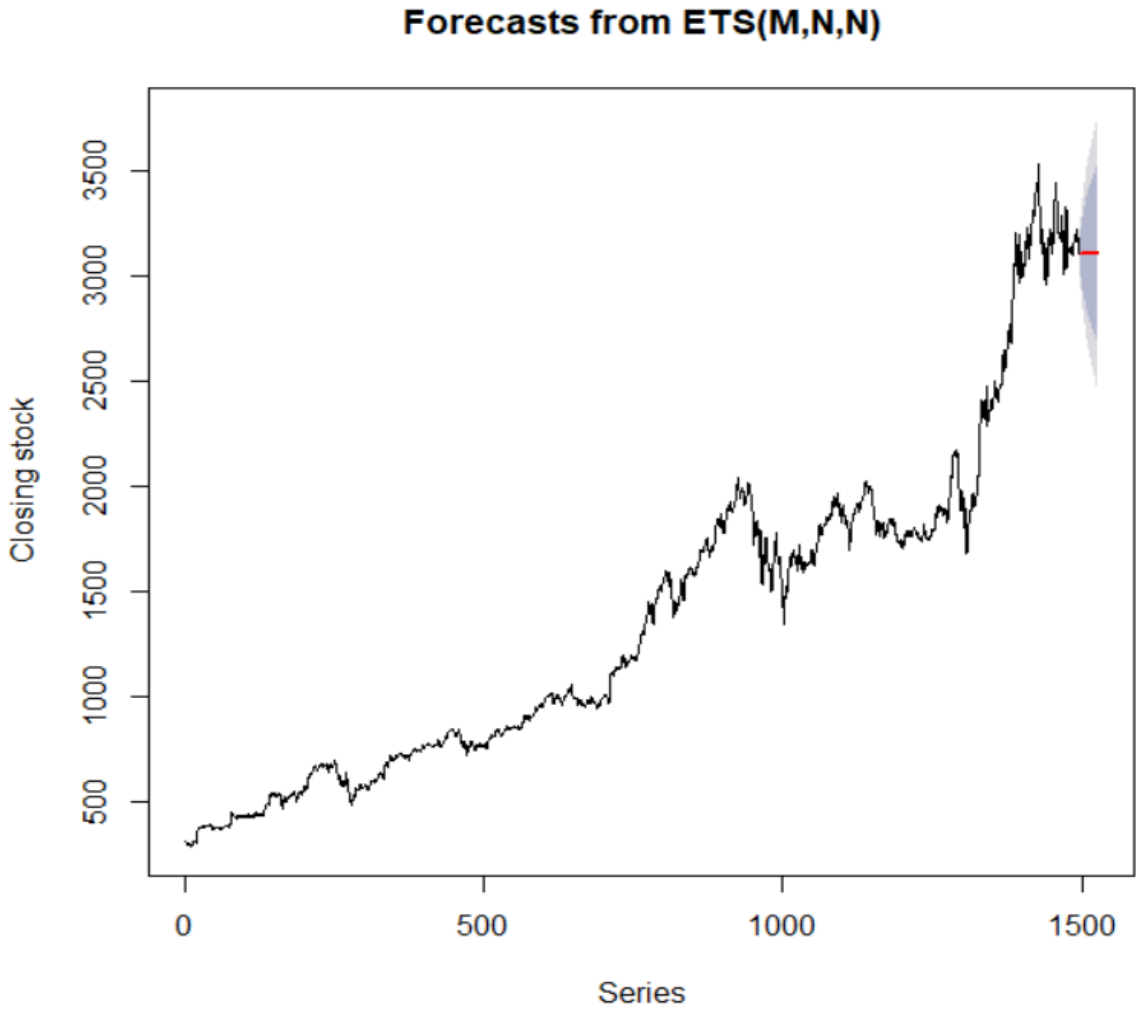

Figure 3.37: Forecast Plot for ETS(M,N,N)

The accuracy for this model:

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
Training set	1.909139	32.58867	19.05021	0.1383544	1.318285	0.9991381	-0.04575169

Figure 3.38: Accuracy of ETS(M,N,N) model

ETS(M,A,N)

Forecast for this model :

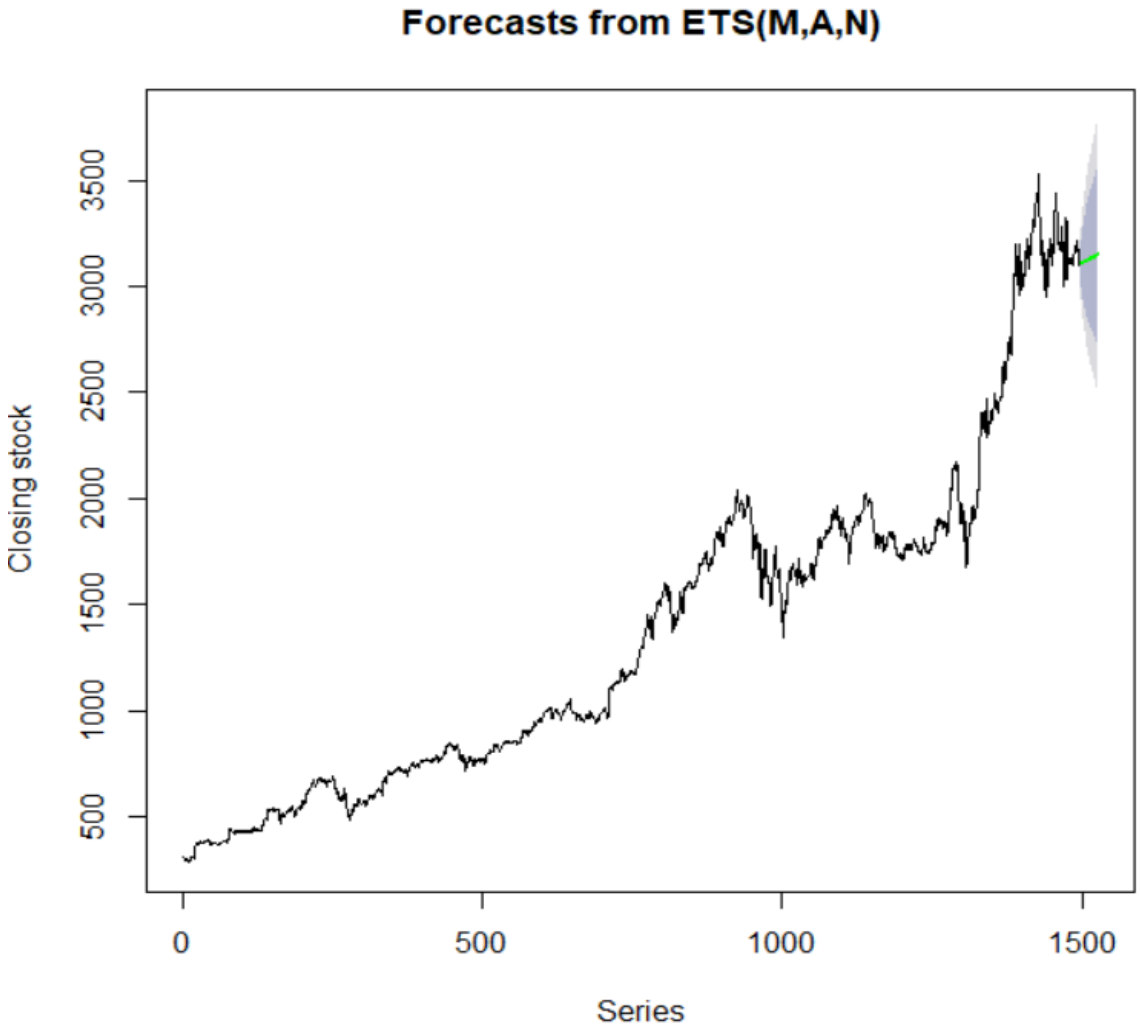

Figure 3.39: Forecast Plot for ETS(M,A,N) model

The accuracy for this model:

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
Training set	0.5463344	32.52234	18.97161	-0.004824417	1.316074	0.9950157	-0.03296453

Figure 3.40: Accuracy of ETS(M,A,N) Model

ETS(M,Ad,N)

Forecast for this model :

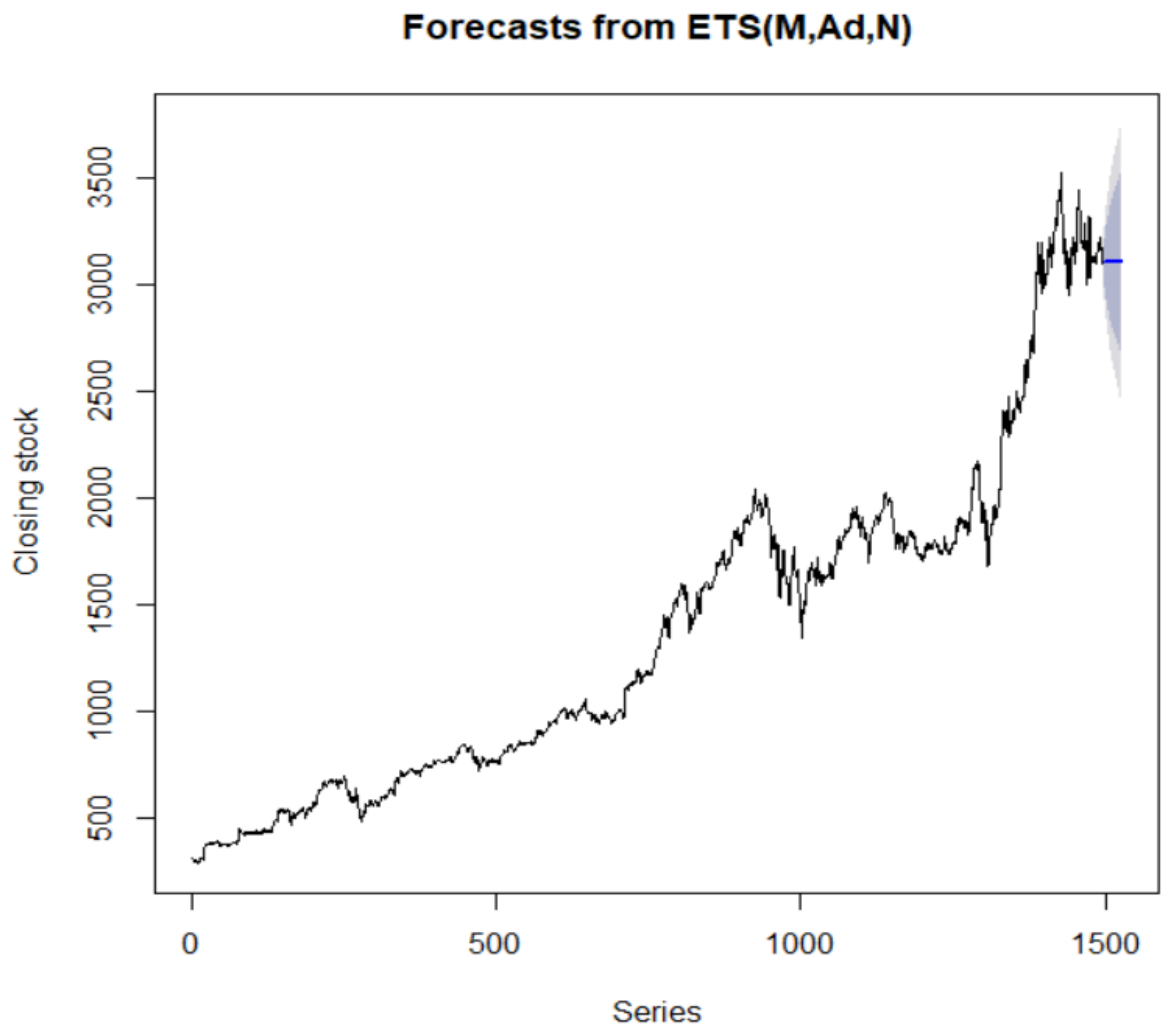

Figure 3.41: Forecast Plot for ETS(M,Ad,N) model

The accuracy for this model:

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
Training set	1.816439	32.58462	19.04411	0.1281414	1.318435	0.998818	-0.0446543

Figure 3.42: Accuracy of ETS(M,Ad,N) model

Based on the above 3 plots and MAPE's observed, we conclude that the model ETS(M,A,N) is giving us the best forecast with good accuracy.

Chapter 4

Results and Conclusions

Models	MAPE	Accuracy
ARIMA	1.3124	98.6875
Prophet	5.3878	94.6122
KNN Regession	2.4470	97.5530
Neural Network	1.3121	98.6879
ETS	1.31607	98.6839

Table 4.1: Models with their respective MAPE and Accuracy

In this study we focused in the application of different models, learning how to use them with the objective to forecast new price values. As we can see from our results, the models performed with similar future tendency predictions. All the models predicted a tendency of a higher price in 30 next days. We can conclude that the ARIMA and Neural Net models performed very well inside the prediction intervals and the accuracy metrics. The other models as they are new in this forecasting approach and the objective is to apply them in an intuitive form did not performed as well as ARIMA or Neural Net models. Maybe Prophet and Knn needs more tuning for getting more accurated results. Other very relevant point we have not mentioned is that Auto Regressive models, as they base on the past data to predict future values, tend to have an asymptotic prediction in long period future forecasts. Finally we conclude that ARIMA and Neural Nets are the best predicting models in this scenario, while incorporating GARCH to our ARIMA approach we had very interesting results. The other models used did not performed as well as ARIMA and Neural Nets under our metrics but this could be because they may need more tunning phases and training, testing approaches or they are not as effective as the other models because of their main application use in classificatory terms more than forecasting.

Chapter 5

Appendix

5.1 Code

The source code along with the data can be found here: Link (Click Here)

Bibliography

[1] Abdelmoula Dmouj. *Stock price modelling: Theory and Practice*, 2006 (accessed November 20, 2020).

[2] Kenneth Page. *Stock Price Forecasting Using Time Series Analysis, Machine Learning and single layer neural network Models*, 2019 (accessed November 24, 2020).

[3] Abhinav Tipirisetty. *Stock Price Prediction using Deep Learning*, 2018 (accessed November 18, 2020).

[4] Prathamesh Thakar. *Forecasting Time Series Data - Stock Price Analysis*, 2020 (accessed November December 6, 2020).

[5] Joy Gracia Harjanto. *Analyzing Stocks Using R*, 2018 (accessed November 25, 2020).