Group Project Report

Group D8

3/16/2020

Introduction:

In this project, we seek to inspect a well-known time series, the S&P 500 index, in order to identify trends and attempt to make a future forecast.

Abstract

Using one year of historical closing values from the S&P 500 index, we attempt to fit a variety of models, and analyze the trends. From the portmanteu test on the original data, we could not say that the data is not generated from a random noise process at the 5% level. To remove an upward trend, the data was differenced, but this yielded the same portmanteu test result. Using model selection, we identified the top ARIMA models to model this sequence using leave-one-out CV and RMSE and AIC as metrics. The best model by this criteria is ARIMA(1, 1, 0). Using this model, we attempt to predict the next 5 closing price values and demonstrate the comparison to the actual values.

Data

The S&P 500 Index is a collection of some of the largest US stocks from a variety of industries, weighted by their market capitalization. A closing price is the last price at which the stock was trading during the previous trading day. In the case of S&P 500, this is the last weighted calculation of the index for the trading day, calculated at 4PM ET/EDT. In this report, we use the S&P 500 index closing price for January 1, 2019 to Dec 31st 2019 for model training, and January 2 to January 8 2020 for model prediction.

The data source is Yahoo Finance, which we used because it is a free, trusted source for accurate stock data. We used the quantmod package (Quantitative Financial Modelling & Trading Framework for R) which has an API for importing the data from Yahoo.

The S&P 500 is superior to individual stocks for modeling, as it is an aggregate – this allows us to obtain a "big picture" in comparison to individual stocks, which are more volatile and subject to more extreme fluctuation. Our motivation for using this data was to obtain a prediction model based on historical price data. Such a model would not only be useful for profiting from market trades, but also as an indicator for US stock movement, and therefore GDP growth and economic sentiment. Of course, a simple model would be unlikely, so we used methods such as Neural Networks to model the data.

Analysis:

```r
library(doSNOW)
library(forecast)
library(ggfortify)
library(parallel)
library(quantmod)
library(tcltk)
options('getSymbols.warning4.0' = F)
```

We aim to predict the daily closing values of The S&P 500 Index from 2020–01–02 to 2020–01–08. In this case, we will use one year of historical closing values for S&P 500 from 2019–01–01 to 2020–01–01, which can be accessed through yahoo finance. After downloading, the dataset looks like this:

Using data from Yahoo! Finance

```r
getSymbols(Symbols = '^GSPC',
           src     = 'yahoo',
           auto.assign = T,
           from    = '2019-01-01',
           to      = '2020-01-01')
```

```
## Warning: 'indexClass<-' is deprecated.
## Use 'tclass<-' instead.
## See help("Deprecated") and help("xts-deprecated").
```

```
## [1] "^GSPC"
```

```r
data <- GSPC[, 'GSPC.Close']
head(data)
```

```
##            GSPC.Close
## 2019-01-02    2510.03
## 2019-01-03    2447.89
## 2019-01-04    2531.94
## 2019-01-07    2549.69
## 2019-01-08    2574.41
## 2019-01-09    2584.96
```

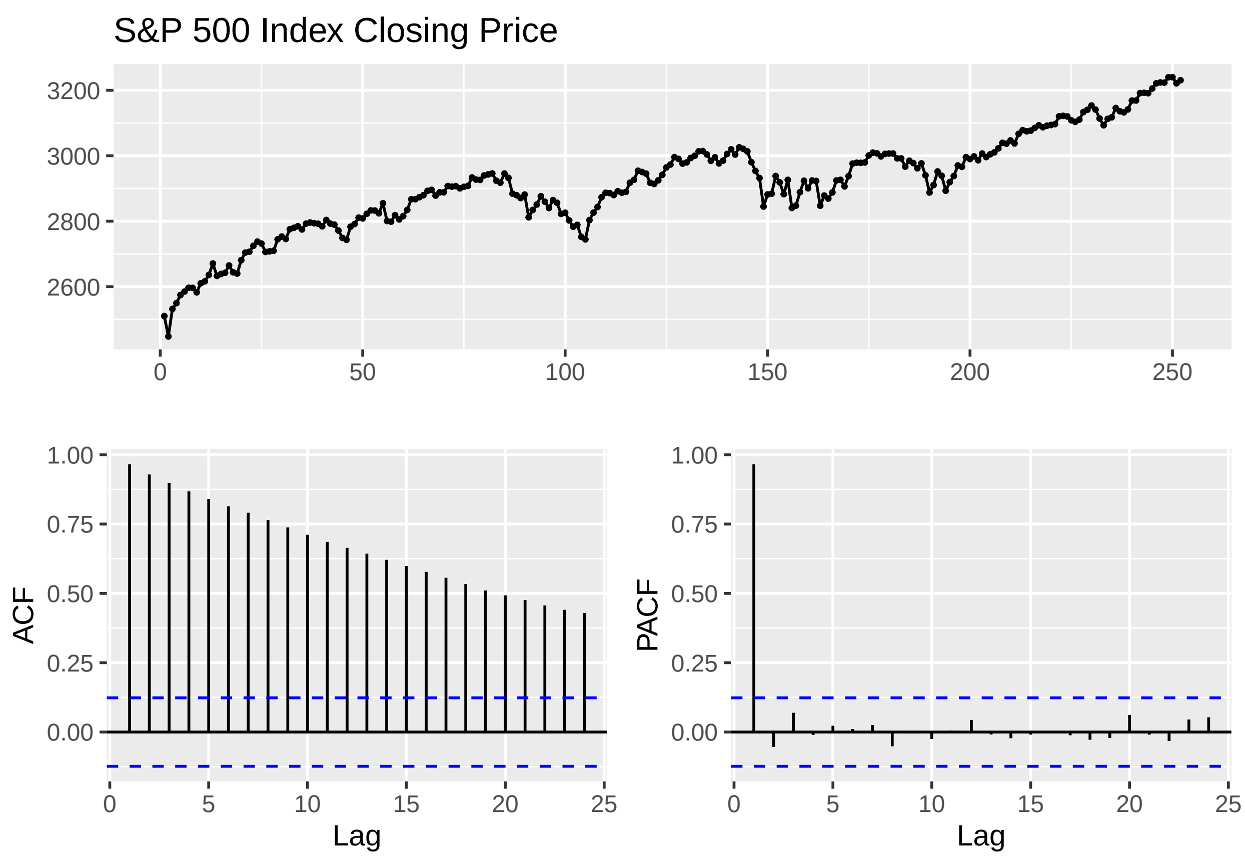
```r
tail(data)
```

```
##            GSPC.Close
## 2019-12-23    3224.01
## 2019-12-24    3223.38
## 2019-12-26    3239.91
## 2019-12-27    3240.02
## 2019-12-30    3221.29
## 2019-12-31    3230.78
```

```r
sp500 <- ts(data)
```

Time Series Plot

```r
ggtsdisplay(sp500, main = 'S&P 500 Index Closing Price')
```



The time series plot shows an increasing linear trend with no obvious seasonality.

ACF plot shows a very slow decay in time suggesting that the series might not be stationary.

PACF cuts off at lag 1

Portmanteau Test

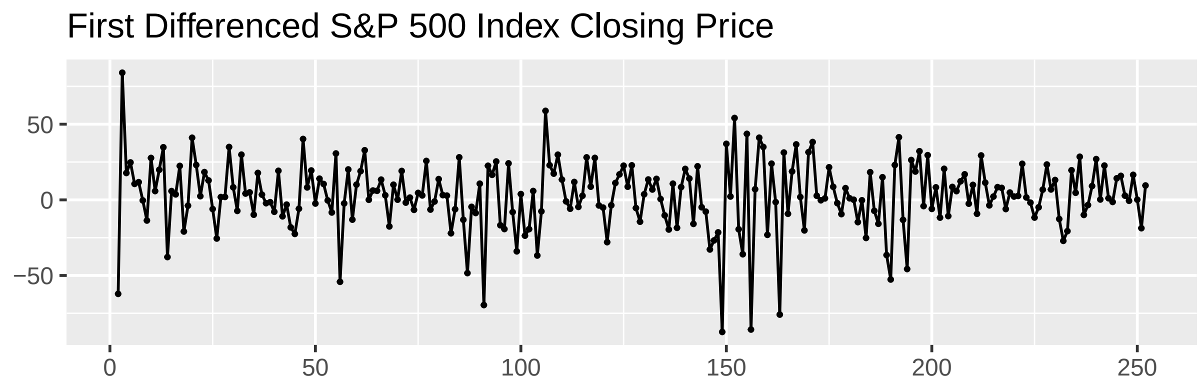
```r
Box.test(sp500, lag = 25, type = 'Box-Pierce')
```

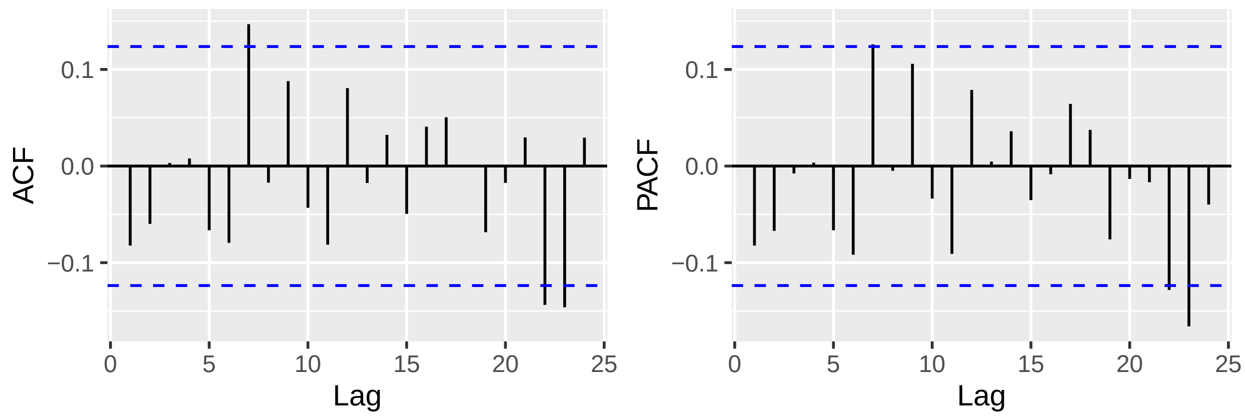
```
##
##  Box-Pierce test
##
## data:  sp500
## X-squared = 2891.3, df = 25, p-value < 2.2e-16
```

The p-value of the portmanteau test for residuals is less than 0.05. Therefore, we have sufficient evidence to reject the null hypothesis that the series is white noise. Then, to make the series stationary, we compute the differences between consecutive observations.

First Differenced Time Series Plot

```r
sp500 %>%
  diff() %>%
  ggtsdisplay(main = 'First Differenced S&P 500 Index Closing Price')
```





The differenced series plot shows variation without apparend trend.

Although both ACF and pacf plots show significant values at lag 7, 22,and 23, but they drop to zero quickly and most of values are not significant. Based on that, we could probably conclude that the series is a white noise process.

Portmanteau Test

```r
sp500 %>%
  diff() %>%
  Box.test(lag = 25, type = 'Box-Pierce')
```

```
##
##  Box-Pierce test
##
## data:  .
## X-squared = 30.757, df = 25, p-value = 0.1972
```

The p-value of the portmanteau test for residuals is greater than 0.05. Therefore, we do not have sufficient evidence to reject the null hypothesis at 5% significant level, so we can conclude that the differenced series is consistent with a white noise process. However, we need to further prove whether the white noise model can perform well in the prediction.

In the next part, we will use Leave-one-out cross validation to measure the goodness of prediction for all possible models(ARMA(p,1,q) where p+q< 9). For every i = 1,. . . ,251: a) train the model on every point except i, compute the test error on the held out point. b) Average the test errors

To evaluate the effectiveness of our methods, we will use the root mean square error (RMSE) and Akaike information criterion (AIC) metrics. For both metrics, the lower the value, the better the prediction.

Time Series Cross Validation for Model Selection

```r
cluster <- makeSOCKcluster(detectCores(logical = T) - 1)
registerDoSNOW(cluster)

nfolds <- length(sp500) - 1
# Leave-one-out
# kfolds <- 21:nfolds
# 12-Fold (~1 month rolling window)
kfolds <- round(seq(5, nfolds, length.out = 52))

# For Progress Bar window:
pb <- tkProgressBar(max = length(kfolds))
opts <- list(progress = function(n) setTkProgressBar(pb, n))
# For console output:
# opts <- list(progress = function(n) cat(sprintf('Fold %d is complete\n', n)))

fit_arima <- function(x, p, q) {
  model <- tryCatch({
      return(Arima(x, order = c(p, 1, q), include.constant = T))
  }, error = function(e) {
    tryCatch({
      return(Arima(x, order = c(p, 1, q), include.constant = T, method = 'ML'))
    }, error = function(e) {
      return(Arima(x, order = c(p, 1, q), include.constant = T, method = 'ML', transform.pars = F))
    })
  })
  return(model)
}

score <- foreach(k = kfolds, .options.snow = opts, .packages = c('forecast')) %dopar% {
  # initialize data.frame for each thread
  spe <- data.frame(matrix(0, 5, 5), row.names = c('0', '1', '2', '3', '4'))
  colnames(spe) <- rownames(spe)
  aic <- data.frame(matrix(0, 5, 5), row.names = c('0', '1', '2', '3', '4'))
  colnames(aic) <- rownames(aic)
  # Split sp500 data into train and validation set
  train <- sp500[1:k]
  validation <- sp500[k + 1]
  for (p in 0:4) {
    for (q in 0:4) {
      if (p == 0 && q == 0) next # Skip ARIMA(0, 1, 0)
      model <- fit_arima(x = train, p, q)
      aic[as.character(p), as.character(q)] <- model$aic
      y_hat <- forecast(model, h = 1)$mean[1]
      spe[as.character(p), as.character(q)] <- (y_hat - validation)^2
    }
  }
  return(list(spe, aic))
}
close(pb)

rmspe_list <- sapply(score, function(x) x[1])
rmspe <- sqrt(Reduce('+', rmspe_list) / length(score))

aic_list <- sapply(score, function(x) x[2])
aic <- Reduce('+', aic_list) / length(score)

result <- data.frame(matrix(ncol = 4, nrow = 0))
colnames(result) <- c('p', 'q', 'RMSPE', 'AIC')
for (i in 1:5) {
  for (j in 1:5) {
    if (i == 1 && j == 1) next
    result[nrow(result) + 1,] <- c(i - 1, j - 1, rmspe[i, j], aic[i, j])
  }
}
```

Best models based on RMSPE

```
print(result[order(result$RMSPE)[1:5], ])
```

```
##   p q    RMSPE      AIC
## 5 1 0 22.62201 1159.596
## 1 0 1 23.22832 1159.304
## 2 0 2 23.37023 1160.796
## 6 1 1 23.37041 1160.604
## 3 0 3 23.43676 1162.449
```

Best models based on AIC

```
print(result[order(result$AIC)[1:5], ])
```

```
##    p q    RMSPE      AIC
## 1  0 1 23.22832 1159.304
## 5  1 0 22.62201 1159.596
## 6  1 1 23.37041 1160.604
## 2  0 2 23.37023 1160.796
## 10 2 0 23.80278 1161.078
```

Best model: ARIMA(1, 1, 0), because it has the lowest RMSPE, and the second lowest AIC.

Fit best model on the full train data

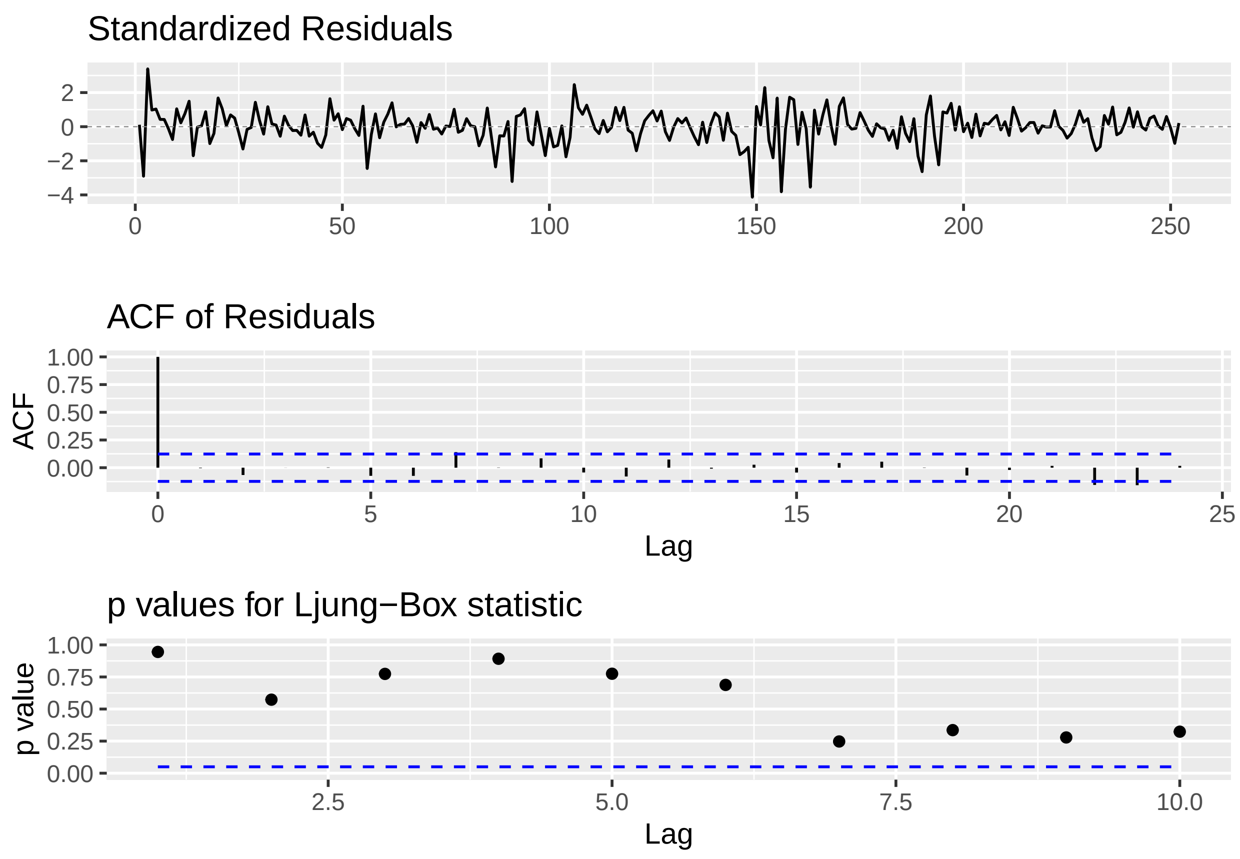
```
best_model <- Arima(sp500, order = c(1, 1, 0), include.constant = T)
summary(best_model)
```

```
## Series: sp500
## ARIMA(1,1,0) with drift
##
```

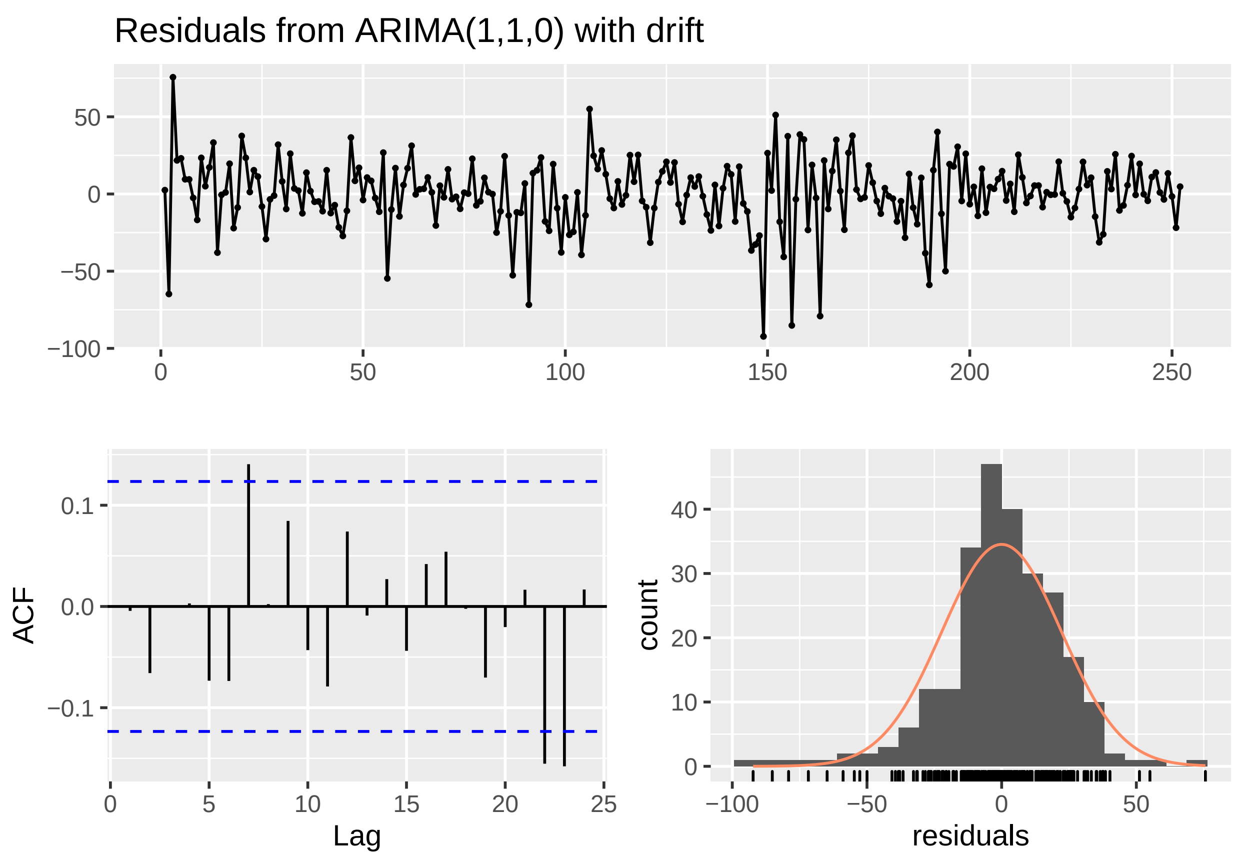
```
## Coefficients:
##           ar1   drift
##       -0.0849  2.8897
## s.e.   0.0639  1.2936
##
## sigma^2 estimated as 498:  log likelihood=-1134.57
## AIC=2275.13   AICc=2275.23   BIC=2285.71
##
## Training set error measures:
##                      ME     RMSE      MAE          MPE      MAPE      MASE
## Training set -0.0110232 22.1817 15.96988 -0.001719674 0.5560667 0.9770227
##                      ACF1
## Training set -0.004317636
```

Residual Analysis

```
ggtsdiag(best_model)
```



```
checkresiduals(best_model, lag = 25)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,0) with drift
## Q* = 31.884, df = 23, p-value = 0.1026
##
## Model df: 2.   Total lags used: 25
```

We can see that there is no pattern apparent in the residuals analysis plot. The acf values are not significant for lags other than 0. THe p-values for Ljung-Box test are also large suggesting nothing untoward about the fit of the model.Hence, we will use the ARIMA(1,1,0) to forcast the next 5 closing price of S&P 500, which are the closing values from 2020–01–02 to 2020–01–08.

Forecasting

Retrieve the next 5 closing prices

```
getSymbols(Symbols = '^GSPC',
           src     = 'yahoo',
           auto.assign = T,
           from    = '2020-01-01',
           to      = '2020-03-31')
```

```
## Warning: 'indexClass<-' is deprecated.
## Use 'tclass<-' instead.
## See help("Deprecated") and help("xts-deprecated").
## [1] "^GSPC"
```

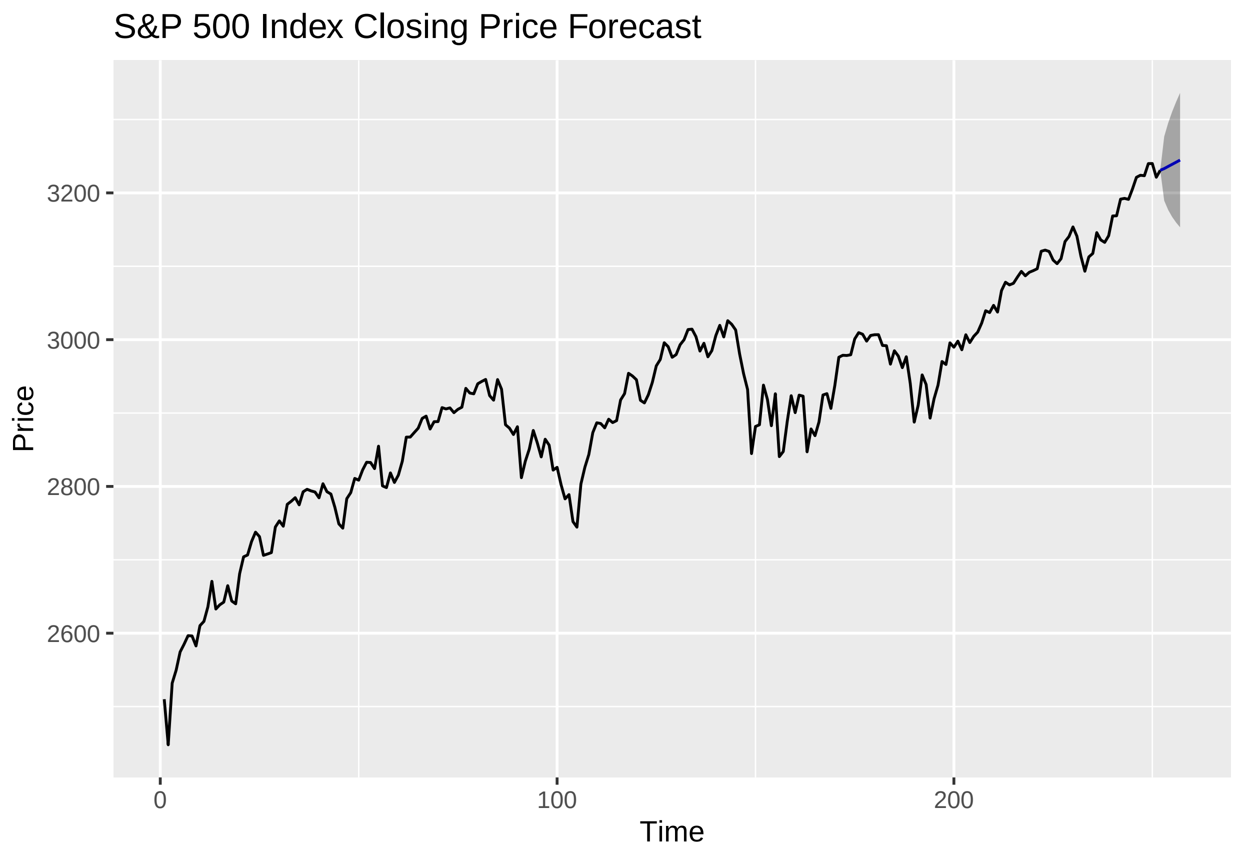
```
data <- GSPC[1:5, 'GSPC.Close']
head(data)
```

```
##            GSPC.Close
## 2020-01-02    3257.85
## 2020-01-03    3234.85
## 2020-01-06    3246.28
## 2020-01-07    3237.18
## 2020-01-08    3253.05
```

```
test <- as.vector(data)
```

Forecast

```
forecast <- forecast(best_model, h = 5, level = 95)
autoplot(forecast) +
  ggtitle(label = 'S&P 500 Index Closing Price Forecast') +
  ylab(label = 'Price') +
  xlab(label = 'Time')
```



Evaluate MSE

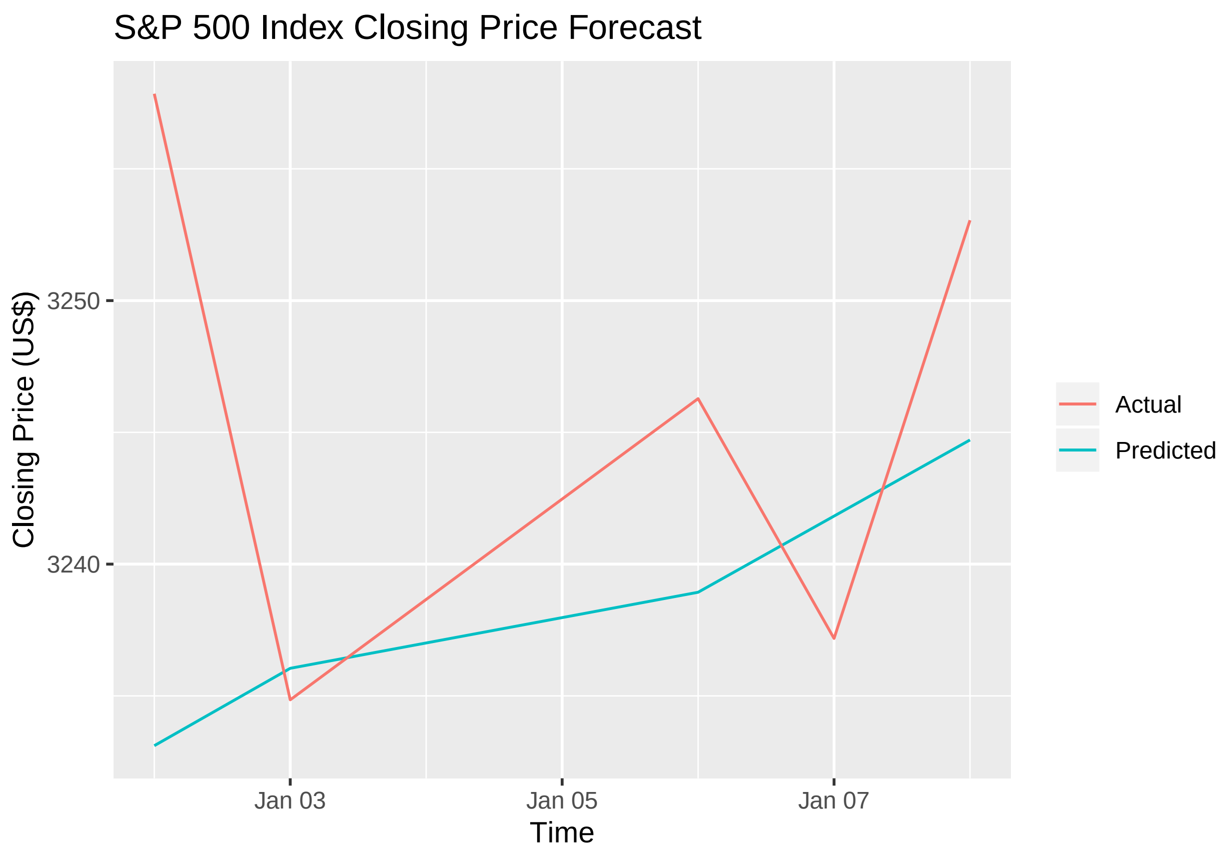
```r
pred <- as.vector(forecast$mean)
accuracy(pred, test)
```

```
##                ME     RMSE      MAE       MPE      MAPE
## Test set 6.917322 12.31737 9.253049 0.2123299 0.2844937
```

The Root Mean Squared Error is 12.31737, and we also obseved all actual values are lying between the 95% prediction confidence interval. Thus, the ARIMA(1,1,0) seems to perform well in the prediction.

Prediction vs Actual

```r
forecast_data <- data.frame(date = index(data),
                            price = c(pred, test),
                            predicted = pred,
                            actual = test)
ggplot(forecast_data, aes(x = date, y = predicted)) +
  geom_line(aes(color = 'Predicted')) +
  geom_line(aes(y = actual, color = 'Actual')) +
  xlab(label = 'Time') +
  ylab(label = 'Closing Price (US$)') +
  ggtitle(label = 'S&P 500 Index Closing Price Forecast') +
  theme(legend.title = element_blank())
```



Conclusions:

We set out to determine if there was a meaningful model that could be found, which would be able to predict future stock price of the S&P500 index. In this project, we collected 252 observations from Yahoo Finance API. The data represents the daily stock price from 2019–01–01 to 2020–01–01 (no including 104 days for the weekends when the exchange is closed). We were also able to obtain the actual stock prices for the period of 2020–01–02 to 2020–01–08 (7 values) which we will attempt to forecast using the data from 2019. The actual values compared to our forecasted values will be used as a measure of the quality of our model.

Preliminary data exploration showed that the daily stock price of the S&P500 index was not a stationary series, the data plot showed an obvious upward trend, without seasonality or periodicity. Moreover, the acf showed significant correlation at all lag points. Ljung-Box test statistic was less than the critical values, therefore we concluded that the series was not a realization of a white noise (stationary) process. Before further downstream analysis, we had to transform the original, non-stationary, data series into a stationary series.

The series was transformed using diff = 1, generate a new series. When plotted, the new series data values centered around mean zero, with no obvious trend. The acf and pacf plots show significant values at lag 7, 22,and 23. After doing the Ljung-Box test, we concluded that the new series is a realization from a stationary process, and that the significant values in the acf and pacf were likely to be outliers.

Next we wanted to find the optimal model for forecasting. To do this, we enumerated over possible values of p and q (d = 1), constrainted by p >= 0 and p <= 4, and q >= 0, and q <= 4. We used a leave-one-out cross-validation approach quantify the accuracy of each model. The chosen model used ARIMA(1, 1, 0), because it has the lowest RMSPE (22.62201), and the second lowest AIC (1159.596). Fitting the AR(1) model to the full dataset returned an alpha coefficient of -0.0849. Residual analysis on the fitted model shows that the residuals follow a realization of a white noise process, as expected from a good fitting model.

We forecasted the first five stock prices from the year 2020, from the period of 2020–01–02 to 2020–01–08 (less 1 weekend). The actual values are shown to fall within the 95% confidence interval of the forecasted values. As the time increases into the future, the confidence interval widens, as expected. Overall, we conclude that an AR(1) model is a good approximatation for the daily stock price of the S&P500 index for the year of 2019. The AR(1) model is also accuracte for forecasting purposes, up until the first week of 2020.

During post-analysis, we looked at various years through out the S&P500 data and realized that we had been extremely fortunate selecting 2019 as our training dataset. The trends in other years were not as clear and there is no guarantee that using other years as training would yield an AR(1) as the best fitted model. Furthermore, the stock market experiences various periods of extreme growth, stagnation, and crashes, however there was no such novel event during the year of 2019. Taking into account these extenuating circumstances, it is extremely unlikely that an AR(1) would be the best fitted model for the S&P500 in general.

Moving forward, we would like to try to use the entire S&P500 dataset from inception in 1926 until 2019. This may yield a more accurate model of the S&P500 stock index. However, even that model is unlikely to be able to predict market crashes like what we are currently experiencing due to the Covid-19 global pandemic. More sophisticated models which can account for anomaly detection would be necessary in order to generate a model which can accurately model the S&P500 index.