A Time-Series Analysis on the S&P 500 Stock Index

Abstract

Time series analysis and prediction have very broad applications in many areas from economics, finance to neutral science and engineering. The objective of this study is to examine the major steps and components of time series analysis and how to use Auto-Regressive Integrated Moving Average (ARIMA) model to implement those steps and predict S&P 500 stock index. Support Vector Machine (SVM) model is applied to the same dataset to provide a comparison with ARIMA. Well known daily S&P 500 stock index historical data between October 2013 and November 2018 from Yahoo Finance is used in this study.

Introduction

A time series is a set of observations or events recorded or ordered by fixed time interval. This simply means that particular values are recorded at a constant interval which may be hourly, daily, weekly, every 10 days, and so on. What makes time series different is that each data point in the series is dependent on the previous data points. For example, the daily S&P 500 stock market index is a time series ordered by day. And given a historical 3 months, 1 year or 5 years index data, prediction of the index moving trend is a time series analysis.

Time series analysis involves proper statistic model selections, parameter estimations and model goodness comparisons, and predict the future values of the time series. There are a number of techniques for time series analysis and prediction, Auto Regressive Integrated Moving Average (ARIMA) models are very popular class of forecasting model that utilize historical information to make predictions.

Through this project, ARIMA models are experimented through a process to determine the optimal p, d, q parameters, and a best fit model is chosen to forecast S&P 500 Stock Index daily closing values and trend based on past five years historical end of day index data. R is used in this project to do time series analysis.

Literature Review

Time series analysis theories and especially analysis of financial time series are reviewed and referenced during this project. The special natures of time series analysis, various techniques and models, and the applications of time series analysis in financial industry are well researched.

Articles on ARIMA model implementations and specific topics explaining ADF, ACF, PACF and AIC provide in-depth knowledge and understanding of the model and the approach to apply the model to actual financial time series data analysis. The augmented Dickey-Fuller (ADF) test is a formal statistical test for stationarity. Auto Correlation Function (ACF) and Partial Auto

Correlation Function (PACF) are used to identify ARIMA model parameters (p, d, q). And Akaike Information Criterion (AIC) is used to determine the best fit model.

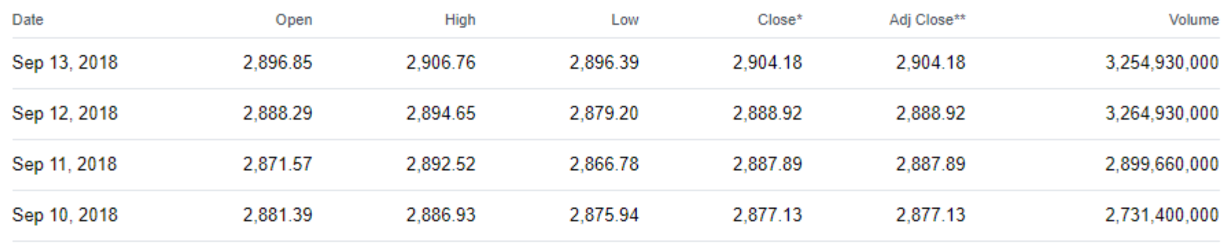
Dataset

The dataset used in this project is past 5 years S&P 500 index from September 2013 to September 2018. The data has been recorded on daily basis so the time interval is constant. The data is publicly available from Yahoo Finance.

Index dataset has seven attributes:

- recorded date (Date)
- index values at market open and close (Open, Close)
- highest and lowest index values during the trading day (High, Low)
- adjusted index value after market close by incorporating company's' dividend distributions and corporate actions (Adj Close)
- total number of shares of the index constituent stocks traded during the day (Volume)

Below is a sample of the index data:



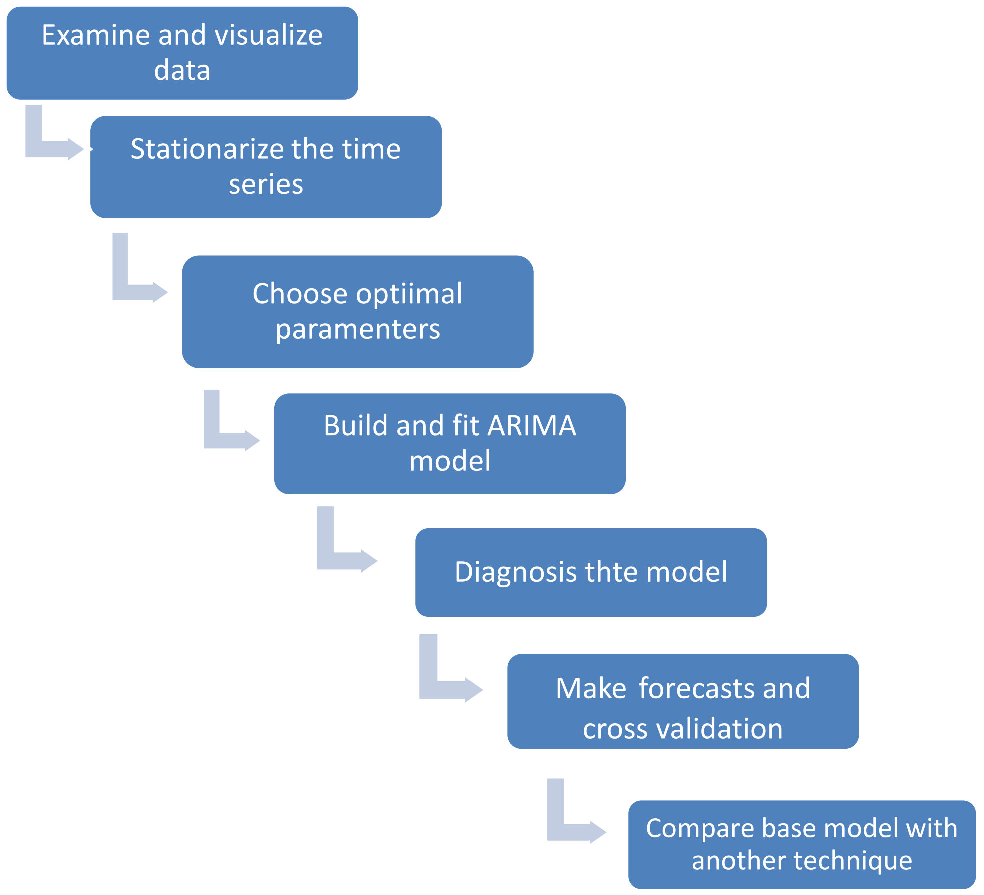
Date	Open	High	Low	Close*	Adj Close**	Volume
Sep 13, 2018	2,896.85	2,906.76	2,896.39	2,904.18	2,904.18	3,254,930,000
Sep 12, 2018	2,888.29	2,894.65	2,879.20	2,888.92	2,888.92	3,264,930,000
Sep 11, 2018	2,871.57	2,892.52	2,866.78	2,887.89	2,887.89	2,899,660,000
Sep 10, 2018	2,881.39	2,886.93	2,875.94	2,877.13	2,877.13	2,731,400,000

This project focuses on using univariate time series forecasting methods to predict daily index value at market close. Therefore, the attribute Close is used for analysis and forecast.

Approach

ARIMA stands for **Auto-Regressive Integrated Moving Average** and is specified by these three order parameters: *(p, d, q)*. The process of fitting an ARIMA model is sometimes referred to as the Box-Jenkins method. An **auto regressive (AR(p))** component is referring to the use of past values in the regression equation for the series. The auto-regressive parameter *p* specifies the number of lags used in the model. The *d* represents the degree of differencing in the **integrated (*I(d)*)** component. Differencing a series involves simply subtracting its current and previous values *d* times. Often, differencing is used to stabilize the series when the stationarity assumption is not met. A **moving average (MA(q))** component represents the error of the model as a combination of previous error terms. The order *q* determines the number of error terms to include in the model.

The following framework specifies the step by step approach that is taken to complete the time series forecasting of S&P 500 Index based on ARIMA .


Examine and visualize data
Stationarize the time series
Choose optiimal paramenters
Build and fit ARIMA model
Diagnosis thte model
Make forecasts and cross validation
Compare base model with another technique

Step 1: Examine and visualize data

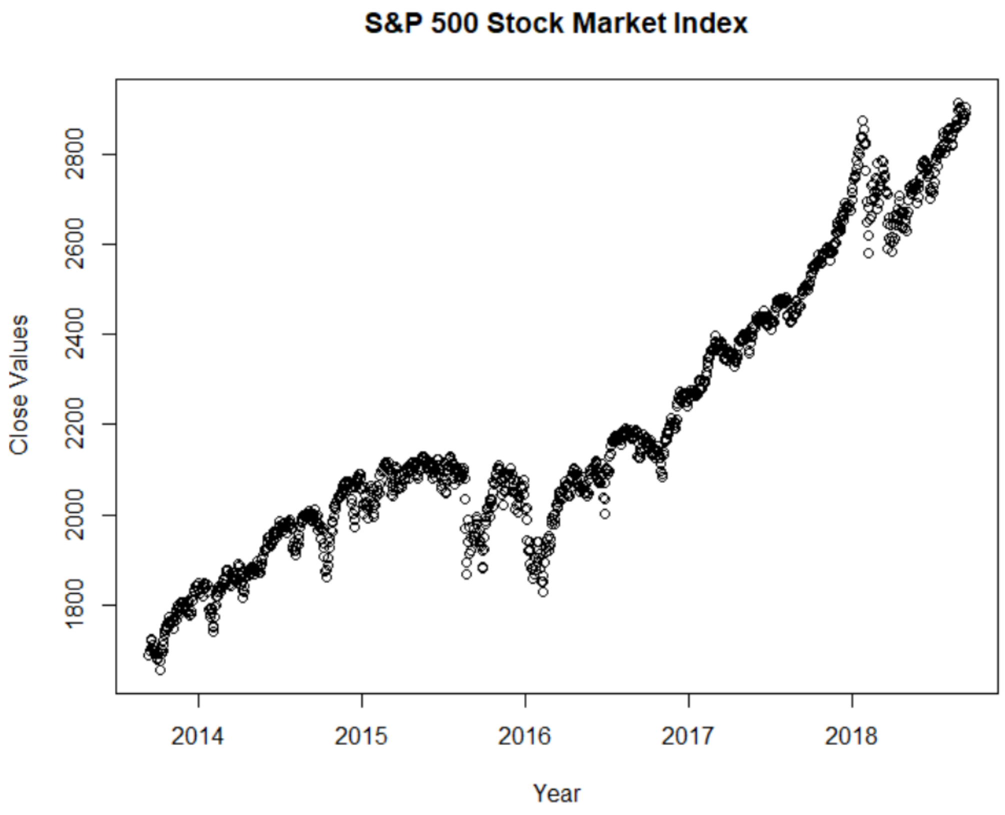
It is important to analyze and understand the characteristics of the data before building a time series model. We start off by loading the necessary R packages 'forecast' and 'tseries' and reading in the dataset to be analyzed - five years S&P 500 Stock Market Index. Plotting data is an easy perceptible approach to study the data so it's a widely used step in the exploratory data analysis phase. Important components of the time series dataset, such as trend, seasonality, and stationarity can be inferenced by data visualization via plotting.

Here is a quick summary of above mentioned components:

- **Trend:** A dataset is said to have a trend when it has either a long term increase or decrease.
- **Seasonality**: A dataset is said to have seasonality when it has patterns that repeat over known and fixed periods of time, for instance, daily, monthly, quarterly, or yearly.

- **Stationarity**: A stochastic process is called *stationary* if the mean and variance are constant which means their joint distribution does not change over time.

To ensure the model will not be biased by any outliers, daily index close values are cleaned first via time series functions tsclean(). Then close values are plotted to help examine the data. From the plot diagram below, we can see that daily index close values have an obvious uptrend. The plot also shows that index close values do not have constant means and variance so it's not stationary. It does not have obvious repeating patterns over certain period of time so index close values do not have seasonality. In order to use ARIMA model to do the time series analysis and prediction, the data must be stationary [1][2]. In next step, stationarity will be further analyzed in detail.



Step 2: Stationarize the time series

ARIMA uses previous lags in the series to model its future behavior. A stable time series with consistent behaviors results less modeling uncertainty. Therefore, a stationary series is required to fit an ARIMA model. A series is said to be stationary when its mean, variance, and autocovariance do not vary between different time period. The ADF (Augmented Dickey-Fuller) test is a formal statistical test for stationarity. To confirm the stationarity observation in step 1, ADF test is performed. The null hypothesis assumes that the series is non-stationary.

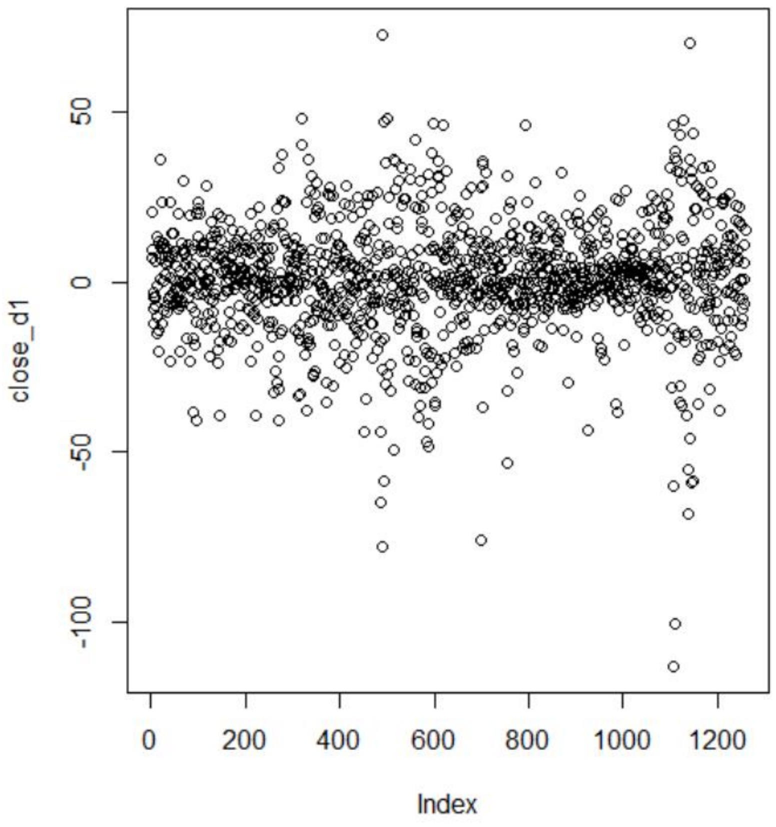
ADF test below shows that p-value > 0.5, so null hypothesis is accepted, i.e. the series is not stationary.

```
adf.test(ts_close, alternative = "stationary")

        Augmented Dickey-Fuller Test

data:   ts_close
Dickey-Fuller = -1.6706, Lag order = 10, p-value = 0.7178
alternative hypothesis: stationary
```

Differencing, a simple data transformation approach, is usually used to correct a non-stationary series. The idea behind this approach is that the change from one period to another might be constant although the original data series are not. Differencing is performed by subtracting one period's or interval's values from the previous period's or interval's values. In addition to correcting non-stationarity of the series, differencing can help removing its trend too.

The number of differences performed is represented by the *d* parameter of ARIMA model. The order of *d = 1* is started with and then whether further differencing is required is re-evaluated. If the differenced series still fails ADF stationary test, the differencing process will be repeated with d=2, 3, … until stationarity is achieved.

```
close_d1 <- diff(ts_close, differences = 1)
plot(close_d1)
```

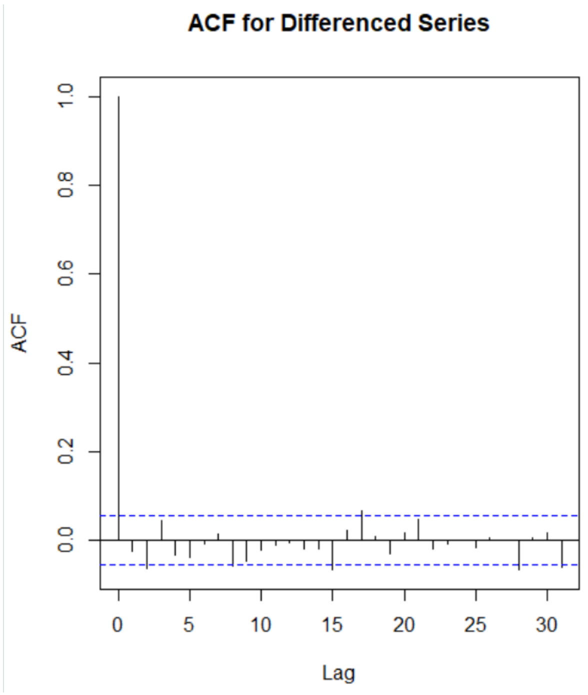


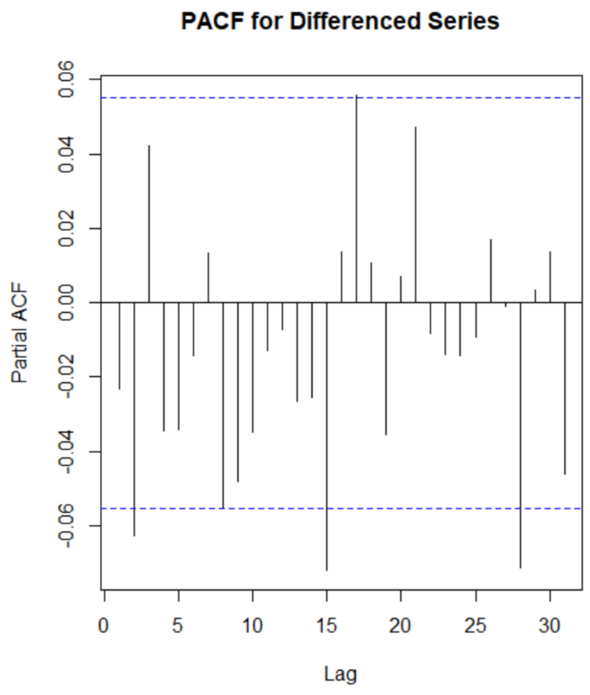
First differenced series is plotted above and it shows an oscillating pattern around 0 with no visible strong trend. This suggests that the series with one differencing (d=1) transformation is probably already stationary and should be included in the model. ADF test on the differenced index closing values can be further used to confirm this observation. p-value is less than 0.5 so we reject null hypothesis and accept alternative hypothesis that the series is already stationary.

```
adf.test(close_d1, alternative = "stationary")

        Augmented Dickey-Fuller Test

data:  close_d1
Dickey-Fuller = -12.313, Lag order = 10, p-value = 0.01
alternative hypothesis: stationary
```

Step 3: Plot ACF/PACF charts and choose optimal parameters

ACF (Auto Correlation Function or Autocorrelation plots) displays correlation between a series and its lags. ACF plots can help in determining the order of the *MA (q)* model. PACF(Partial





Autocorrelation Plots) displays correlation between a variable and its lags that is not explained by previous lags. PACF plots are useful when determining the order of the *AR(p)* model.

The blue line in ACF and PACF plots shows significant different values than zero. The ACF graph above does not have significant cut off or spikes, which suggests this is mostly an MA(0) process. The PACF graph shows geometric decay, which means this is mostly an AR(0) process. Because the differenced time series with d=1 is examined, the model is assumed to be

ARIMA(0,1,0). Next step will test this model and a couple of other models to find a best fit model.

Step 4: Build and fit ARIMA model

The parameters (0, 1, 0) found in the previous step might be an approximate estimate. Based on this assumption, more (p,d,q) combinations are explored and compared in order to identify the optimal terms. A number of approaches for comparing quality of fit across multiple models exist. Akaike Information Criteria (AIC) and Baysian Information Criteria (BIC) are the two of the most widely used. These two criteria are closely related and can be interpreted as an estimate of how much information would be lost if a given model is chosen. The choice would be the one with minimal AIC and BIC values.

```
arima(close_d1, order=c(0,0,0))  #ARIMA(0,1,0) aic = 10685.41
arima(close_d1, order=c(1,0,0))  #ARIMA(1,1,0) aic = 10686.73
arima(close_d1, order=c(1,0,1))  #ARIMA(1,1,1) aic = 10679.06, lowest
arima(close_d1, order=c(0,0,1))  #ARIMA(0,1,1) aic = 10686.63
arima(close_d1, order=c(0,0,2))  #ARIMA(0,1,2) aic = 10683.67
arima(close_d1, order=c(2,0,0))  #ARIMA(2,1,0) aic = 10683.75
```

The experiments of ARIMA model with various (p, d, q) combinations show that ARIMA(1, 1, 1) (since already differenced series is used in arima(), d is set to 0 but the model with actual series should set d = 1) has the least AIC. This suggests ARIMA(1,1,1) is the best fit model. We can leverage the *auto.arima()* function provided in R *forecas*t package to verify the experiment conclusion.

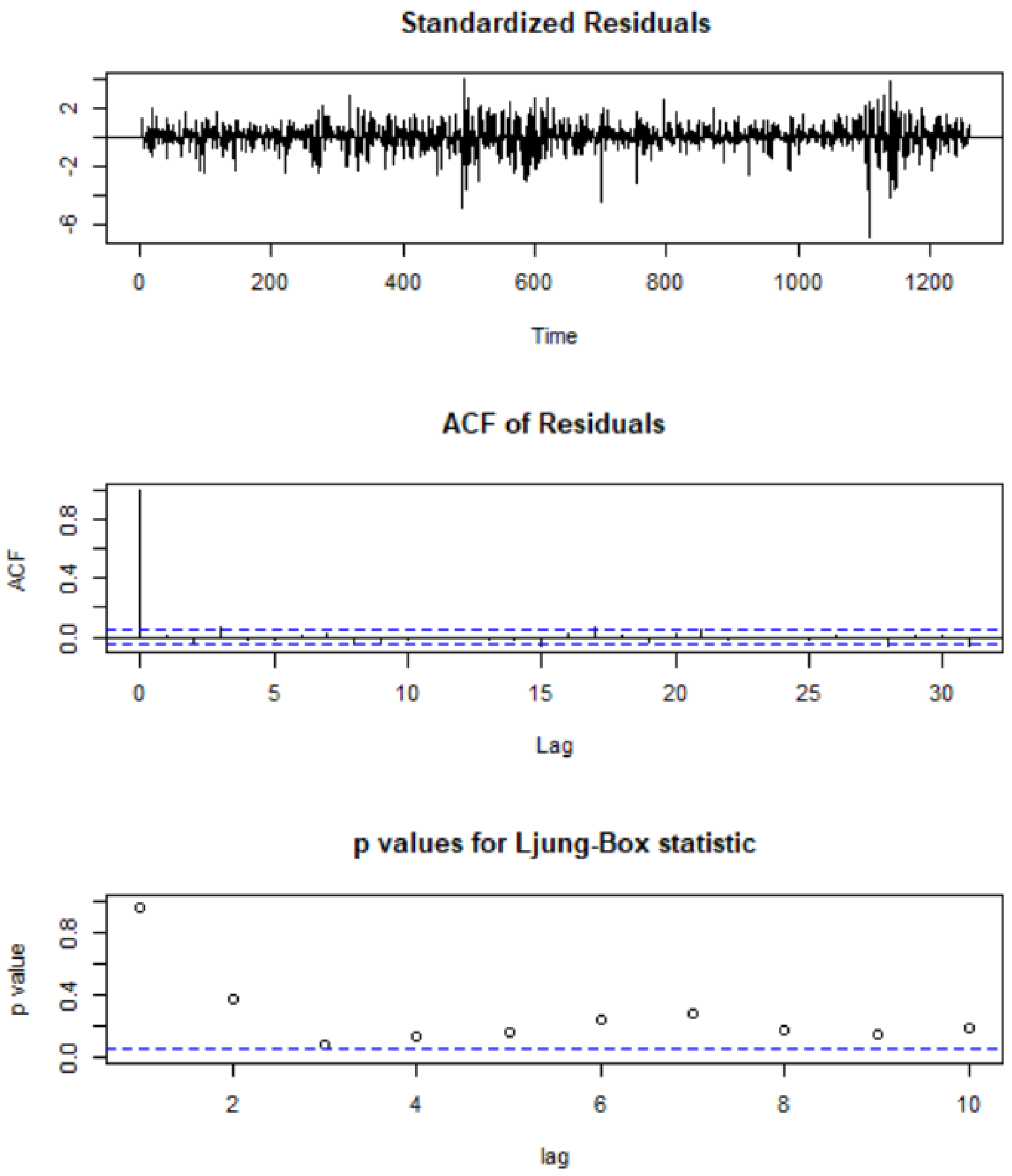
```
auto.arima(close_d1, seasonal = FALSE)
Series: close_d1
ARIMA(1,0,1) with non-zero mean

Coefficients:
         ar1      ma1    mean
      0.9369  -0.9688  0.9378
s.e.  0.0263   0.0186  0.2376

sigma^2 estimated as 281.5:  log likelihood=-5335.53
AIC=10679.06   AICc=10679.1   BIC=10699.62
```

We used differenced series with d = 1, so auto.arima() confirms that ARIMA(1, 1, 1) is the best fit model. The model uses an autoregressive term of first lag (p = 1) and a moving average model of order 1 (q=1), and incorporates differencing of degree 1 (d=1).

Step 5: Diagnosis the model

So now a fitted model ARIMA(1,1,1) is identified that can be used to make predictions, but we need examine whether the model is in line with observations and analysis done from previous steps. We can use tsdiag() diagnosis function to examine the residuals of the model. If the model is not specified correctly, usually this is reflected in residuals in the form of trends, skewness, or any other patterns not captured by the model, and there will be no significant autocorrelations present among residual data points. If the model is correctly specified, residuals should look like white noise, meaning they are normally distributed.

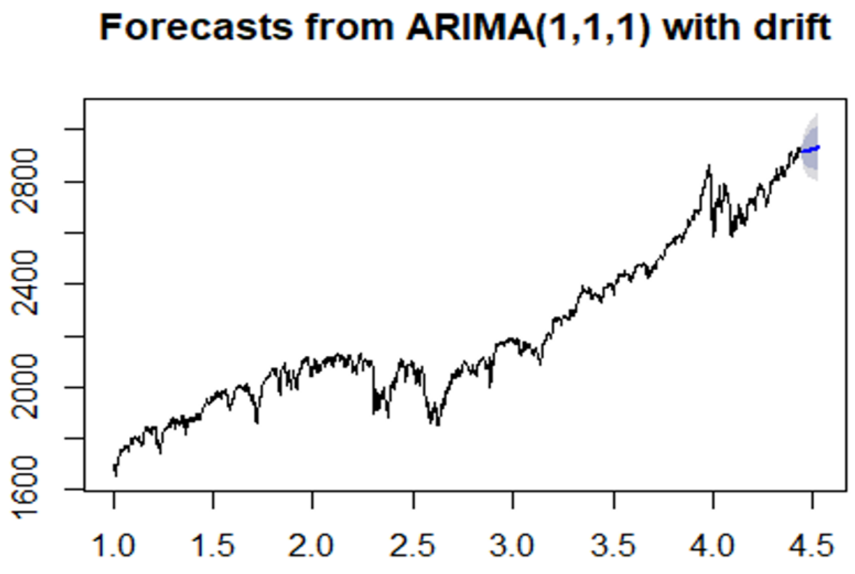
As we can see from the diagnosis output, the residual data points seem no pattern and all moving around zeros which is similar to white noise process. We can also see that none of the ACF with respective of lags are significant. This means that everything is under confidence interval, the blue lines. There is no correlation between residuals hence the model seems statistically fits the data. If the diagnosis shows that the model does not fit the data well, we have to iterate step 4 & 5 by tweaking order parameters.



Step 6: Make forecasts and cross validation

Now it's ready to make forecasts on the future index closing values and trends using the final ARIMA(1,1,1) model. The S&P 500 index close values of thirty business days from October 1st to November 9th are forecasted based on the past five years index historical data.

The blue line in the shadow below shows the fit provided by the model.

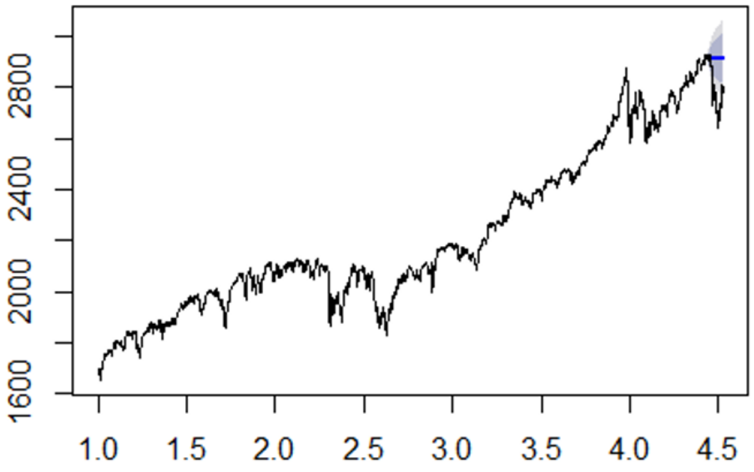


Results

The model errors are calculated using *accuracy()* function and the results are shown as below. MAPE (The mean absolute percentage error) is quite low which indicates the forecast performance is not bad.

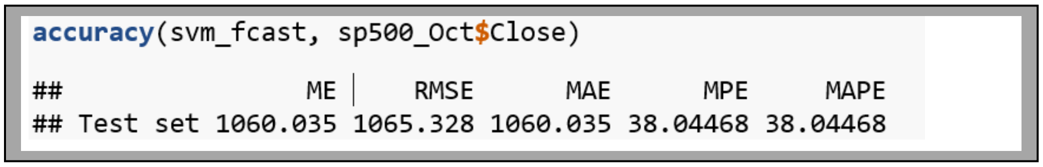
```
accuracy(fcast, sp500_Oct$Close)

##                        ME          RMSE        MAE          MPE          MAPE
## Training set    0.02250337     16.58955    11.55024   -0.006219626  0.5332682
## Test set     -139.01892711    162.13242   141.11746   -5.089648731  5.1614038
##                      MASE          ACF1
## Training set    0.9897036      -0.01070866
## Test set       12.0919119          NA
```

However, when using the following plot to compare October forecasted index close values and the actual values, forecast shows a different trend from what actually happened. This suggests that other patterns like seasonality may exist in the data or unexpected events had significant impacts on stock market which caused changes of the trend.



Compare Base ARIMA with SVM Model

To further evaluate how ARIMA model performs, SVM model is applied to the same five years index dataset. Errors of the SVM forecast is shown below. It has a much higher MAPE than ARIMA model. This indicates that ARIMA model forecast performs better on this time series dataset than SVM.



```
accuracy(svm_fcast, sp500_Oct$Close)

##                 ME     RMSE      MAE      MPE     MAPE
## Test set 1060.035 1065.328 1060.035 38.04468 38.04468
```

Conclusions

This study thoroughly researched and experimented time series analysis techniques especially well-known Auto-Regressive Integrated Moving Average (ARIMA) model. ARIMA model has strict requirements on the time series dataset, for example, stationary and seasonality. Data must be well prepared and preprocessed before applying the model. Since ARIMA has been studied over decades, many tools like differencing the data to make it stationary are developed to simplify data preprocessing as well as model parameter identification process. There exist clear time series analysis and forecast steps to follow when using ARIMA. This report explained how to follow and implement these steps to analyze five years S&P 500 stock market index close values and predict future trend. This study and project implementation will enable me to apply time series analysis techniques especially ARIMA to the actual data science projects at work place.

GitHub:

This project GitHub repository can be found here. The complete R script, R markdown and the dataset used in this project are shared through this public repository.

References:

[1] Robert H. Shumway, David S. Stoffer. Time Series Analysis and Its Applications With R Examples, Fouth Edition, Springer

[2] Ruey S. Tsay. Analysis of Financial Time Series. Financial Economics, University of Chicago, 2002 by John Wiley & Sons Inc.

[3] Tavish Srivastava. A Complete Tutorial on Time Series Modeling in R. December 16, 2015, https://www.analyticsvidhya.com/blog/2015/12/complete-tutorial-time-series-modeling/

[4] Mohamad As'ad. Finding the Best ARIMA Model to Forecast Daily Peak Electricity Demand. https://ro.uow.edu.au/cgi/viewcontent.cgi?referer=https://www.google.ca/&httpsredir=1&article=1011&context=asearc

[5] Jason Brownlee. A Gentle Introduction to Autocorrelation and Partial Autocorrelation. https://machinelearningmastery.com/gentle-introduction-autocorrelation-partial-autocorrelation/

[6] AISHWARYA SINGH. Build High Performance Time Series Models using Auto ARIMA in Python and R. https://www.analyticsvidhya.com/blog/2018/08/auto-arima-time-series-modeling-python-r/

[7] Raul Eulogio. Performing a Time-Series Analysis on the S&P 500 Stock Index. https://www.datascience.com/blog/stock-price-time-series-arima

[8] R. Samsudin, A. Shabri and P. Saad, 2010. A Comparison of Time Series Forecasting using Support Vector Machine and Artificial Neural Network Model. Journal of Applied Sciences, 10: 950-958. https://scialert.net/fulltextmobile/?doi=jas.2010.950.958