Stock Forecast

Syeda Farha Shazmeen , Priya Kurian pullolickal

February 16, 2018

1.Introduction

The stock market allows companies to raise money by offering stock shares and corporate bonds. It is a combination of buyers and sellers. Stock forecast/prediction is a practice of trying to predict the fututre value of a company's stock or trading on an exchange. If the stock prediction is successful it can lead to huge profit to the trader.

In this assignment we have created a program that can forecast the price of a given stock for a given no of days.We have taken only the closing price of the given stock for prediction.

2.Getting and Visualizing Stock Data

Before working with stock market data, collecting data is very important and data analysis can help in understanding the methods and frameworks to be used. For our analysis we have considered **APPLE STOCK** from Yahoo! Finance since 01-01-2017 to 14-02-2018. The **quantmod** package provides easy access to Yahoo! finance and useful features for financial modelling.

2.1 Getting Data from Yahoo! Finance with quantmod

```
library(quantmod)
library(tseries)
library(forecast)
library(MASS)
library(ggplot2)
start <- as.Date("2017-01-01")
end<-as.Date("2018-02-14")
t=getSymbols("AAPL", src = "yahoo", from = start ,to=end)
AAPL=AAPL[-281,]
n=dim(AAPL)[1]
set.seed(12345)
head(AAPL)
```

```
##            AAPL.Open AAPL.High AAPL.Low AAPL.Close AAPL.Volume
## 2017-01-03    115.80    116.33   114.76     116.15    28781900
## 2017-01-04    115.85    116.51   115.75     116.02    21118100
## 2017-01-05    115.92    116.86   115.81     116.61    22193600
## 2017-01-06    116.78    118.16   116.47     117.91    31751900
## 2017-01-09    117.95    119.43   117.94     118.99    33561900
## 2017-01-10    118.77    119.38   118.30     119.11    24462100
##            AAPL.Adjusted
## 2017-01-03      113.8476
## 2017-01-04      113.7202
## 2017-01-05      114.2985
## 2017-01-06      115.5727
## 2017-01-09      116.6313
```

```
## 2017-01-10      116.7489
```

The **getsymnols()** create a global environment called AAPL which has stock market data for APPLE since 2017-01-01 to 2018-02-14. This dataset contains 6 attributes. On a trading day ,**AAPL.Open** is the Opening price of the stock at the start of day. **AAPL.High** is the highest price of the stock. **AAPL.Low** is the lowest price of the stock. **AAPL.high** is the price of the stock at closing time. **AAPL.Volume** is the number of stock traded. **AAPL.Adjusted** is the closing price of the stock that adjusts the price of the stock for corporate actions.

1.1 Linear Regression for Stock Prediction

We have considered The simple and the most common model for prediction i.e., linear model/Regression. It is an approach for finding the relationship between indepedent variable i.e., AAPL.open and dependent variable i.e., AAPL.Close.

```
## The mean square error using linear regression is:  16.76692
```

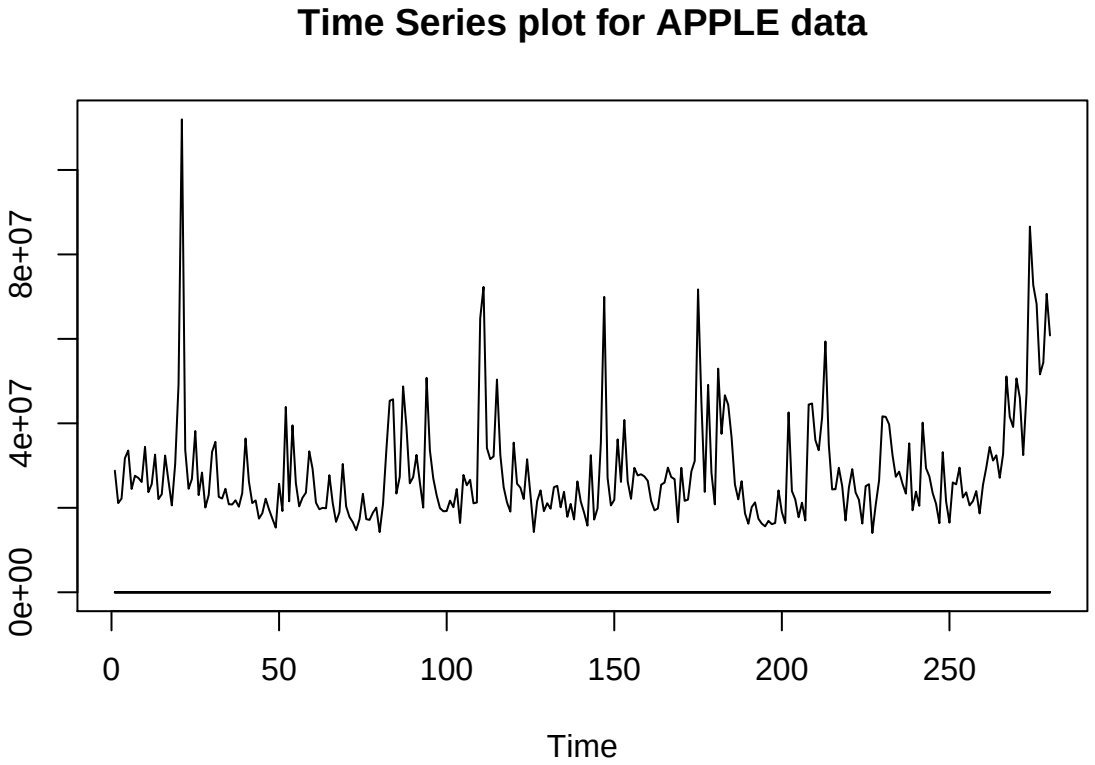
we could observe that the MSE value is vey high.so linear model may not be an appropriate choice for our data.

Why Time Series Model?

Time series is a series of observations made over a certain time interval. It is commonly used in economic forecasting as well as analyzing sesonal trends in large periods of time. The main idea is to use a certain number of previous observations to predict future observations.

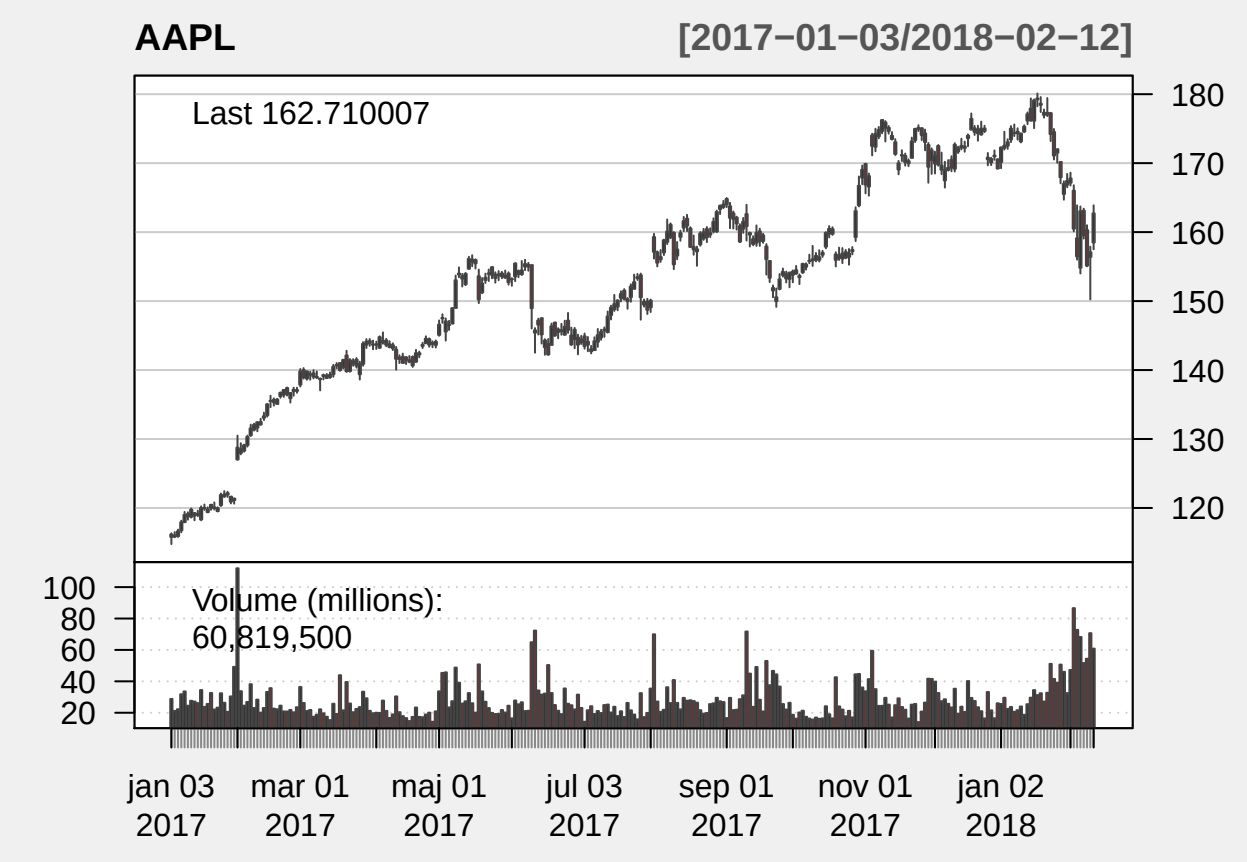
2.1 Stock Data Visulaization

For plotting several time series objects on a common plot we can use **ts.plot**. The plot below shows the different attributes of AAPL in time series plot.



The time series data can be viewed using standard financial chart called as CandleChart. candleChart() is from quantmod package. From the below plot we can see black candlestick shows a day were the closing price was higher than the open which indicates a gain. Where as a red candlestick shows a day where the open price is higher than closing price which indicates loss.

With a candlestick chart, a black candlestick indicates a day where the closing price was higher than the open (a gain), while a red candlestick indicates a day where the open was higher than the close (a loss). The wicks indicate the high and the low, and the body the open and close (hue is used to determine which end of the body is the open and which the close).



2.2 Moving Averages

Moving averages is the forecasting approach that averages values over multiple periods. It reduces noise and finds the patterns from data. Its' smoothing techniue generates averages from the consecutive time periods.

A q-day moving average is, for a series x_t and a point in time t, the average of the past q days: that is, if MA_t^q denotes a moving average process, then:

$$MA_t^q = \frac{1}{q}\sum_{i=0}^{q-1} x_{t-i}$$

Traders are always interested in the moving averages for different periods.In the above plot the moving average for 20 days is calculated.The moving average gives a smoother line when compared to the actual data.The stock data can be observed above or below the moving average.Like 20 days we could also calculate the moving averages for 50,100 or 200 days.

3.1 Check if Data is Stationary

Before performing the time series modeling we want to check if the data is stationary because non-stationary data cannot be suitable for making accurate predictions. For the data to be stationary it should have following:

1.**Constant mean** across all time-t.

2.**Constant variance** across all time-t.

3.**Autocovariance** - Distance between two observations depends on autocovariance, which we will refer to as the lag(h).

Log transformation can be used on time series data to make constant variance or to make its distribution more normal. It may be better modelled with standard linear autoregressive integrated moving average (ARIMA) processes.

Before applying ARIMA model we have to do the following steps:

3.2 Augmented Dickey-Fuller Test

This test is performed to verify whether a series is stationary or not? We perform it using **adf.test()**. This tests the null hypothesis. It rejects the null hypothesis when the p-value is less than or equal to specified significance level. We can consider a significance level of **5%**. when we perform the adftest for our time series data following are the results:

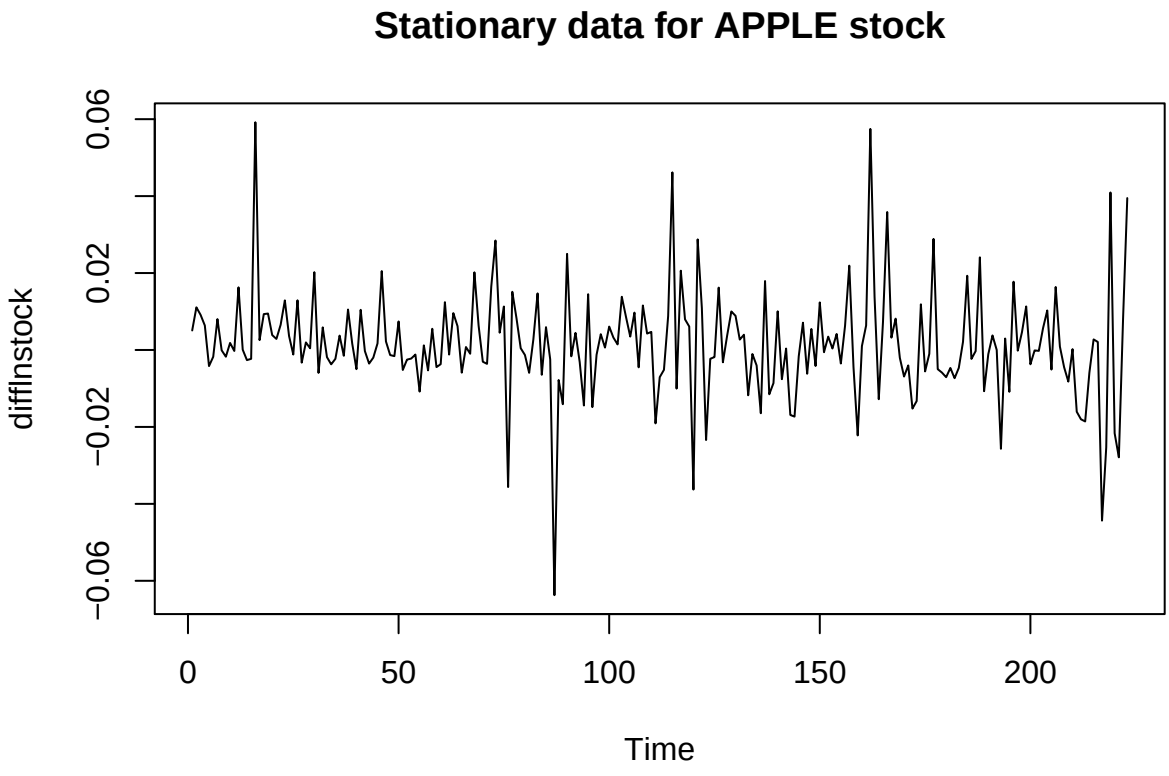
```
    Augmented Dickey-Fuller Test

data:  lnstock
Dickey-Fuller = -2.7803, Lag order = 6, p-value = 0.2483
alternative hypothesis: stationary
```

Here we can see the p-value of **0.2483**. Hence we fail to reject the null hypothesis.

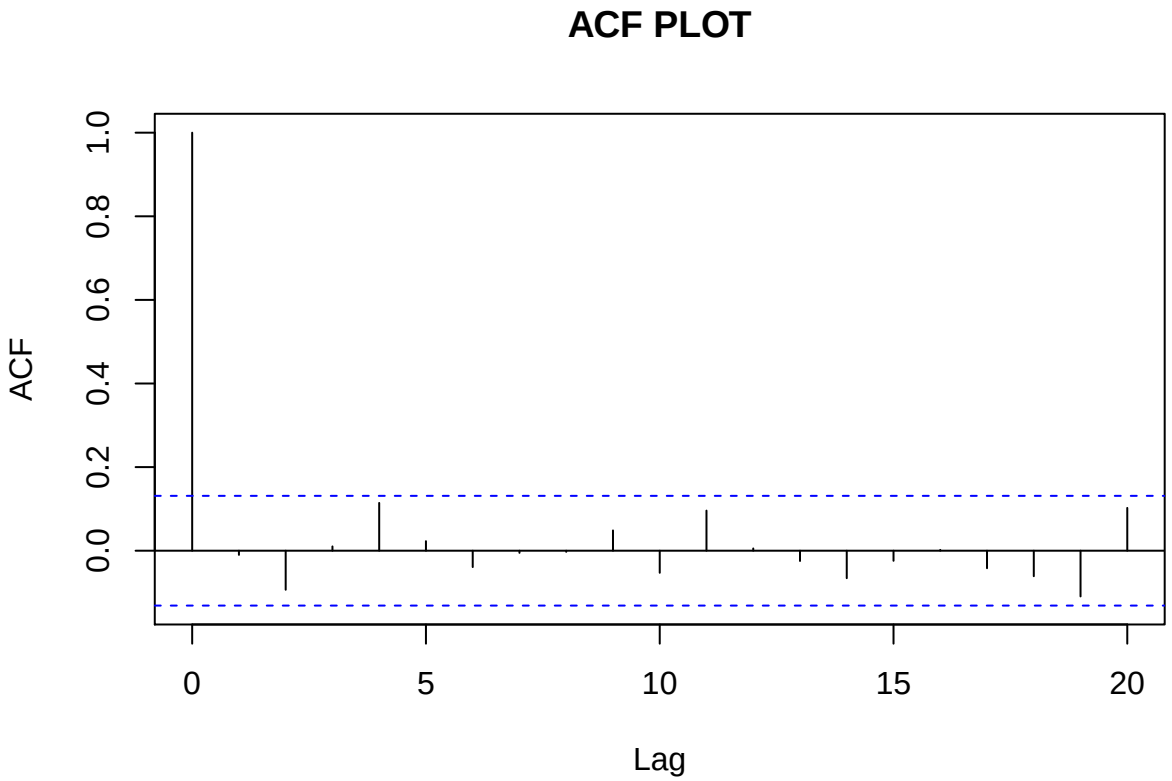
3.3 Correlation Functions:

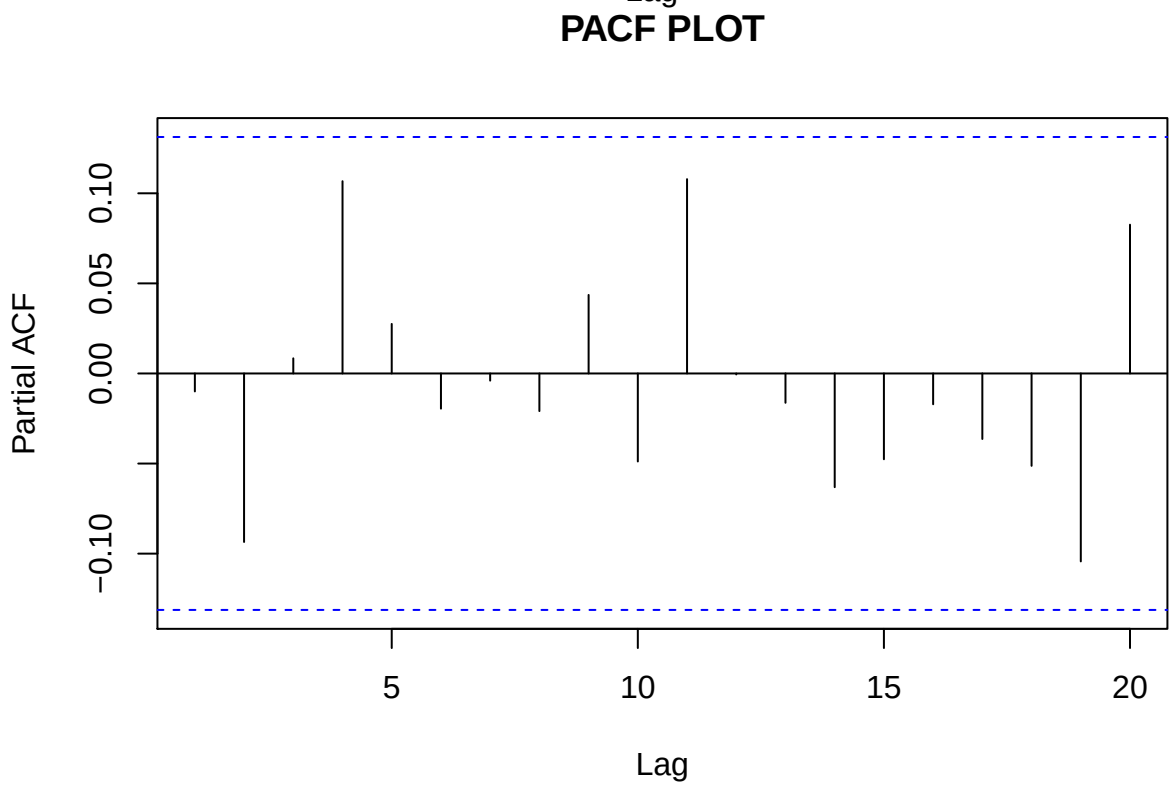
One common function used on non stationary time series data is finding difference of the series. The difference is to see it as $x(t) - x(t-k)$ where k is the number of lags go back. In R, the difference operator for xts is made available using the **diff()** command. The second argument is the order of the difference. The plot below shows the AAPL.close data which is stationary



3.3.1 ACF -Auto Correlation Function

The function **acf()** computes an estimate of the autocorrelation function of a mulivariate time series.In R acf starts with lag 0, that is the correlation of a value with itself. Function **Pacf()** computes an estimate of the partial autocorrelation function of a mulivariate time series. pacf starts at lag 1.


ACF PLOT
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
Lag


PACF PLOT
Partial ACF
−0.10
0.00
0.05
0.10
5
10
15
20
Lag

4. ARIMA Model

ARIMA stands for auto-regressive integrated moving average and is specified by these three order parameters: (p, d, q).We can perform the auto.arima function on the data which returns the best ARIMA model according to either AIC/BIC to fit the model.

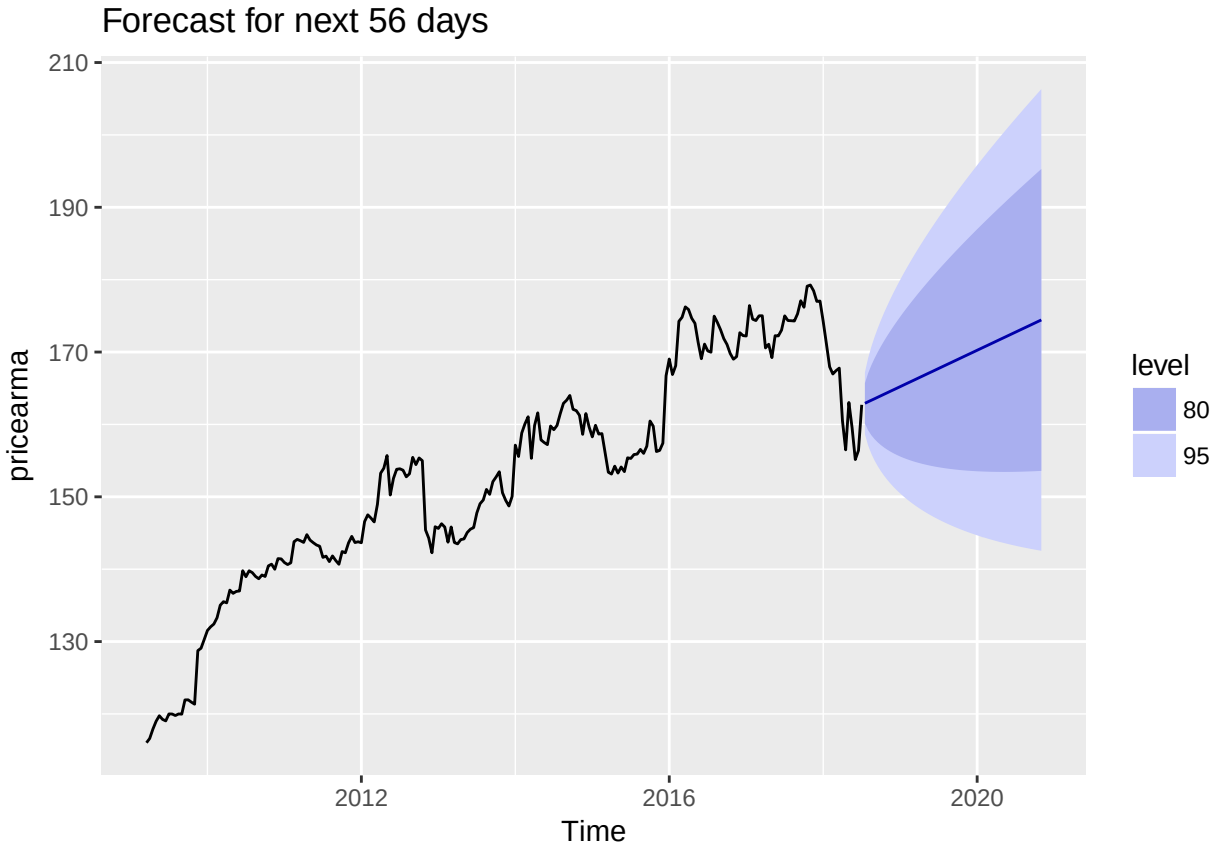
```
Series: pricearma
ARIMA(0,1,0) with drift

Coefficients:
       drift
      0.2094
s.e.  0.1453

sigma^2 estimated as 4.727:  log likelihood=-489.11
AIC=982.23   AICc=982.28   BIC=989.04

Training set error measures:
                      ME     RMSE     MAE         MPE      MAPE      MASE
Training set 0.0005170115 2.164452 1.43735 0.003951843 0.9257345 0.1648914
                     ACF1
Training set -0.007144951
```

The forecast function can be used to forecast from time series model. We can see the prediction interval.



If we consider the **Mean Percentage Error** is approximately 7%. It shows roughly 7% deviation on average from original price and forecasted price. Below shows the original and forecasted price.

```
ActualPrice ForecastPrice
```

```
2017-05-08      153.01      160.7019
2017-05-09      153.99      160.1297
2017-05-12      156.10      161.0566
2017-05-16      155.47      160.5660
2017-05-19      153.06      159.9357
2017-05-22      153.99      160.1181
2017-05-24      153.34      161.0120
```

```
The Mean Percentage error with Arima model is -0.0733415
```

Conclusion

In this assignment the task was to forecast the future price of a particular stock.We used linear regression and Arima Models to do the forecast.The prediction done using the linear model had an error rate of 16.76%.The stock market data varies with time which makes times series model a more appropriate one.When we performed forecasting using the Arima model we got an error rate of 7%.This matches with our expectation than a time series model is appropriate for stock market forecast.The weakness of our model is that we have assumed that we have used the closing price as the only parameter to preict the future price.In the real world the future price is affected by more parameters like news about the financial results,new contracts etc.

Future work

As described in section 4 we chose the ARIMA Model using the auto.arima function in R.The arima model returned by the auto.arima function may not be always the optimal model for predicting the stock price.All the possible Arima models can be obtained by specifying all possible values for p,q and d inside sarima function.Then find the most suitable model for our data from that.May be this can give a better error rate than the auto.arima function.

References

1.https://www.investopedia.com/terms/s/stockmarket.asp#ixzz57UH6q1Ec

2.https://www.investopedia.com/terms/s/stockmarket.asp

3.http://beancoder.com/linear-regression-stock-prediction/

4.https://ntguardian.wordpress.com/2017/03/27/introduction-stock-market-data-r-1/

Appendix

```
#Linear Regression
 #dividing the data into 50/50 (training/testing)
library(quantmod)
library(tseries)
library(forecast)
library(MASS)
library(ggplot2)
start <- as.Date("2017-01-01")
end<-as.Date("2018-02-14")
t=getSymbols("AAPL", src = "yahoo", from = start ,to=end)
AAPL=AAPL[-281,]
n=dim(AAPL)[1]
set.seed(12345)
trainIndex=sample(1:n, floor(n*.5))
train_data=AAPL[trainIndex,]
test=AAPL[-trainIndex,]
##making the linear model
model=lm(AAPL.Close~AAPL.Open,train_data)
##making a prediction
prediction=predict(model,newx=test)
MSE<-mean(((test[,"AAPL.Close"])-prediction)^2)
cat("The mean square error using linear regression is: ",MSE)


#First 10 rows of data
head(AAPL)
#Time series plot for stock data
ts.plot(AAPL,main="Time Series plot for APPLE data")
#candle chart
candleChart(AAPL, up.col = "black", dn.col = "red",
            theme = "white",main="Candle Chart for APPLE stock data")

#adding moving average as 20 to candle chart
candleChart(AAPL, up.col = "black", dn.col = "red",
            theme = "white", subset = "2016-01-04/",main="Moving Average for 20 days")
addSMA(n = 20)

####Arima model

#divinding data into 80 for training and 20% for testing
stock<-AAPL[-281,]
#dividing the data into 50/50 (training/testing)
n=dim(stock)[1]
set.seed(1234)
trainIndex=sample(1:n, floor(n*.80))
train_data=stock[trainIndex,]
test=stock[-trainIndex,]
lnstock=log(train_data[,"AAPL.Close"])
#Dickey Fuller Test
adf.test(lnstock)
#Difference of stack
```

```
difflnstock<-diff(lnstock,1)
difflnstock = difflnstock[!is.na(difflnstock)]
ts.plot(difflnstock,main="Stationary data for APPLE stock")
#PACF and ACF correlation
acf(difflnstock,lag.max=20,main="ACF PLOT")
pacf(difflnstock,lag.max=20,main="PACF PLOT")

# Finding Arima model using auto.arima
pricearma<-ts(exp(lnstock),start=c(2009,06), frequency = 20 )
model<-auto.arima(pricearma)
summary(model)

forecast1<-forecast(model,h=56)
autoplot(forecast1, main="Forecast for next 56 days ")
pricearma<-ts(lnstock,start=c(2009,06), frequency = 20 )
model1<-auto.arima(pricearma)

#exponential of lnstock

lnexp<-exp(lnstock)

#forecasting
forecast2<-forecast(model1,h=56)

#mean of forecasted
m<-as.numeric(forecast2$mean)
eexpm<-exp(m)

#error

data1<-data.frame(test[,"AAPL.Close"],eexpm)
colnames(data1)<-c('ActualPrice','ForecastPrice')
data1[14:20,]

##forecast error
MSE<-(data1$Actual-data1$Forecast)/(data1$Actual)
MSE_error<-mean(MSE)
cat("The Mean Percentage error with Arima model is",MSE_error)
```