Project Title: **Time series analysis and arima modelling on stock prices of Reliance Industries Limited**

CONTENTS

Abstract

This project report includes the time series analysis on the stock prices of Reliance Industries Limited from the year 1995-2020 and construction of an optimal arima model using closing price for the year 2020 on Rstudio. The project includes the explanation of time series analysis,stock market and arima modelling. Then the variables of the dataset is explained. Then we will see the objectives of the project and the methodology applied. The project was performed on Rstudio . It includes the libraries loaded, data preparation and cleaning. It also includes data exploration. Then a filtered data for the year 2020 with the variable closing price can be observed which would be later gets transformed into a time series data. The time series data was tested for stationarity using the augmented dickey fuller test. But it was not stationary, so it was stationarize using logarithmic and differencing technique.And then it was again tested for stationarity. The arima modelling was then done using the autocorrelation plots and auto.arima function in Rstudio. Then the forecasting was done for the next 60 days. And atlast the model adequacy analysis was done using the residuals obtained from the model and then the project is concluded.

Introduction

➢ Time series analysis

When the observations are kept in timely manner in a form of series than it is called time series. When the data which is based on time series is analyzed to obtain useful statistics and some other characteristics of the data using different methods, it is called time series analysis. Also, a arima model is used to forecast future values which are taken on the basis of recently observed values, this method is called time series forecasting.

➢ Stock market

Stock market is where individuals purchase/sell portions of openly recorded organizations. It offers a stage to work with consistent trade of offers. The stock market is a road where financial backers exchange offers, bonds, and subsidiaries. This exchanging is worked with by stock trades, which can be considered as business sectors that associate purchasers and merchants. Stocks are units of an organization's fairly estimated worth. Investors are people who buy stocks to turn out to be part proprietors in the organization. Exchanging includes purchasing or selling this value. In any case, note that an individual can exchange the financial exchange just through an enrolled middle person known as a stock merchant. The purchasing and selling of offers occur through electronic medium.

➢ ARIMA modelling

Auto Regression Integrated Moving Average additionally knows as ARIMA, in this model the previous qualities are utilized to foresee the future estimations of the information. It has 3 request boundaries (p, d, q). Auto-Regression is utilizing past qualities to anticipate future qualities and is indicated by 'p' .Moving Average is utilizing the previous blunders to foresee the future estimations of the details and is signified by 'q'. The level of differencing is signified by 'd'.

Dataset details

I have downloaded the data on stock prices of Reliance Industries Limited from 1995-2020 from the National Stock Exchange of India Ltd.. The description of columns in the file is as follows:-

- Symbol — Organisation's name(Reliance)
- Series — We have many types of series but we will filter it to only one series i.e. EQ.
- Date — Date of trade
- Prev.Close — Alludes to the earlier day's end cost of a stock
- Open.Price — Starting price of a stock trading
- High.Price — Highest price in a stock trading day
- Low.Price — Lowest price in a stock trading day
- Last.Price — Last price of the stock
- Close.Price — Finishing cost of exchanging on a stock trade when the business sectors close.
- Average.Price(VWAP)—It is the volume weighted average price
- Total.Traded.Quantity(Volume) — Total volume of the stock traded
- Turnover —It is a measure of buyers versus sellers of a particular share.
- No..of.Trades—Total no. of shares traded in a single day
- Deliverable.Qty— Volume of offers moved starting with one demat account then onto the next
- X..Dly.qt.to.Traded.Qty — Stocks that are moved starting with one demat account then onto the next.

Objectives of the project

- To conduct a time series analysis on the stock prices of Reliance Industries Limited from the year 1995-2020 on Rstudio
- To build an ideal arima model utilizing shutting cost for the year 2020 on Rstudio
- To forecast the end cost for the following 60 days and then to check the adequacy of the model.

Methodology

The project is performed on Rstudio and the initial step is stacking of required libraries. At that point we will plan and clean the information on the Rstudio. Data exploration will be done to contemplate the stock costs concerning time. At that point we will filter the data for the end cost for the year 2020, and afterward it will be changed into a time series data. Then the data will be tried for stationarity utilizing the augmented dickey fuller test. Yet, in the event that it gets come about out as a non-fixed information, it will be stationarized utilizing logarithmic and differencing technique.And then it will be again tried for stationarity. At that point we will perform arima displaying utilizing the autocorrelation plots and auto.arima work in Rstudio. At that point we will forecast the end cost for the following 60 days. Also, atlast we will play out a model sufficiency investigation utilizing the residuals obtained from model and after that we will give the end and references.

Loading Libraries

```
library(plyr)
library(dplyr)
library(ggplot2)
library(gridExtra)
library(forecast)
library(tseries)
library(xts)
```

Data Preparation and Cleaning

```
> setwd("C:/Akshit/sem6/ntcc/Reliance_stocks")
> dataset <- ldply(list.files(), read.csv, header=TRUE)
> head(dataset)
    Symbol Series        Date Prev.Close Open.Price High.Price Low.Price Last.Price
1 RELIANCE     EQ 01-Jan-1996     204.65     205.00     206.10    203.65          -
2 RELIANCE     N1 01-Jan-1996      71.50      71.50      70.80     70.80          -
3 RELIANCE     N2 01-Jan-1996     123.00     124.00     124.00    124.00          -
4 RELIANCE     N7 01-Jan-1996      90.00      90.00      90.00     90.00          -
5 RELIANCE     NB 01-Jan-1996      96.95     100.00     100.00    100.00          -
6 RELIANCE     EQ 02-Jan-1996     205.75     205.25     206.25    202.65          -
  Close.Price Average.Price Total.Traded.Quantity   Turnover No..of.Trades Deliverable.Qty
1      205.75        205.26               3717450  763051385             -               -
2       70.80         70.80                    70       4956             -               -
3      124.00        124.00                    10       1240             -               -
4       90.00         90.00                   100       9000             -               -
5      100.00        100.00                    50       5000             -               -
6      204.15        204.13               6024650 1229789050             -               -
  X..Dly.Qt.to.Traded.Qty
1                       -
2                       -
3                       -
4                       -
5                       -
6                       -
```

First, I setted up the working directory and combined all the 25 excel files into a single dataset and named it as “dataset”. Then used the head function to observe the names and the first six observations of each column in the dataset. Also, we can spot that columns- Last.Price, No. of Trades, Deliverable.Qty and X.Dly.Qt.to.Traded.Qty have blank values in the first six rows.

```
> dim(dataset)
[1] 9005   15
```

Dim function gives us the dimensions of our data i.e. total number of rows and columns. So, we have 9005 number of rows and 15 number of columns.

```
> str(dataset)
'data.frame':   9005 obs. of  15 variables:
 $ Symbol                : chr  "RELIANCE" "RELIANCE" "RELIANCE" "RELIANCE" ...
 $ Series                : chr  "EQ" "N1" "N2" "N7" ...
 $ Date                  : chr  "01-Jan-1996" "01-Jan-1996" "01-Jan-1996" "01-Jan-1996" ...
 $ Prev.Close            : num  204.7 71.5 123 90 97 ...
 $ Open.Price            : num  205 71.5 124 90 100 ...
 $ High.Price            : num  206.1 70.8 124 90 100 ...
 $ Low.Price             : num  203.7 70.8 124 90 100 ...
 $ Last.Price            : chr  "-" "-" "-" "-" ...
 $ Close.Price           : num  205.8 70.8 124 90 100 ...
 $ Average.Price         : num  205.3 70.8 124 90 100 ...
 $ Total.Traded.Quantity : int  3717450 70 10 100 50 6024650 1130 500 20 140 ...
 $ Turnover              : num  7.63e+08 4.96e+03 1.24e+03 9.00e+03 5.00e+03 ...
 $ No..of.Trades         : chr  "-" "-" "-" "-" ...
 $ Deliverable.Qty       : chr  "-" "-" "-" "-" ...
 $ X..Dly.Qt.to.Traded.Qty: chr  "-" "-" "-" "-" ...
```

Structure function compactly displays the structure of our data. We can observe here that we have 15 variables out of which 7 variables have character class, other 7 variables have numeric class and only 1 variable has integer class.

Also, we can spot that the Date variable is in character format. So, I will convert it into date format. Also, I will keep only the necessary variables required for analysis.

```
> stock_data=filter(dataset,Series=="EQ")
> stock_data=select(stock_data,Date,Open.Price:Low.Price,Close.Price:Total.Traded.Quantity)
> stock_data=stock_data%>%rename(VWAP=Average.Price,Volume=Total.Traded.Quantity)
> stock_data$Date=as.Date(stock_data$Date,format="%d-%b-%Y")
> str(stock_data)
'data.frame':   6228 obs. of  7 variables:
 $ Date       : Date, format: "1996-01-01" "1996-01-02" "1996-01-03" ...
 $ Open.Price : num  205 205 208 204 203 ...
 $ High.Price : num  206 206 217 204 203 ...
 $ Low.Price  : num  204 203 205 201 201 ...
 $ Close.Price: num  206 204 206 204 202 ...
 $ VWAP       : num  205 204 207 202 202 ...
 $ Volume     : int  3717450 6024650 7473500 7744000 5952000 6675550 13880150 9875700 14625600 13377500 ...
```

Here, i filtered the equities(EQ) series from all the mixed series and named the new dataset as stock_data. Then in stock_data, i selected the necessary variables required for analysis, renamed the column Average.Price as VWAP(volume weighted average price and Total.Traded.Quantity as Volume and atlast converted the class of the Date variable from character to date. Finally, I displayed the structure of the whole new dataset called stock_data. So, now we have 6228 rows and 7 variables in our dataset and all are in proper format.

Data Exploration

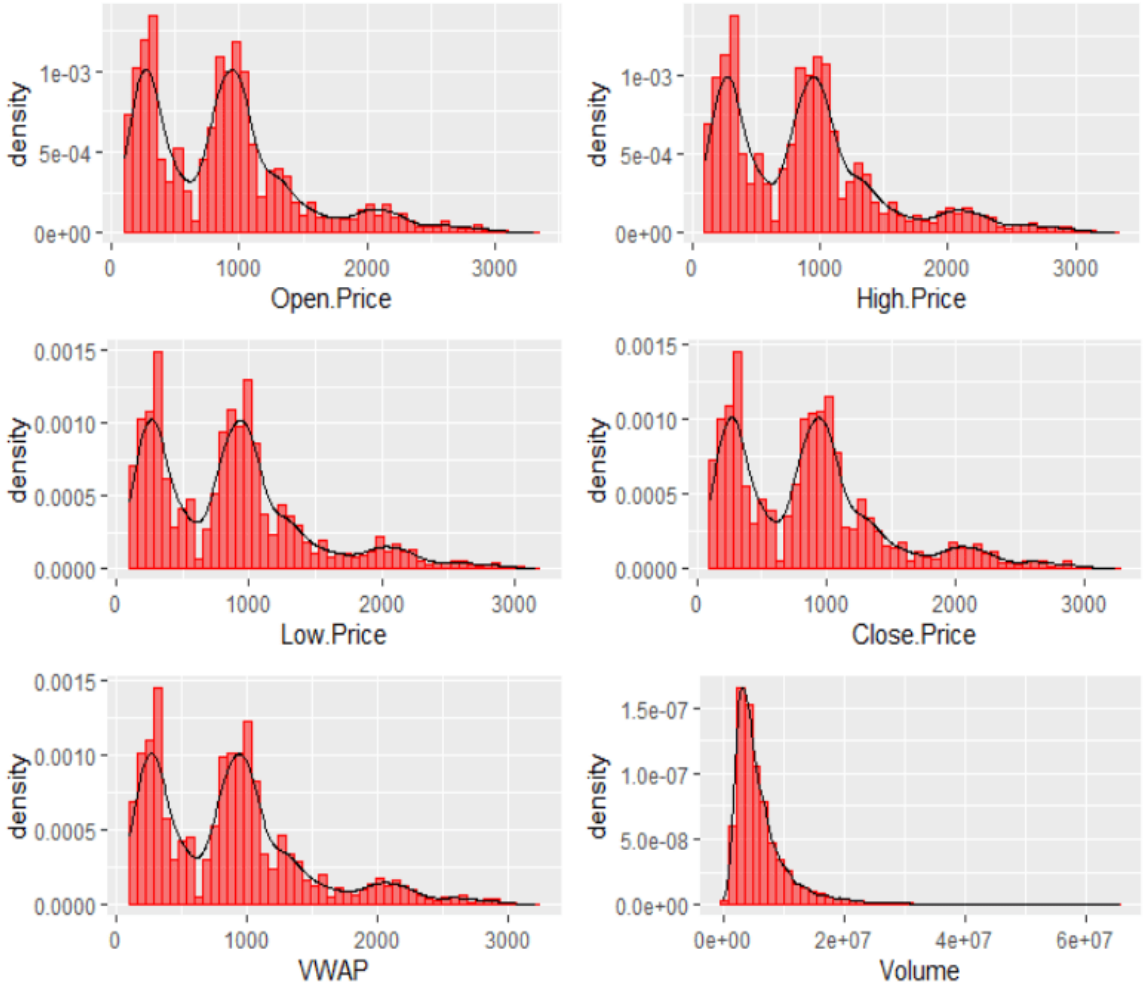
1. Descriptive statistics:

```
> summary(stock_data)
      Date             Open.Price      High.Price      Low.Price       Close.Price         VWAP       
 Min.   :1996-01-01   Min.   : 102.6   Min.   : 105.6   Min.   :  99.6   Min.   : 101.3   Min.   : 101.8  
 1st Qu.:2002-03-27   1st Qu.: 336.0   1st Qu.: 340.9   1st Qu.: 330.0   1st Qu.: 336.5   1st Qu.: 336.7  
 Median :2008-06-10   Median : 854.1   Median : 864.0   Median : 843.0   Median : 852.9   Median : 853.2  
 Mean   :2008-06-22   Mean   : 870.5   Mean   : 882.9   Mean   : 856.9   Mean   : 869.4   Mean   : 870.0  
 3rd Qu.:2014-09-12   3rd Qu.:1095.0   3rd Qu.:1111.0   3rd Qu.:1077.4   3rd Qu.:1092.1   3rd Qu.:1092.7  
 Max.   :2021-01-01   Max.   :3298.0   Max.   :3298.0   Max.   :3141.3   Max.   :3220.8   Max.   :3197.8  
     Volume        
 Min.   :   52820  
 1st Qu.: 3024896  
 Median : 4612591  
 Mean   : 6230383  
 3rd Qu.: 7354147  
 Max.   :65230894
```

The summary function provided us the descriptive statistics of each column of our data. We can see here, the data starts from the date '1996-01-01' and ends at '2021-01-01'. Also, we can observe the minimum,maximum, 1st quartile,median,3rd quartile and mean of each variable. We can observe a good raise in the 3rd quartile values from the 1st quartile passing from the median. This tells us that the stock prices are increasing with passage of time.

2. Distribution of the data:

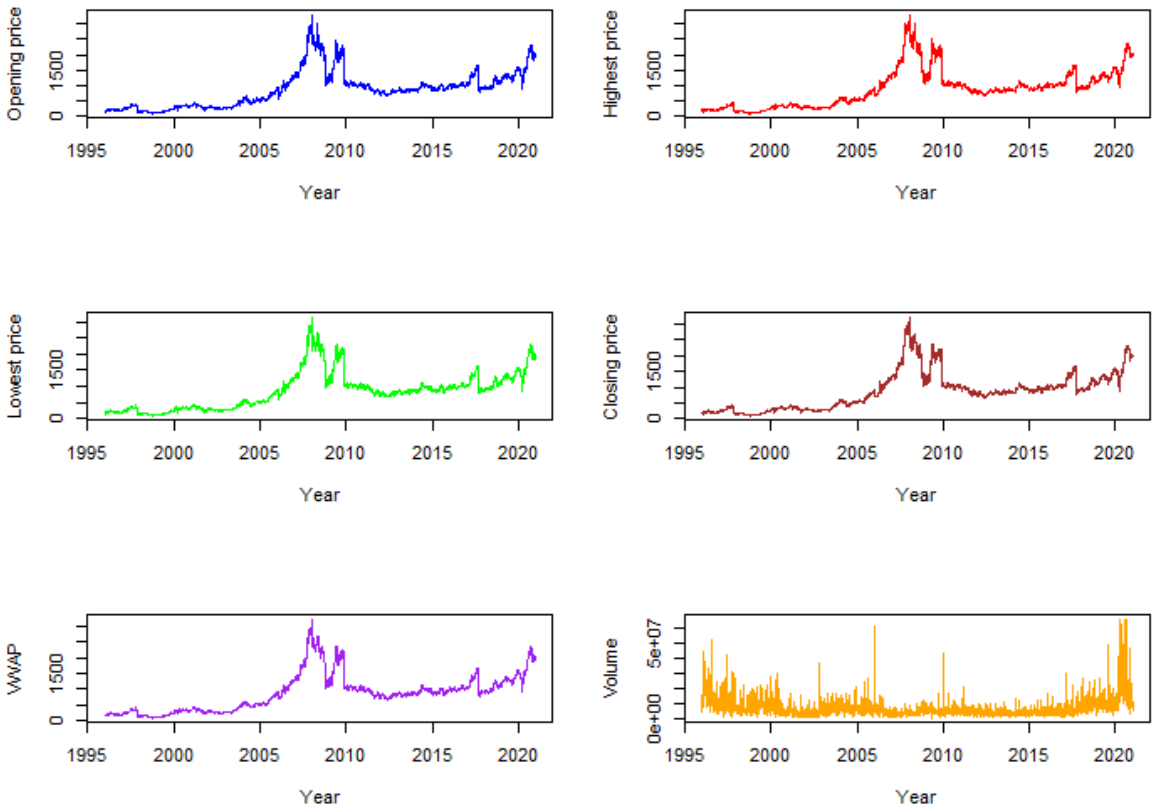
```
> options(repr.plot.width=12, repr.plot.height=12)
> p1 = ggplot(stock_data, aes(Open.Price)) + geom_histogram(bins = 50, aes(y = ..density..),
 col = "red", fill = "red", alpha = 0.5) + geom_density()
> p2 = ggplot(stock_data, aes(High.Price)) + geom_histogram(bins = 50, aes(y = ..density..),
 col = "red", fill = "red", alpha = 0.5) + geom_density()
> p3 = ggplot(stock_data, aes(Low.Price)) + geom_histogram(bins = 50, aes(y = ..density..),
 col = "red", fill = "red", alpha = 0.5) + geom_density()
> p4 = ggplot(stock_data, aes(Close.Price)) + geom_histogram(bins = 50, aes(y = ..densit
y..), col = "red", fill = "red", alpha = 0.5) + geom_density()
> p5 = ggplot(stock_data, aes(VWAP)) + geom_histogram(bins = 50, aes(y = ..density..), col =
 "red", fill = "red", alpha = 0.5) + geom_density()
> p6 = ggplot(stock_data, aes(Volume)) + geom_histogram(bins = 50, aes(y = ..density..), col
 = "red", fill = "red", alpha = 0.5) + geom_density()
> grid.arrange(p1,p2,p3,p4,p5,p6)
```



All the graphs are rightly skewed . Also, we can spot that the variables opening price, highest price, lowest price,closing price and vwap are showing equal distribution property and all are following bimodal distribution. This is because from the year 1995 to 2005, many of the shutting price values were below 500 and from the year 2010 to 2020, most of the closing price values were nearly 1000.

3. Historic time series:

```
> par(mfrow=c(3,2))
> plot(stock_data$Date,stock_data$Open.Price,xlab="Year",ylab="Opening price",type = "l",col="blue")
> plot(stock_data$Date,stock_data$High.Price,xlab="Year",ylab="Highest price",type = "l",col="red")
> plot(stock_data$Date,stock_data$Low.Price,xlab="Year",ylab="Lowest price",type = "l",col="green")
> plot(stock_data$Date,stock_data$Close.Price,xlab="Year",ylab="Closing price",type = "l",col="brown")
> plot(stock_data$Date,stock_data$VWAP,xlab="Year",ylab="VWAP",type = "l",col="purple")
> plot(stock_data$Date,stock_data$Volume,xlab="Year",ylab="Volume",type = "l",col="orange")
```

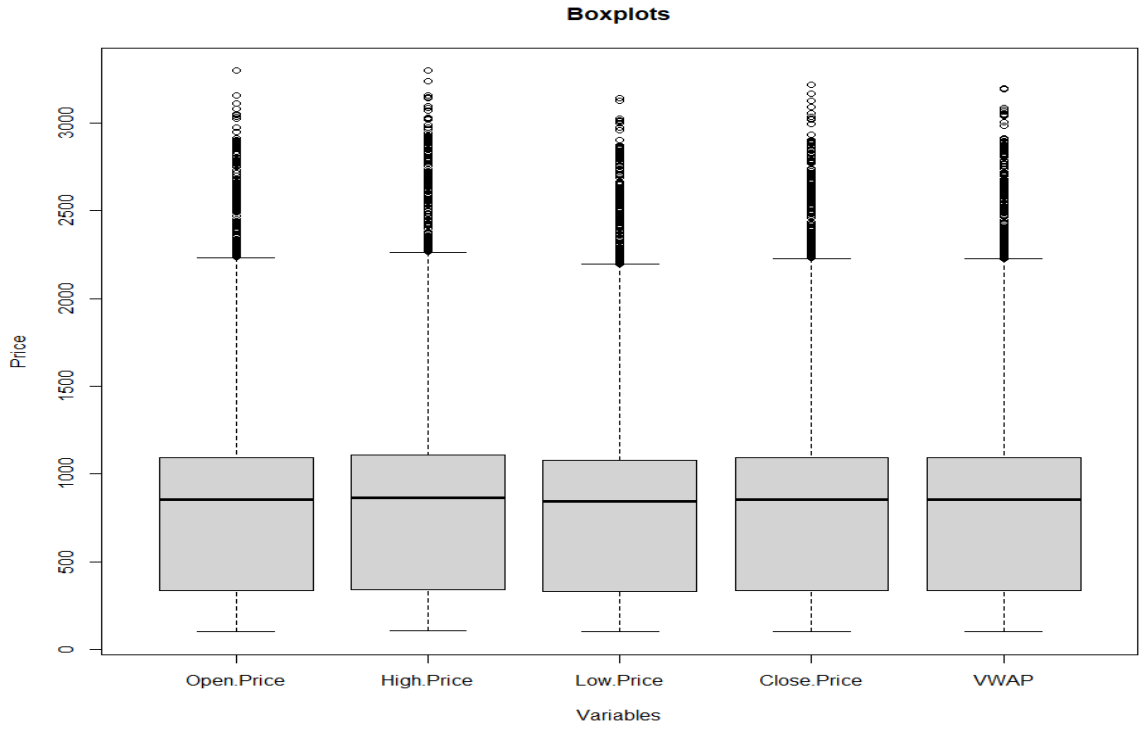


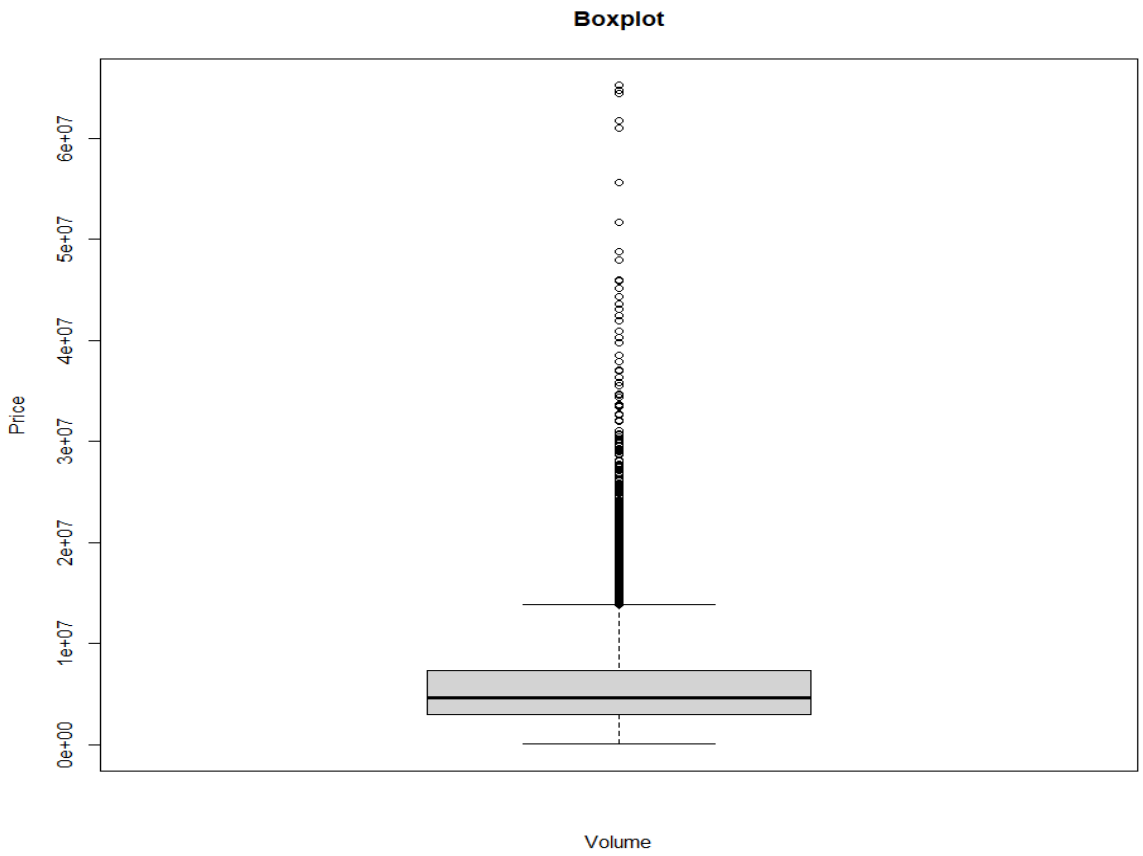
We can see here, the variables opening price, highest price, lowest price,closing price and vwap are showing similar pattern. Every one of them are giving a slow expansion in the pattern throughout the long term. There were two spikes, one in Jan 2008 and another in May-2009. First spike was because Reliance IPO(initial public offering) was sold on 14-Jan-2008. Also reliance launched jio in 2007 after which we can spot an upward trend in its stock prices. Now we can spot that in 2008 the share prices showed a decreasing trend because of the recession period. But after recession, the market again healed. Then a sudden downward is spotted in the end of 2009. Also, another increasing trend can be spotted when jio telecom was introduced in 2016. Then in 2017, another sudden dip is observed. After which it again started increasing till feb 2020, and then in march 2020 it

showed another dip because of the nationwide lockdown which was imposed due to the increasing number of cases of the COVID-19.

Now, in the volume chart, we can observe the volumes trading in 2020 were the highest. The slight stage lie between 2008–2016, in this stage there hasn't been huge volumes exchanged during these years.

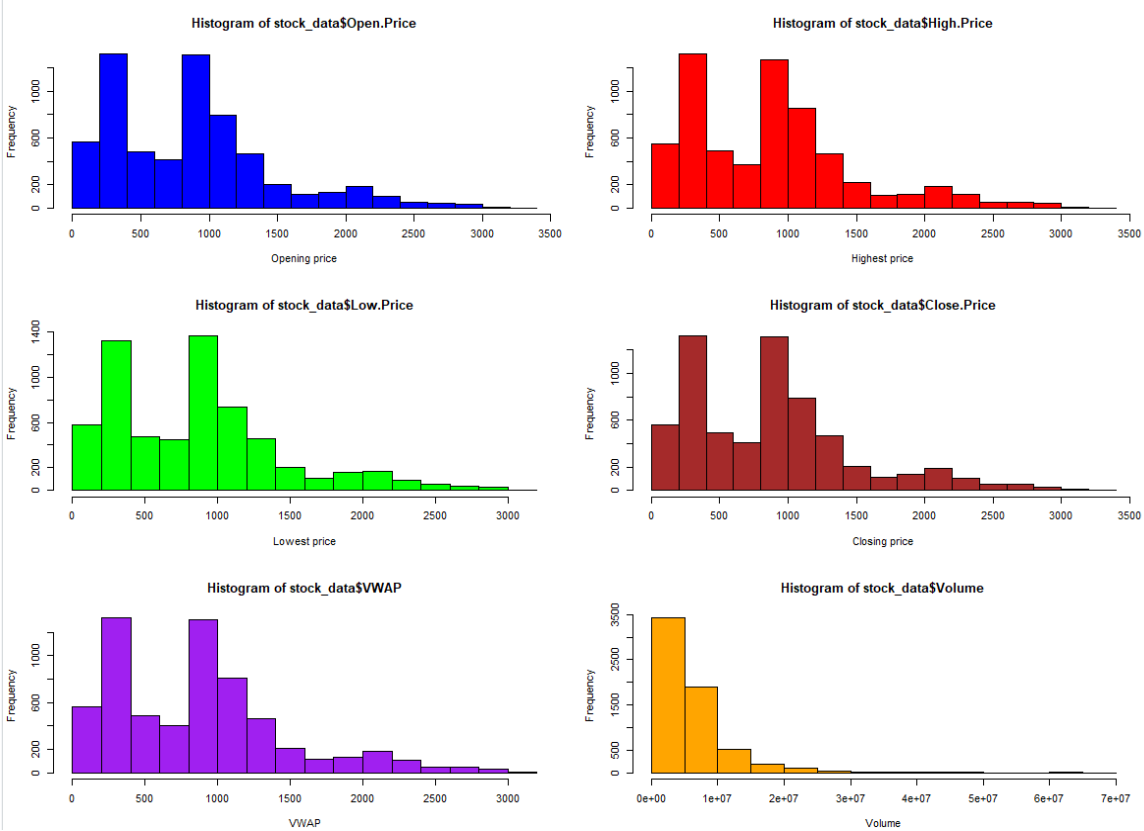
```
> boxplot(stock_data[,-c(1,7)],xlab="Variables",ylab="Price",main="Boxplots")
> boxplot(stock_data[,-c(1,2,3,4,5,6)],xlab="Volume",ylab="Price",main="Boxplot")
```





Now, here we can spot the graphical portrayal of the descriptive statistics using boxplots. The lowest line is the minimum and the highest line is the maximum. Values which are less than minimum and values which are more than maximum are considered outliers. We can calculate minimum and maximum using the formula Q1-1.5*(IQR) and Q3+1.5*(IQR). Here IQR is the inter quartile range which can be calcuted by Q3-Q1. Now we will look at the shutting price boxplot, the bottom side of the box is Q1=336.5 and the top side of the box is Q3=1092.1. The bold line in the box is the median which is 852.9. As this is the data exploration part, we will not remove the outliers and observe the whole data. During the arima modeling, we will remove the outliers if their would be any.

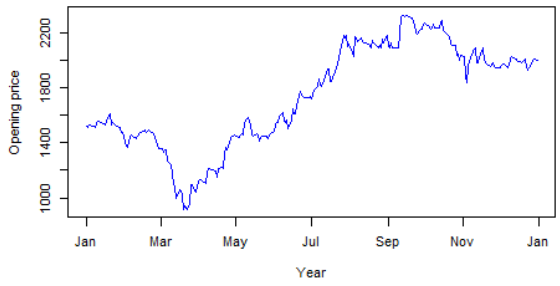
```
> par(mfrow=c(3,2))
> hist(stock_data$Open.Price,xlab="Opening price",col="blue")
> hist(stock_data$High.Price,xlab="Highest price",col="red")
> hist(stock_data$Low.Price,xlab="Lowest price",col="green")
> hist(stock_data$Close.Price,xlab="Closing price",col="brown")
> hist(stock_data$VWAP,xlab="VWAP",col="purple")
> hist(stock_data$Volume,xlab="Volume",col="orange")
```

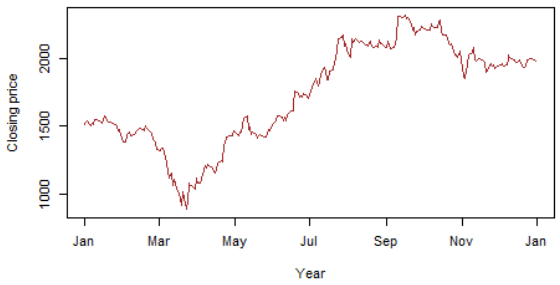


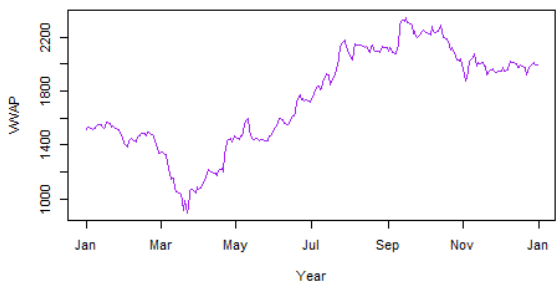
These graphs shows us the frequency distribution of the data using histograms. As we have detected above that the labels opening price, highest price, lowest price,closing price and vwap were showing equal distribution property, here also we can spot that they are showing the same frequency distribution. Also, we observe that more than 1000 times the stock prices were in the range of 200-450 and 750-1000 approximately.

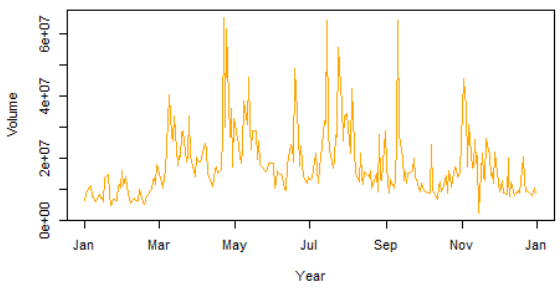
4. For The Year 2020:

```
> par(mfrow=c(3,2))
> data2020=subset(stock_data,subset = stock_data$Date >='2020-01-01'&stock_data$Date <='2020-12-31')
> plot(data2020$Date,data2020$Open.Price,xlab="Year",ylab="Opening price",type = "l",col="blue")
> plot(data2020$Date,data2020$High.Price,xlab="Year",ylab="Highest price",type = "l",col="red")
> plot(data2020$Date,data2020$Low.Price,xlab="Year",ylab="Lowest price",type = "l",col="green")
> plot(data2020$Date,data2020$Close.Price,xlab="Year",ylab="Closing price",type = "l",col="brown")
> plot(data2020$Date,data2020$VWAP,xlab="Year",ylab="VWAP",type = "l",col="purple")
> plot(data2020$Date,data2020$Volume,xlab="Year",ylab="Volume",type = "l",col="orange")
```









Now, I have filtered the data for the year 2020 using the subset function. We can observe that the stock prices showed a gradual fall in the month of march. This is because of the sudden nationwide lockdown which was imposed due to the increasing number of cases of the COVID-19. But just after march, the stock prices went up again because of the various strategies and plans implemented by the organisation. Additionally, this could be because of the interests appeared by Facebook, Google and different organizations on Jio shares.

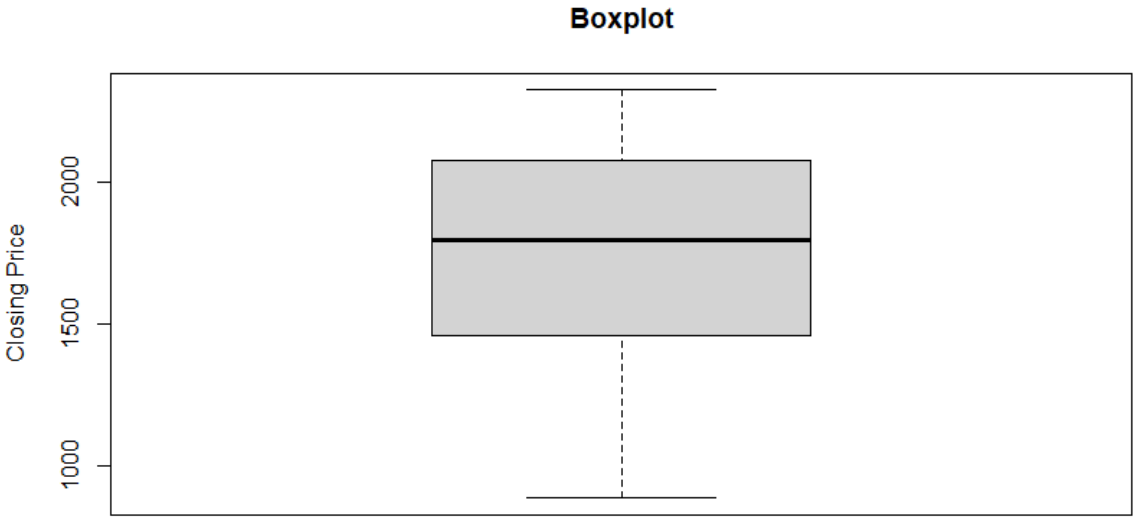
Transforming the data into time series

Now, first we will choose only closing price from the filtered data of year 2020 using the select function.

```
> closedata2020=select(data2020,Date,Close.Price)
```

Now, check for the outliers using boxplot.

```
> boxplot(closedata2020[,-1],ylab="Closing Price",main="Boxplot")
```



We can spot that none of the observations fall outside the minimum and maximum observations. So, the dataset is ready to be transformed into a time series object.

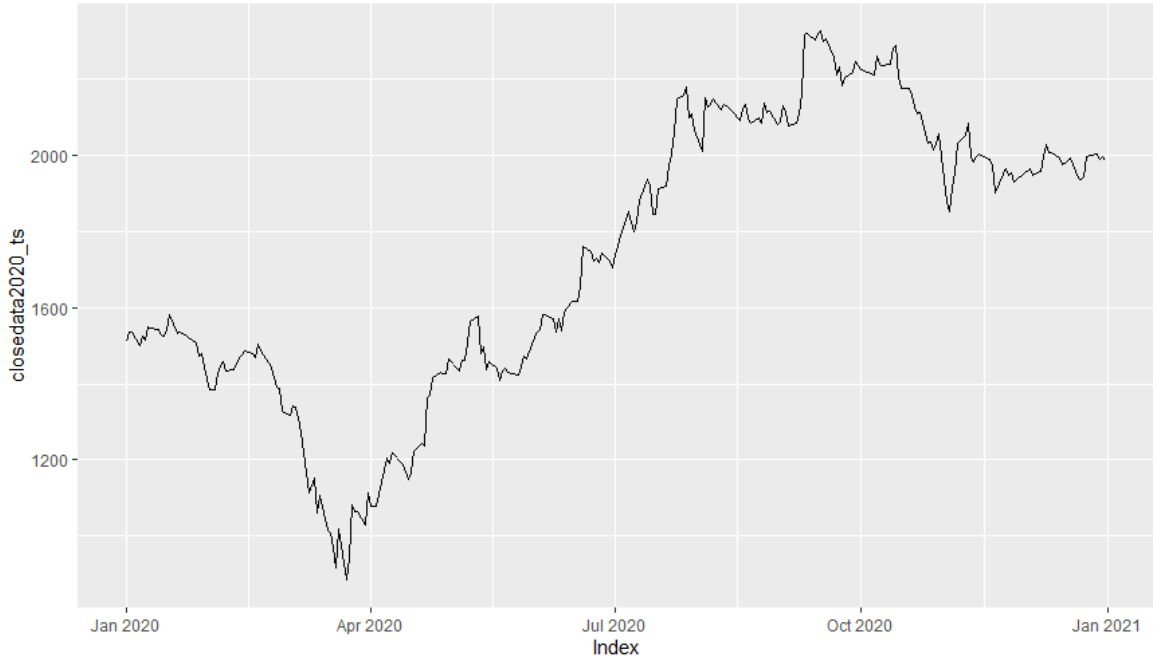
```
> closedata2020_ts=xts(closedata2020[,-1],order.by = closedata2020[,1])
```

Now, the data named “closedata2020_ts” is a time series data with closing price as an only variable in the dataset.

Test For Stationarity

An important assumption in some time series techniques is stationarity. The property of a stationary process is that the auto-correlation ,variance and mean structure do not change over time. So, we will plot the time series to observe the stationarity using graphical method.

```
> autoplot.zoo(closedata2020_ts)
```



We can detect in the plot that there are many leaps and troughs. Also, the data doesn't have constant mean and variance. So,our dataset is not stationary. But to be more sure, we will conduct Augmented Dickey-Fuller Test.

Augmented Dickey-Fuller Test: Augmented Dickey-Fuller test is used to check absence of serial correlation,up to a specified lag k. Now, we will set up the hypothesis-

H_o: Time-series data is non-stationary.

H_1: Time-series data is stationary.

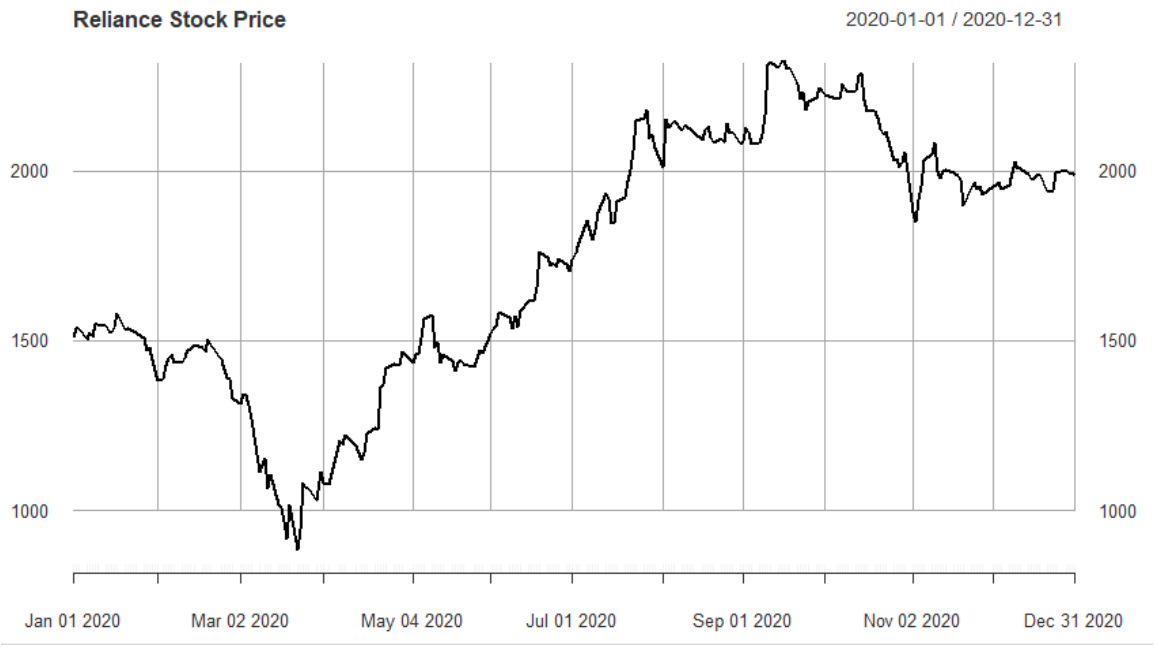
```
> print(adf.test(closedata2020_ts,k=10))

        Augmented Dickey-Fuller Test

data:  closedata2020_ts
Dickey-Fuller = -1.9141, Lag order = 10, p-value = 0.612
alternative hypothesis: stationary
```

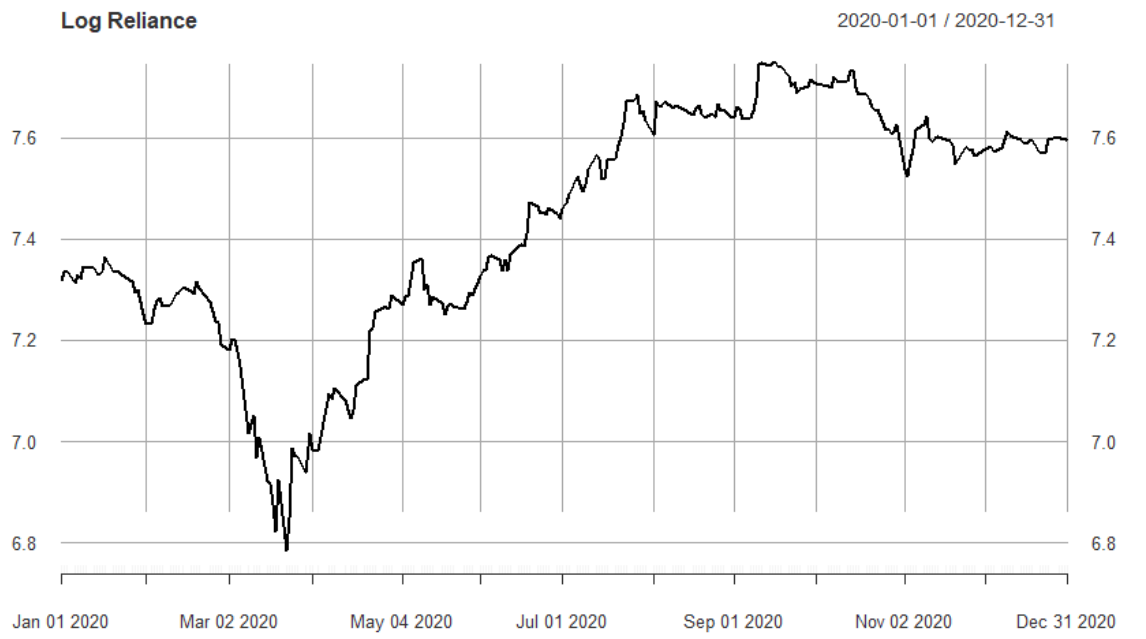
Presently, by applying the adf.test work, we acquired a dickey-fuller esteem = - 1.9141 at slack order=10. Also,as the p-value=0.612 is more than the 0.05 degree of importance, we acknowledge the invalid speculation. In this way, we can close both from the chart and the adf test that the time arrangement information is non-fixed. Presently, first we need to stationarize the time arrangement information.

Stationarize the data

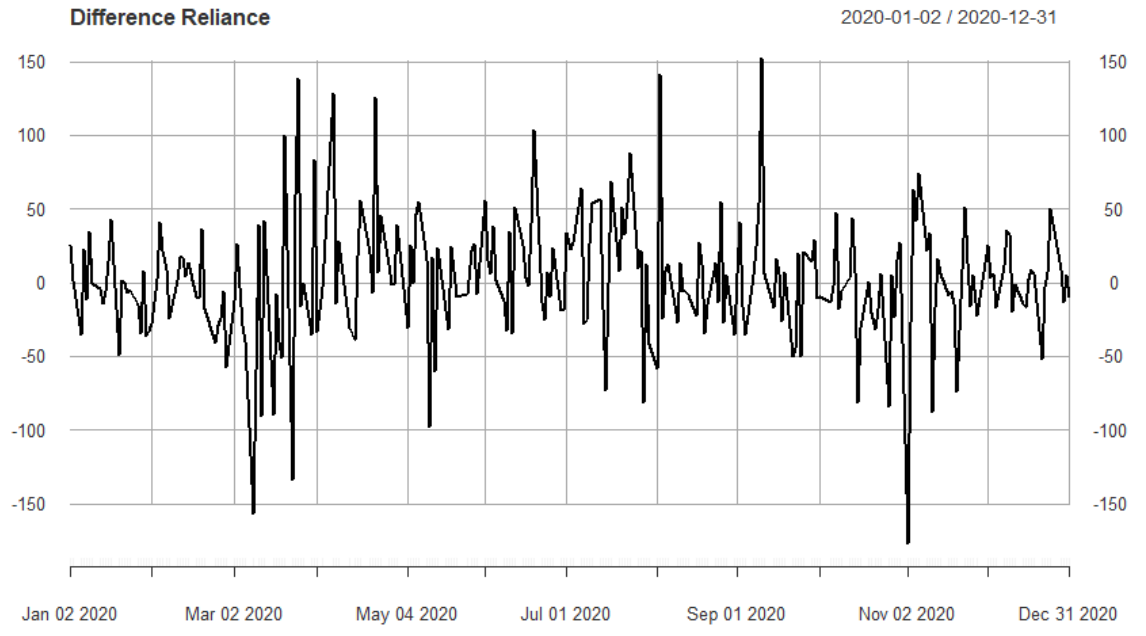
Differencing is a common solution used to stationarize the variable. We will perform differencing using R function diff. I have also used the logarithmic data to stabilize the series. Then I differenced the logarithmic data. Then I compared all the possibilities using graphical representations.

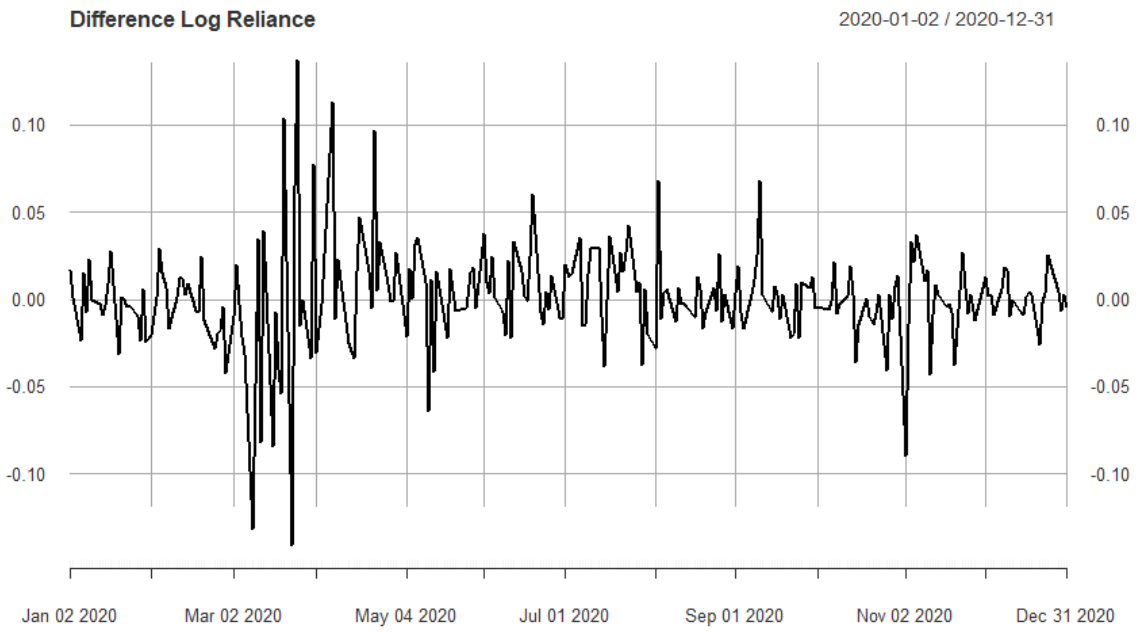
```
> plot(closedata2020_ts,type='l',main='Reliance Stock Price')
> diff_rel=diff(closedata2020_ts)
> diff_rel=diff_rel[-1,]
> plot(diff_rel,type='l',main='Difference Reliance')
> log_rel=log(closedata2020_ts)
> plot(log_rel,type='l',main='Log Reliance')
> difflog_rel=diff(log_rel)
> difflog_rel=difflog_rel[-1,]
> plot(difflog_rel,type='l',main='Difference Log Reliance')
```





We can see here, the logarithmic data seems to be more stable then the original data. So, we will be using logarithmic data to construct the ARIMA model.





We can see here, the differenced log reliance data has lesser variance than the differenced reliance data. Less variance implies more stationarity in the data. Also, we will utilise differenced log data to construct the ARIMA model. But first, we need to test again for the stationarity of the differenced log time series data.

We will again set up the hypothesis-

H_o: Time-series data is non-stationary.

H_1: Time-series data is stationary.

```
> print(adf.test(difflog_rel,k=10))

        Augmented Dickey-Fuller Test

data:  difflog_rel
Dickey-Fuller = -3.9856, Lag order = 10, p-value = 0.01022
alternative hypothesis: stationary
```

Presently, in the wake of applying the adf.test work, we acquired a dickey-fulley esteem = - 3.9856 at slack order=10. Also,as the p-value=0.01022 is not exactly the 0.05 degree of importance, we reject the invalid speculation. In this way, we can finish up both from the chart and the adf test that the differenced log time arrangement information is fixed. Henceforth, we would now be able to move to the model assessment step.

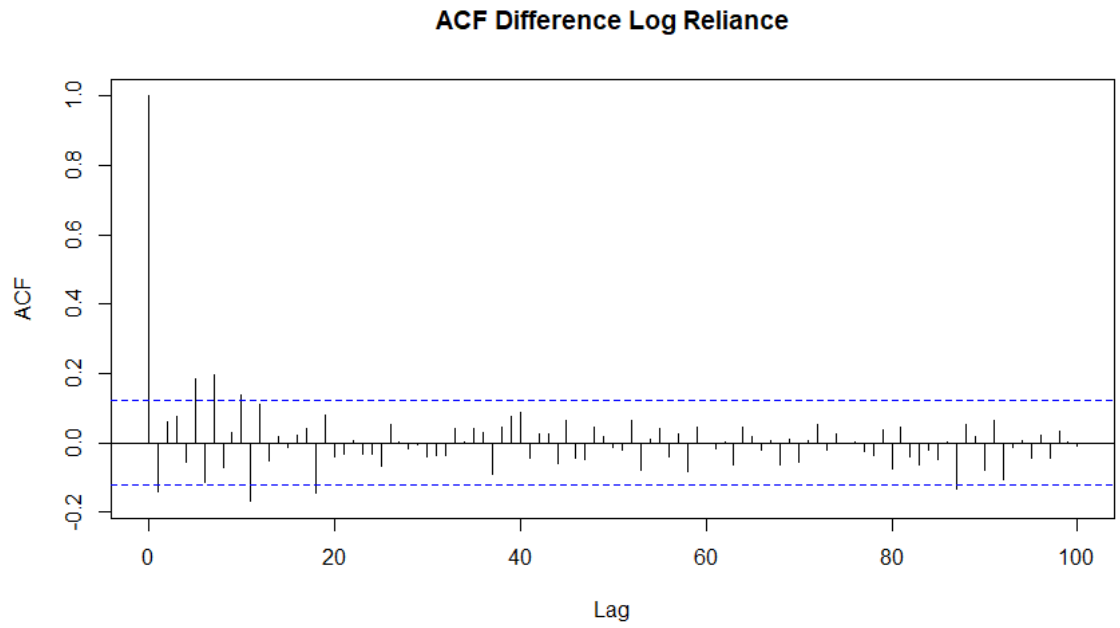
ARIMA Modelling

1. **Autocorrelation plots:** Autocorrelation plots (also known as ACF or the auto correlation function) can help to get the order parameters for ARIMA model. It helps to see the correlation between points, up to and including the lag unit. So, in short, it will let us know how the given time series is correlated with self. In ACF, the correlation coefficient is in y-axis where as the number of lags is shown in the x-axis. Now, the rule is that:
   - If there is a positive autocorrelation at lag 1 then we use the AR model.
   - If there is a negative autocorrelation at lag 1 then we use the MA model.

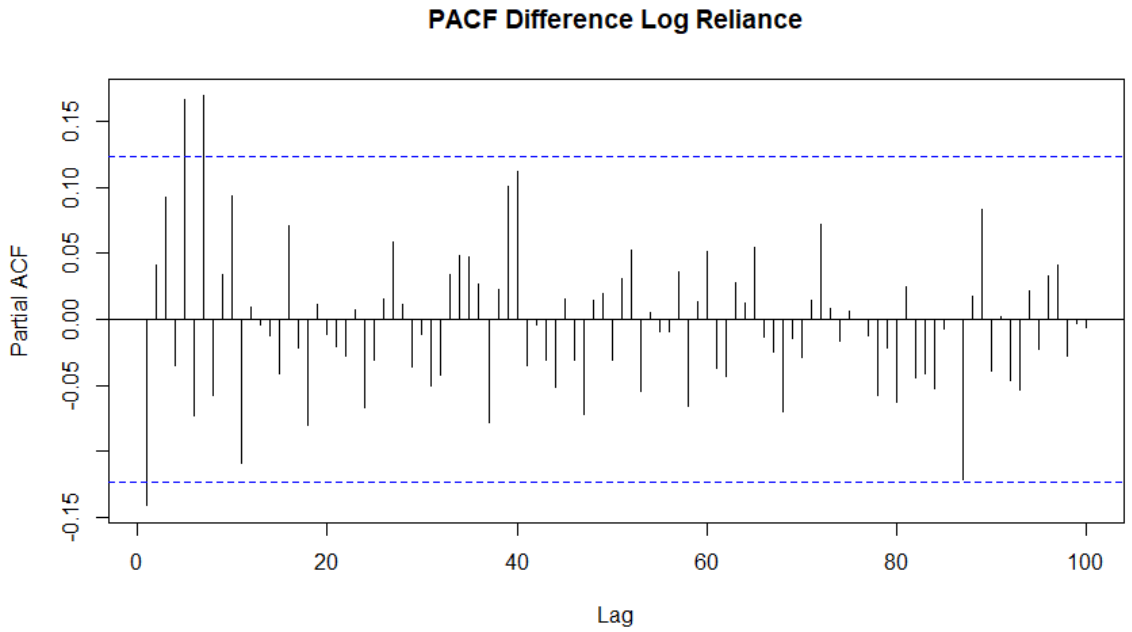
   Partial autocorrelation plots (PACF) displays connection between's a variable and its slacks that isn't clarified by past slacks so ,in short, the pacf at lag k is the correlation that results after removing the effect of any correlations due to the terms at shorter lags. Now, here the rule is that:
   - If the PACF plot drops off at lag n, then use an AR(n) model.
   - If the drop in PACF is more gradual then we use the MA term.

   R plots 95% significance boundaries as blue dotted lines. Now, using the acf and pacf function, we will plot the graphs.

```
> acf_rel=acf(difflog_rel,main='ACF Difference Log Reliance',lag.max=100)
> pacf_rel=pacf(difflog_rel,main='PACF Difference Log Reliance',lag.max=100)
```



As at lag=1, we can observe that there is a negative correlation, so there is a MA term in the model. Also, we can spot at lag 0, the acf=1 because the value has correlation with itself only.



As the PACF plot doesn't show a gradual decrease in the pacf values with respect to the number of lags, we can say that an AR term is presnt in the model.

2. Fitting of the model:

Now, using the auto.arima() function, we will find ideal model for the given data. This function automatically generates a set of optimal (p,d,q). This function tries all the combinations of order parameters and gives us the optimal combination.

Now, we will fit 2 models for the dataset, first with the differencing and another without differencing-

```
> modelfit <- auto.arima(difflog_rel, lambda = "auto")
> modelfit
Series: difflog_rel
ARIMA(2,0,2) with non-zero mean
Box Cox transformation: lambda= 0.8223047

Coefficients:
          ar1      ar2     ma1     ma2     mean
      -1.6650  -0.8974  1.5912  0.7855  -1.2138
s.e.   0.0624   0.0604  0.0890  0.0862   0.0034

sigma^2 estimated as 0.003314:  log likelihood=362.73
AIC=-713.47   AICc=-713.12   BIC=-692.31
```

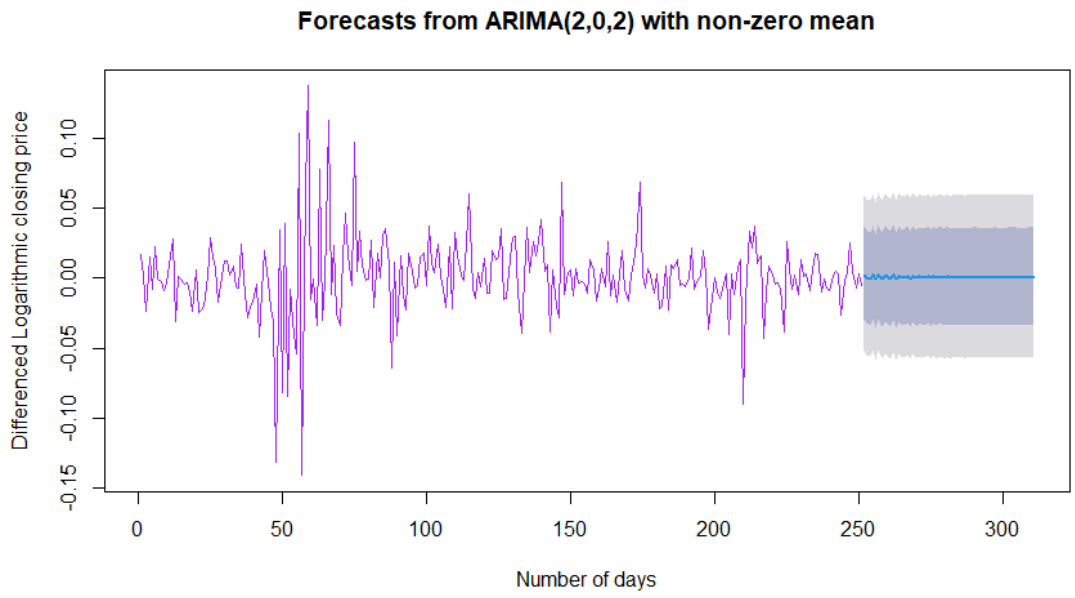
```
> modelfit2 <- auto.arima(log_rel, lambda = "auto")
> modelfit2
Series: log_rel
ARIMA(2,1,2) with drift
Box Cox transformation: lambda= 1.999924

Coefficients:
          ar1      ar2     ma1     ma2   drift
      -1.6668  -0.8938  1.5913  0.7786  0.0081
s.e.   0.0626   0.0611  0.0888  0.0866  0.0121

sigma^2 estimated as 0.04185:  log likelihood=44.46
AIC=-76.93   AICc=-76.58   BIC=-55.78
```

So,one can spot that the difflog_rel data results in arima(2,0,2) model while the log_rel data results in arima(2,1,2) model. So, as in difflog_rel data, the data was already differenced once, we can conclude that the optimal arima model would be ARIMA(2,1,2). Also, one can spot that the AIC and BIC value is less in modelfit than in modelfit2 because modelfit used differenced logarithmic closing price as a variable. Also, sigma^2 is less in modelfit than in modelfit2 which indicates lesser variablity in the model.

3. Forecasting:

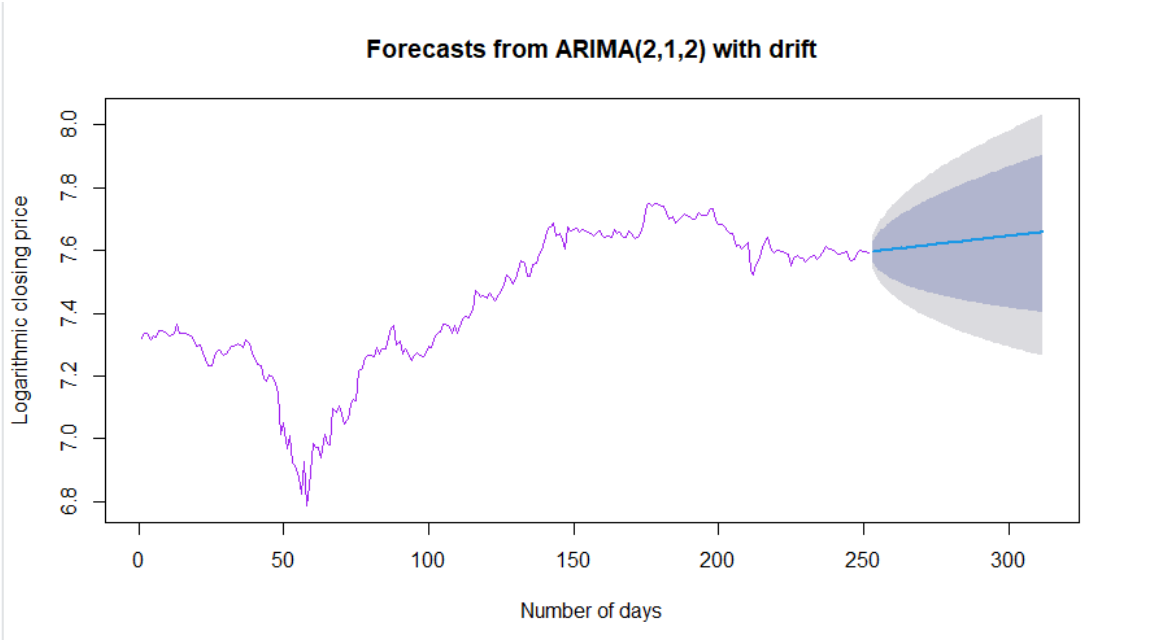
Now,lets forecast the differenced logarithmic closing price for the next 60 days using forecast and plot function.

```
> k=forecast(modelfit,h=60)
> plot(k,col = "purple",ylab="Differenced Logarithmic closing price",xlab = "Number of days")
```



From this graph, we can observe that there is not much difference in the logarithmic closing price from the previous logarithmic closing price. So, the closing price in the next 60 days will be close to each other. The dark grey region is the range in which most of the differenced logarithmic closing price will lie. And the differenced logarithmic closing price will not get out of the light grey interval. Now, we will forecast the logarithmic closing price for the next 60 days using forecast and plot function.

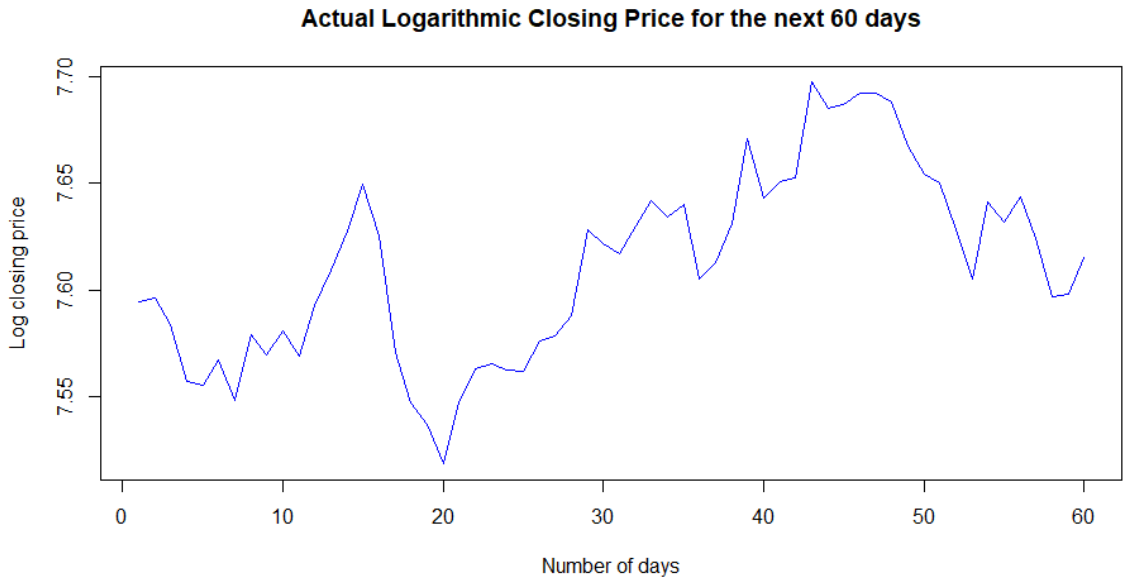
```
> m=forecast(modelfit2,h=60)
> plot(m,col="purple",ylab = "Logarithmic closing price",xlab = "Number of days")
```



From the above plot, one can observe a increasing trend here which tells us that the shutting price will start increasing in the next 60 days. The dark grey region is the range in which most of the logarithmic closing price will lie. We can detect that the dark grey region tells us that the logarithmic closing price will lie in the range on 7.4-7.9. And the logarithmic closing price will not get out of the light grey interval i.e. 7.3-8.0.

Now, lets compare it with the actual closing price for next 60 days. First, we will import the data and name it as TestData. Then we format the Date column as date. We will also take the log of closing price as it will be better for comparing it with the forecasted graph. Then we will plot the graph.

```
> TestData=read.csv("C:/Akshit/sem6/ntcc/TEST_DATA.csv")
> TestData$Date=as.Date(TestData$Date,format="%d-%b-%Y")
> LogTestData=log(TestData$Close.Price)
> plot(LogTestData,type="l",col="blue",main="Actual Logarithmic Closing Price for the next
 60 days",xlab = "Number of days",ylab = "Log closing price")
```

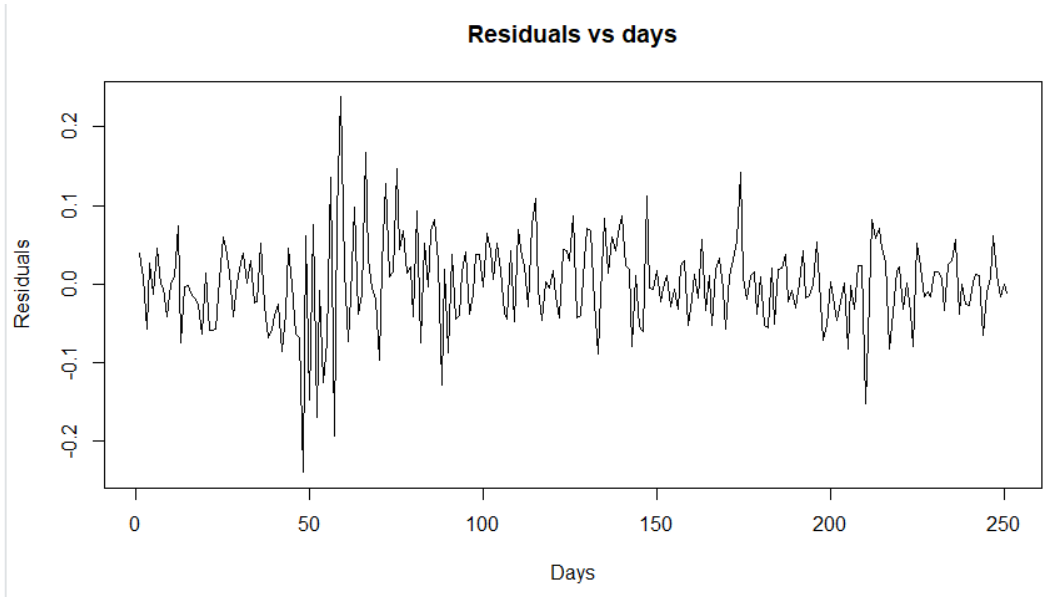


We can spot here, the trend of this plot is increasing as forecasted by the arima(2,1,2) model.

4. Model adequacy analysis:

Now, we will make a time plot of residuals to check the variability.

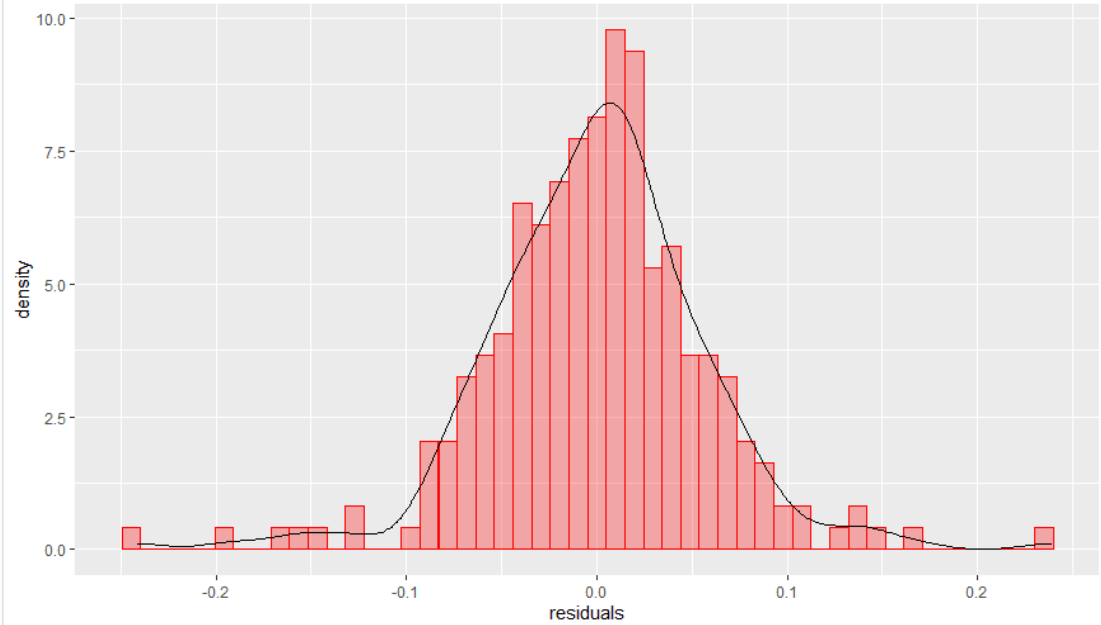
```
> residuals = modelfit$residuals
> plot.ts(residuals,main="Residuals vs days",xlab="Days",ylab="Residuals")
```



We can see here, the residuals have constant variance.

Now, we will check the distribution of the residuals.

```
> ggplot(data.frame(residuals), aes(residuals)) +
+ geom_histogram(bins = 50, aes(y = ..density..), col = "red", fill = "red", alpha = 0.3) +
+   geom_density()
```



The residuals seems to follow normal distribution with mean zero and constant variance, the ARIMA model does seem to provide an adequate predictive model.

Conclusion

Therefore, we observed some useful understandings from the time series analysis. The optimal arima model for the time series data is arima(2,1,2). We also observed that the arima(2,1,2) model gave us an idea that the closing price of reliance shares will increase from Jan,2021 to March,2021. Also, the ARIMA model does seem to provide an adequate predictive model.

References

https://www1.nseindia.com/products/content/equities/equities/eq_security.htm

https://groww.in/blog/how-does-stock-market-work-in-india/

https://www.kaggle.com

https://www.analyticsvidhya.com

https://towardsdatascience.com/time-series-forecasting-arima-models-7f221e9eee06