BITCOIN VALUE PREDICTION

TIME SERIES ANALYSIS

Tarun.S.Sarode

CONTENTS

LIST OF FIGURES

ABSTRACT

Our aim of the project was to predict the Bitcoin value for the next ten days based on the past time series data.

Bitcoin is mined or created, by people (miners) getting their computers to solve mathematical problems, in order to update and verify the ledger.

The given data set was transformed into time series and the intervention point was noted. Splitting the dataset into to two from the intervention point as the current data would provide better prediction in financial time series was the best solution.

GARCH model was applied and the parameters was used to fit the model. The prediction has been made for the next 10 days based on this model.

Further the fitted values were compared with the real values to check the accuracy of the model predicted.

INTRODUCTION

Any form of currency that only exists digitally relying on cryptography to prevent counterfeiting and fraudulent transactions is defined as cryptocurrency. Bitcoin was the very first Cryptocurrency. It was invented in 2009 by an anonymous person, or group of people, who referred to themselves as Satoshi Nakamoto. When someone sends a bitcoin (or a fraction of a bitcoin) to someone else, “miners” record that transaction in a block and add the transaction to a digital ledger. These blocks are collectively known as the blockchain – an openly accessible ledger of every transaction ever made in bitcoin. Blockchains are distributed across many computers so that the record of transactions cannot be altered. Only 21 million bitcoins can ever be mined and about 17 million have been mined so far.

The value of bitcoin is determined by what people are willing to pay for it, and is very volatile, fluctuating wildly from day to day. In April 2013, the value of 1 bitcoin (BTC) was around $100 USD. At the beginning of 2017 its value was $1,022 USD and by the 15th of December, it was worth $19,497. As of the 3rd of March 2018, 1 BTC was sold for $11,513 USD. But now it came back to the window of $3800 USD - $4000 USD. Which shows fluctuations in the prices of the data set and we are using this data to predict the future values.

METHODOLOGY

The dataset focused on has been gathered from **coinmarketcap.com** includes the daily closing price of bitcoin from the 27th of April 2013 to the 24th of February 2019.

The Analysis was done using statistical tools and time series techniques with the use of R language.

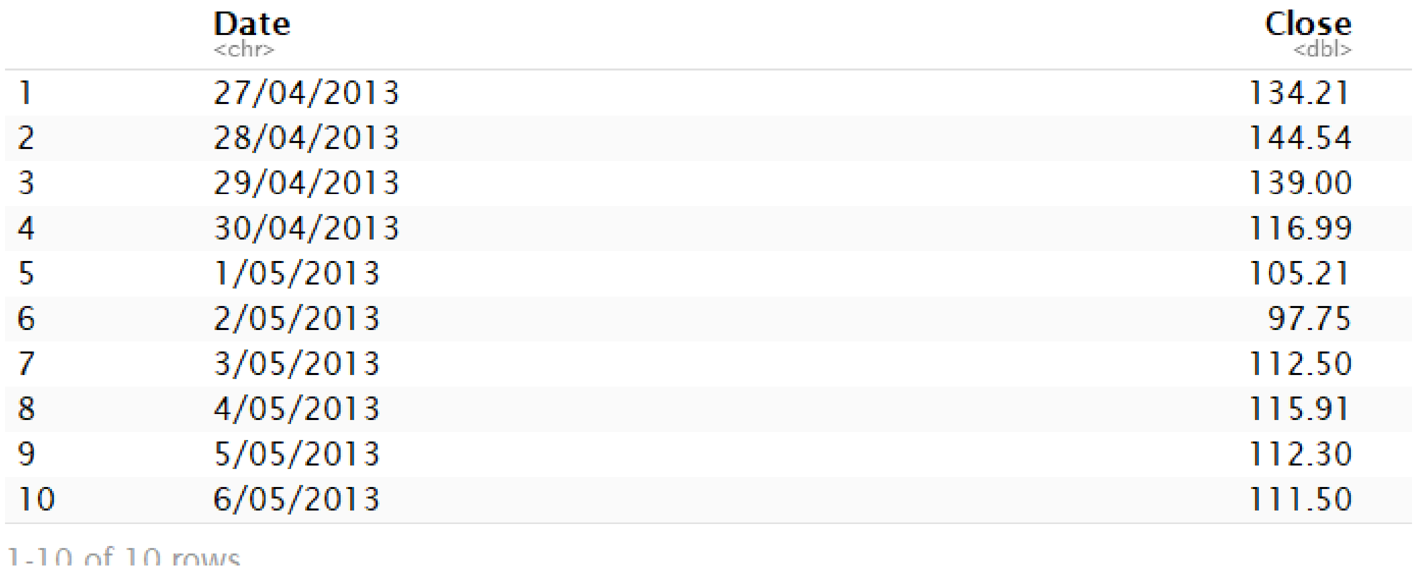
DATA PREPARATION AND EXPLORATION

The Data set had date and value columns with 2,130 observations.

(See appendix () for R chunk.)

```
'data.frame':	2130 obs. of  2 variables:
 $ Date : chr  "27/04/2013" "28/04/2013" "29/04/2013" "30/04/2013" ...
 $ Close: num  134 145 139 117 105 ...
[1] 2130
```

View of the Series.



	Date <chr>	Close <dbl>
1	27/04/2013	134.21
2	28/04/2013	144.54
3	29/04/2013	139.00
4	30/04/2013	116.99
5	1/05/2013	105.21
6	2/05/2013	97.75
7	3/05/2013	112.50
8	4/05/2013	115.91
9	5/05/2013	112.30
10	6/05/2013	111.50

1-10 of 10 rows

The graphical visual of the time series data is as follows:

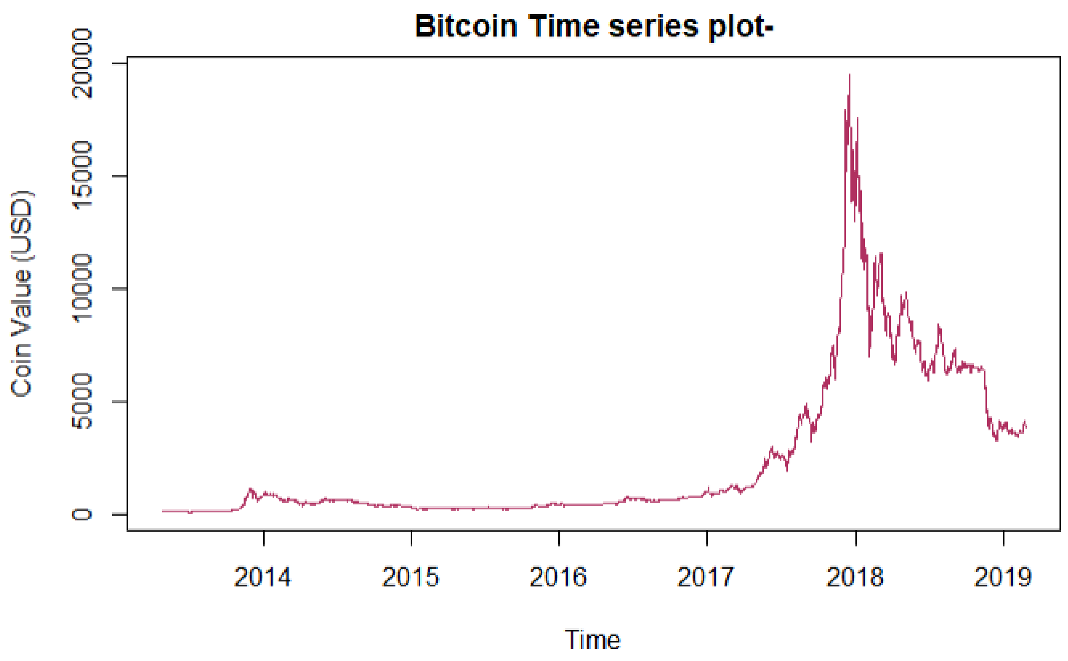


Figure 1 Plot of Time series data

From the plot we can see that there are trends in the series where from the year 2013 to year 2017 there is a steady trend and an increasing trend from 2017 to 2018 which shows higher variability and decreasing trend in 2018 to 2019 which shows high fluctuations in just two years. We can see that there is intervention point in the year 2017 and 2018.There is a moving average trend as well as Auto regressive behavior in the series. There is no visible seasonality in the plot. The following visualization shows us the correlation

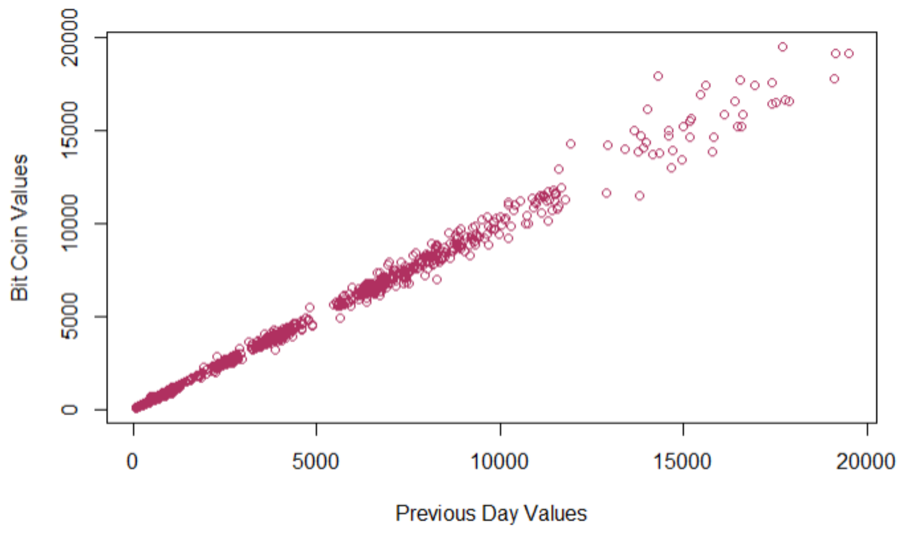


Figure 2 Plot of Coin values Vs Previous values

From the figure 2 we can say that there is a high correlation between the preceding values as well the value is 0.99 which also confirms this.

MODEL SELECTION

First, we are checking if the series is nonstationary by using Dickey-Fuller Test (Unit root test) where the P value should be less than 0.05.

```
#test for stationarity
adfTest (BitcoinDataTS)

##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 1
##   STATISTIC:
##     Dickey-Fuller: -1.2523
##   P VALUE:
##     0.2172
##
## Description:
##  Wed May 29 15:18:36 2019 by user: clarine,shamini,tarun

#adf.test (BitcoinDataTS)
#Indication of non sationarity
```

The result above shows it is not significant as the p-value is greater than 0.05 by which we can consider that the series is non-stationary, and we cannot reject the null hypothesis stating that the series is non-stationary.

We will further decompose the series to have a clear view of the series which would plot the components separately by which we can see if there is non-repeating cycles mixed in with the repeating seasonality components, there may be increasing trend, additive and multiplicative components as well.

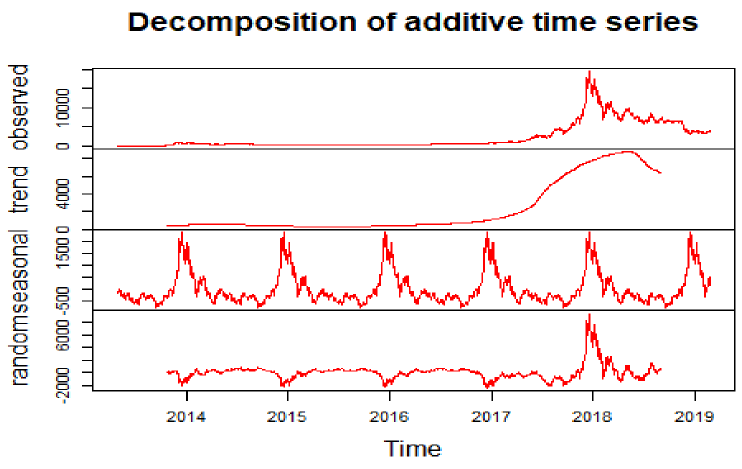


Figure 3 Plot of decomposition of time series

The above plot shows us the following where the observed plot is the basic plot, the next plot shows the trend in the series, followed by the seasonality of the series and the last random shows the residual of the trend fitted above. From which we can consider that the deterministic models cannot be applied to the bitcoin series.

We assumed that the series is partly autoregressive and partly moving average, we tried to obtain a quite general ARMA(p, q) time series model.

Hence, we applied log transformation and differencing to the data set which gives the following plot from which we can see that there is no trend and the series is stationary.

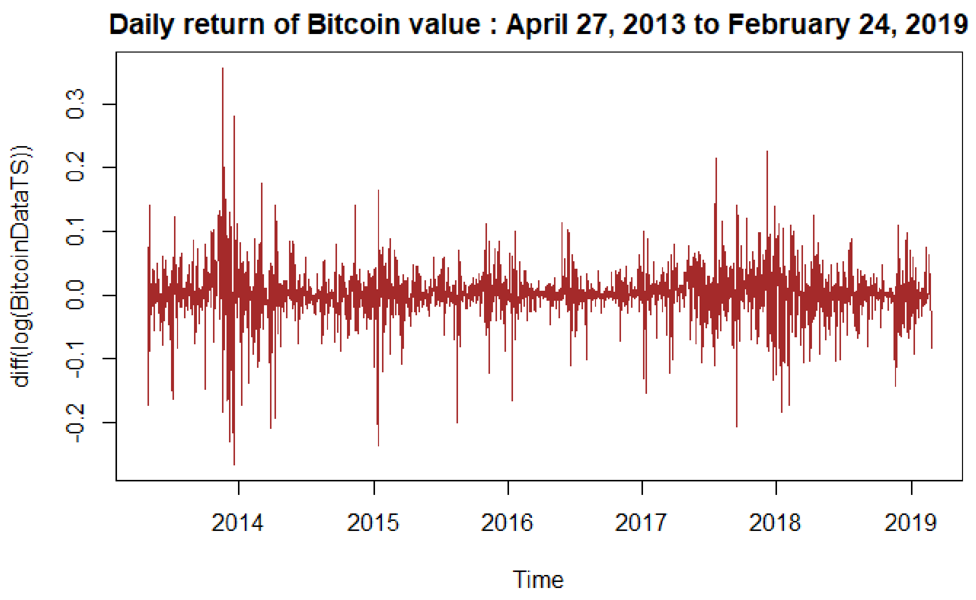


Figure 4 Plot of Transformed and differenced series

This is also crossed checked by applying the Dickey fuller test as follows.

```
p-value smaller than printed p-value
Title:
 Augmented Dickey-Fuller Test

Test Results:
  PARAMETER:
    Lag Order: 1
  STATISTIC:
    Dickey-Fuller: -33.0075
  P VALUE:
    0.01
```

The result shows the p value less than 0.05 which is significant for stationarity.

We examined the differenced series and fitted ARIMA model for the series using the auto. arima() function.

```
Series: diff(log(BitcoinDataTS))
ARIMA(0,0,0) with non-zero mean

Coefficients:
        mean
      0.0016
s.e.  0.0009

sigma^2 estimated as 0.001894:  log likelihood=3653.2
AIC=-7302.4   AICc=-7302.39   BIC=-7291.07
```

From the output we can conclude that ARIMA is not a best model for the series as the (p, d,q) values are ‘0’.

SPLITTING OF THE SERIES

To understand the series better we are splitting the data into two parts from 2013 to 2017 and 2017 to 2019 and carry on the analysis of the series.

In this financial time series data, series consists of a high variation between the current and past

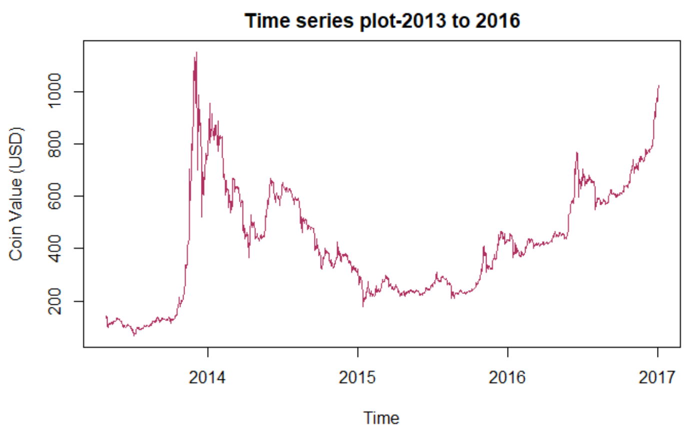


Figure 5 Plot of time series for split 1

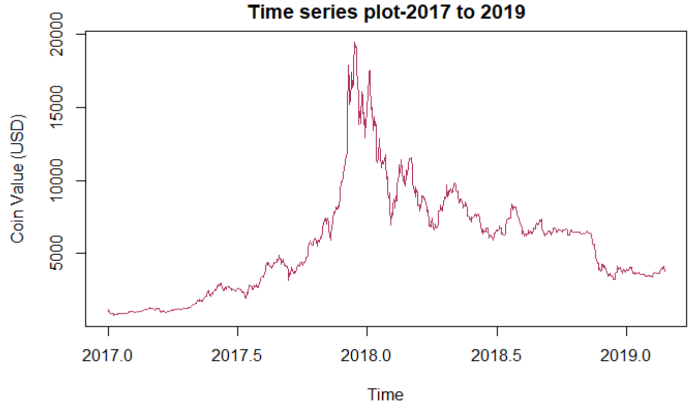
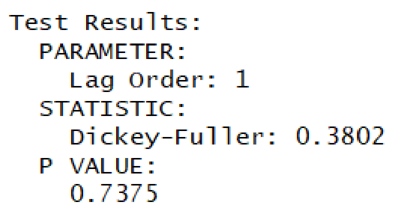
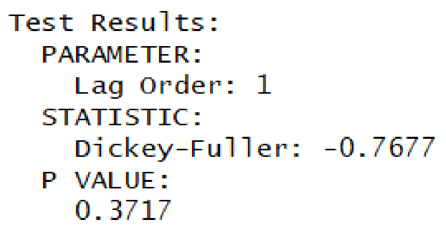


Figure 6 Plot of time series for split 2

The dickey fuller test is applied on both sets again to check the non-stationarity of the split series which showed that the series was nonstationary, and this has been proved by the results below. As the p values are greater than 0.05.



```
Test Results:
  PARAMETER:
    Lag Order: 1
  STATISTIC:
    Dickey-Fuller: 0.3802
  P VALUE:
    0.7375
```



```
Test Results:
  PARAMETER:
    Lag Order: 1
  STATISTIC:
    Dickey-Fuller: -0.7677
  P VALUE:
    0.3717
```

The ACF and PACF plots were derived to check the autocorrelation

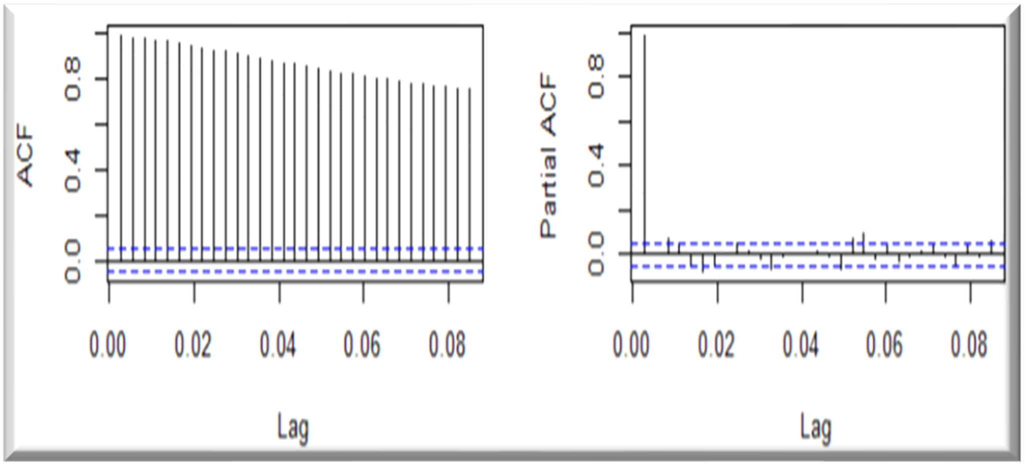


Figure 7 ACF and PACF plots of Time series 1

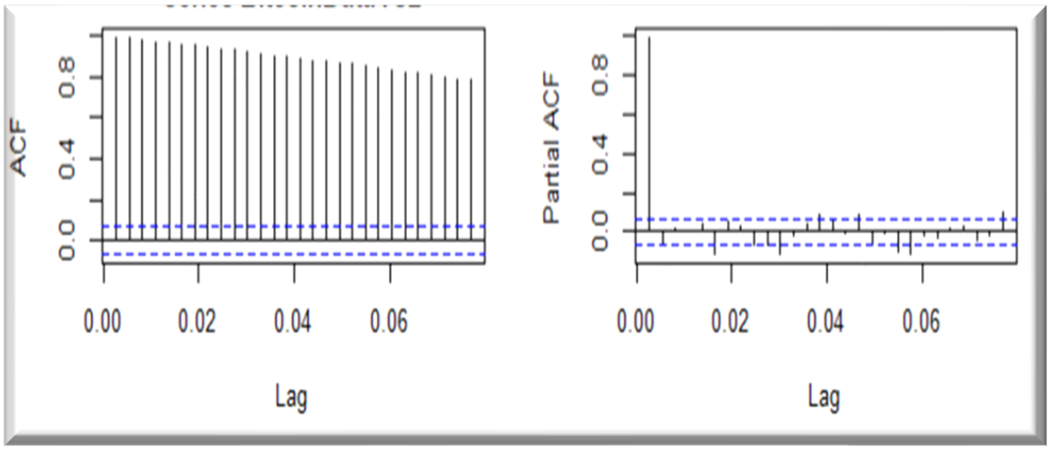


Figure 8 ACF and PACF plots of Time series 2

It has no seasonality as it has no wave pattern and confirms the trend as it has a slowly decaying pattern.

It can be seen that from the ACF and PACF plots of the series that thaer is a slowly decaying pattern in the ACF plot which confirms that the series has trend and from the PACF we can see that there is very high first corelaation which infers that the series is non-stationary.

This problem has been soughed by transforming into log and differencing. Change in values are multiplied by 100 so that they can be interpreted as percentage changes in the price. The multiplication may also reduce numerical errors as the changing values could be very small numbers and render large rounding errors in some calculations.

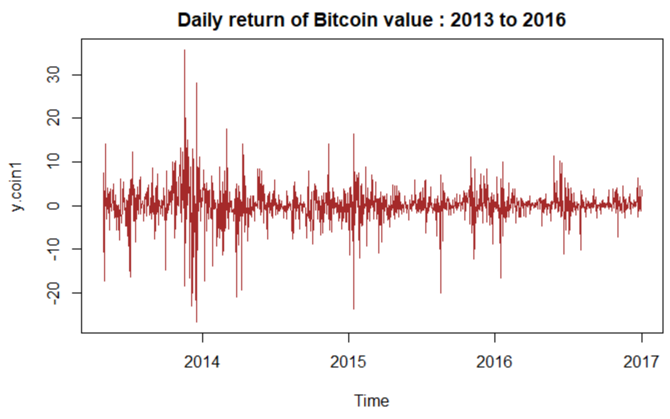


Figure 9 Plot of returns for split data 2

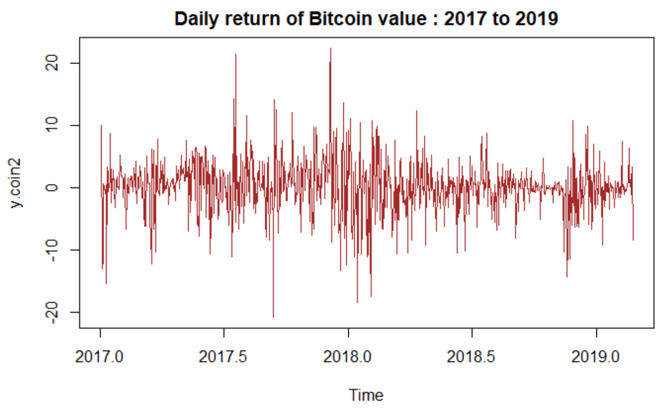
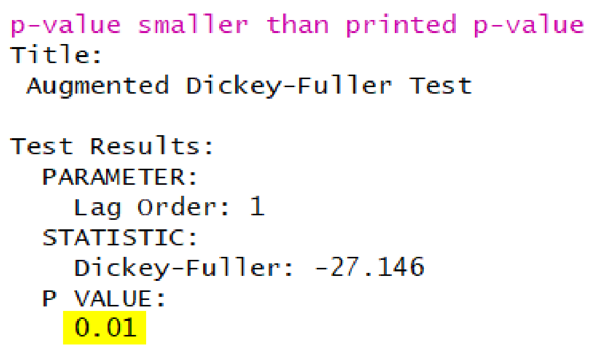


Figure 10 Plot of returns for split data 1

- The time series returns plot appears to be stationary.
- There is sign of neither a trend nor seasonality.
- Observations are bouncing around the mean level.
- Changing variance is obvious.
- Mean is very close to zero

After applying Dickey-Fuller test the result shows the time series is stationary as the test is significant at 5%

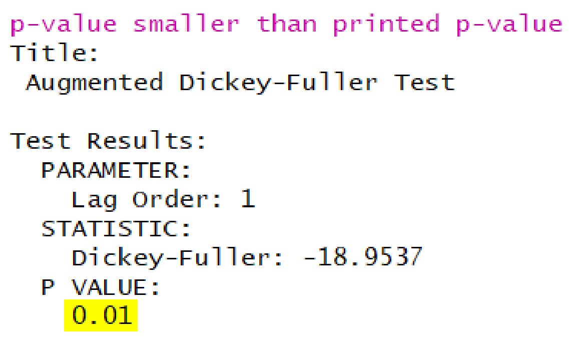
Split-1



```
p-value smaller than printed p-value
Title:
 Augmented Dickey-Fuller Test

Test Results:
  PARAMETER:
    Lag Order: 1
  STATISTIC:
    Dickey-Fuller: -27.146
  P VALUE:
    0.01
```

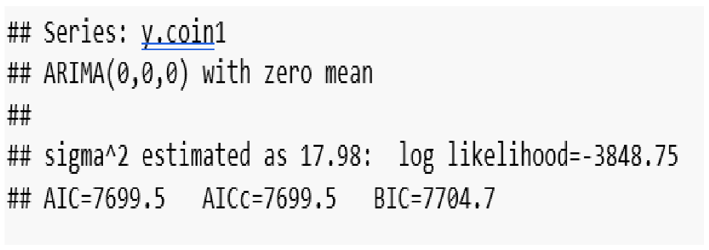
Split-2



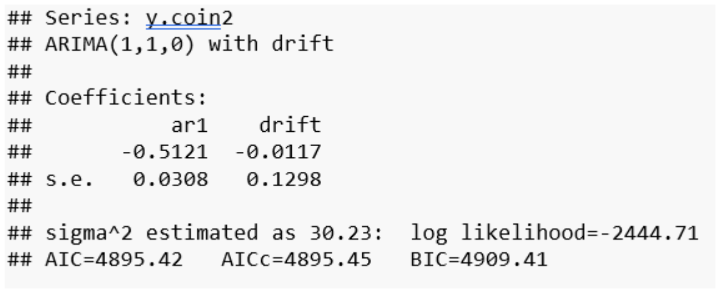
```
p-value smaller than printed p-value
Title:
 Augmented Dickey-Fuller Test

Test Results:
  PARAMETER:
    Lag Order: 1
  STATISTIC:
    Dickey-Fuller: -18.9537
  P VALUE:
    0.01
```

We fitted ARIMA model for the series using the auto. arima() function.

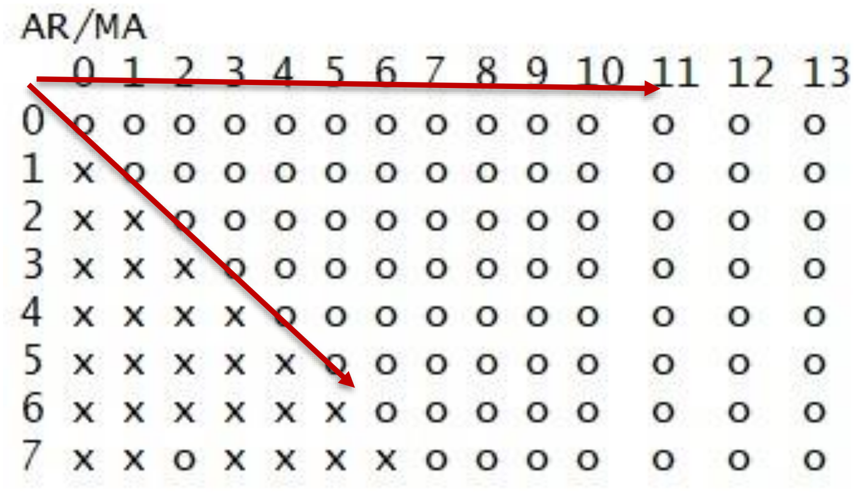


```
## Series: y.coin1
## ARIMA(0,0,0) with zero mean
##
## sigma^2 estimated as 17.98:  log likelihood=-3848.75
## AIC=7699.5   AICc=7699.5   BIC=7704.7
```



```
## Series: y.coin2
## ARIMA(1,1,0) with drift
##
## Coefficients:
##           ar1    drift
##       -0.5121  -0.0117
## s.e.   0.0308   0.1298
##
## sigma^2 estimated as 30.23:  log likelihood=-2444.71
## AIC=4895.42   AICc=4895.45   BIC=4909.41
```

From the output of the split series we can conclude that ARIMA is not a best model for the series – split 1 as the (p,d,q) values are (0,0,0) but the (p,d,q) values are (1,1,0) for the second series. Hence, we consider EACF to make a further inference examined the residuals for the series 2.



```
AR/MA
  0 1 2 3 4 5 6 7 8 9 10 11 12 13
0 o o o o o o o o o o o  o  o  o
1 x o o o o o o o o o o  o  o  o
2 x x o o o o o o o o o  o  o  o
3 x x x o o o o o o o o  o  o  o
4 x x x x o o o o o o o  o  o  o
5 x x x x x o o o o o o  o  o  o
6 x x x x x x o o o o o  o  o  o
7 x x o x x x x o o o o  o  o  o
```

ARMA (0,0) is a clear edge point and it is an indication of white noise and ARMA(1,0) is not a model from this Extended ACF output.

Thus, we concluded that ARIMA cannot be fit to the series 2.

Consequently, we have fitted ARCH and GARCH.

We applied the McLeod-Li test to check if there is presence of ARCH in the series.

McLeod-Li test for ARCH

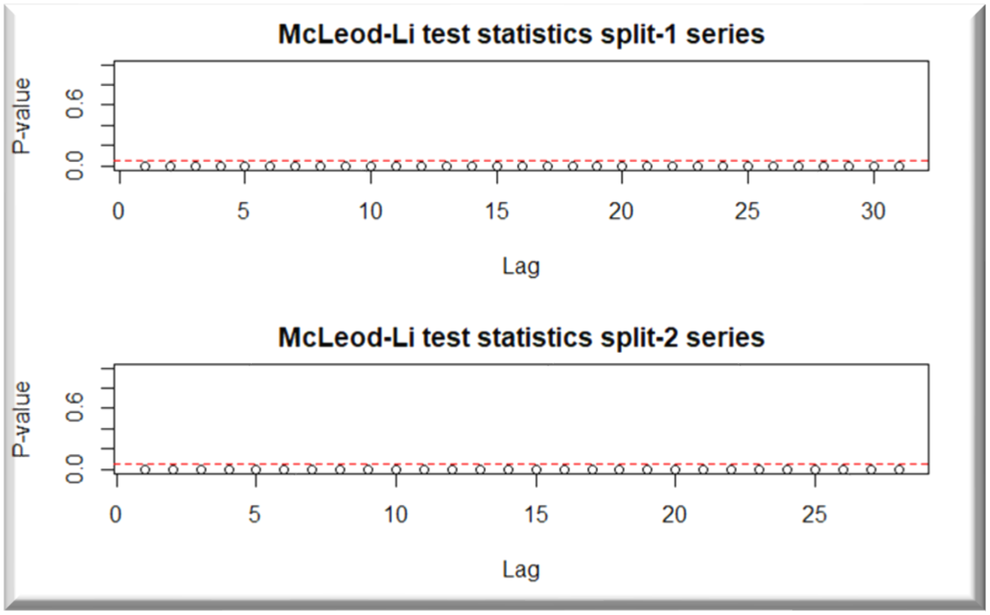


Figure 11 Plots of McLeod-Li test for Split 1 series

McLeod-Li test shows all lags are significant at the 5% significance level. This is a strong evidence for ARCH in this data

Normality test for ARCH

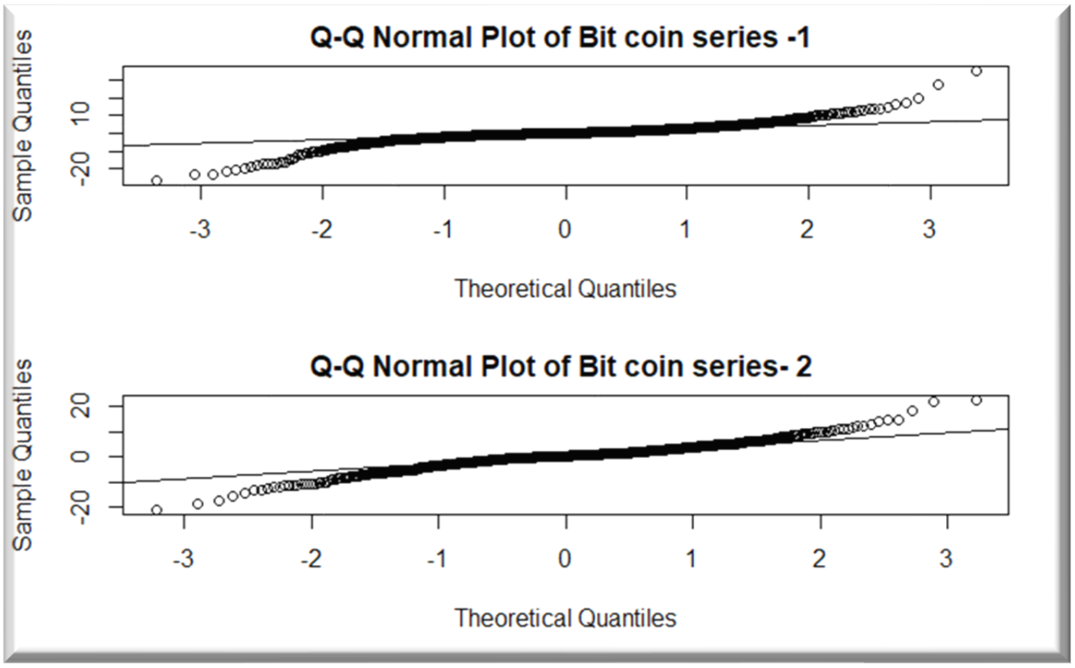
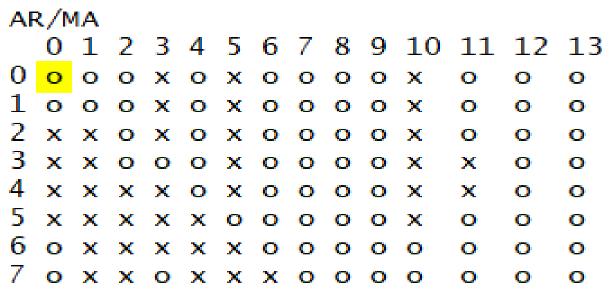


Figure 12 Plots of Q-Q Normal for Split 1 series

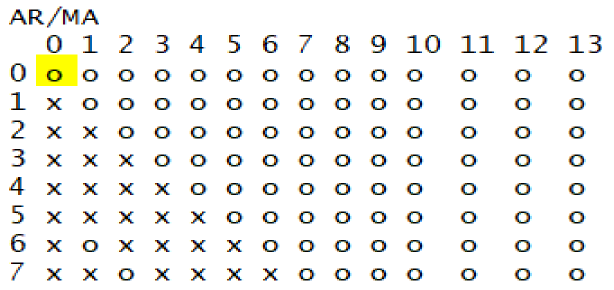
From the above plots we can see that the Q-Q Plot shows a Heavy-tailed distribution, which confirms non-normality in both the series.

EACF plots – for GARCH Model



```
AR/MA
  0 1 2 3 4 5 6 7 8 9 10 11 12 13
0 o o o x o x o o o o x  o  o  o
1 o o o x o x o o o o x  o  o  o
2 x x o x o x o o o o x  o  o  o
3 x x o o o x o o o o x  x  o  o
4 x x x x o x o o o o x  x  o  o
5 x x x x x o o o o o x  o  o  o
6 o x x x x x o o o o o  o  o  o
7 o x x o x x x o o o o  o  o  o
```

Split 1



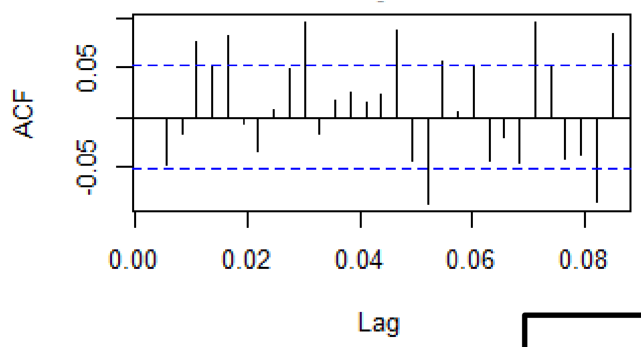
```
AR/MA
  0 1 2 3 4 5 6 7 8 9 10 11 12 13
0 o o o o o o o o o o o  o  o  o
1 x o o o o o o o o o o  o  o  o
2 x x o o o o o o o o o  o  o  o
3 x x x o o o o o o o o  o  o  o
4 x x x x o o o o o o o  o  o  o
5 x x x x x o o o o o o  o  o  o
6 x o x x x x o o o o o  o  o  o
7 x x o x x x x o o o o  o  o  o
```

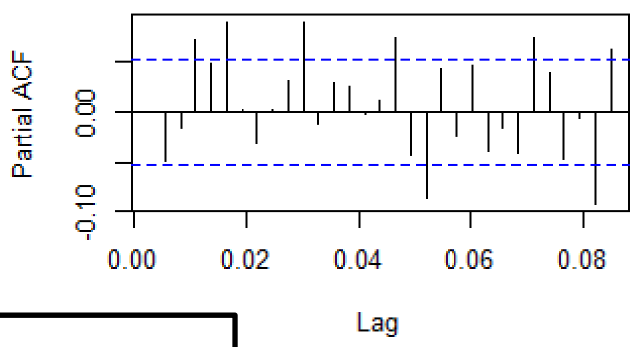
Split 2

The above EACF plots shows (0,0) as an ARMA model and d=1. This is a clear indication of white noise in the series 1 and series 2.

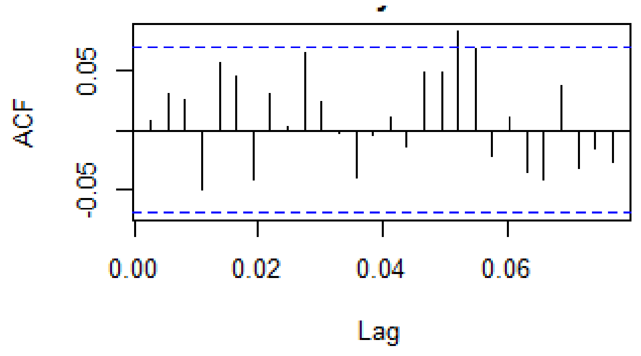
ACF and PACF plots

Time Series Data - Split 1





Time Series Data - Split 2



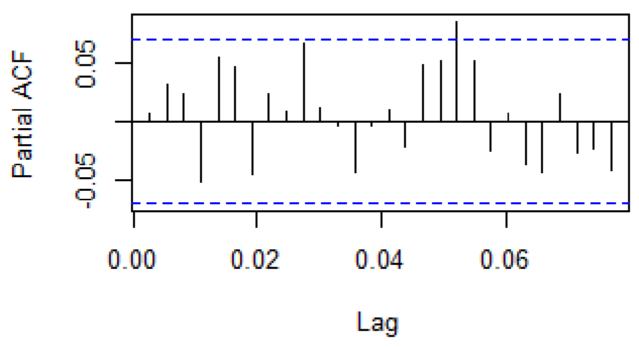


Figure 13 ACF and PACF for both splits

From the ACF and PACF plots of both the series we can observe many correlations in the plots which shows the evidence that our time series are not independent and identically distributed.

Because we observe many significant correlations in these plots, we have some evidence that our time series are not independently and identically distributed.

Due to white noise, Absolute and square root Transformation was done on both splits.

MODEL SPECIFICATION

Split 1

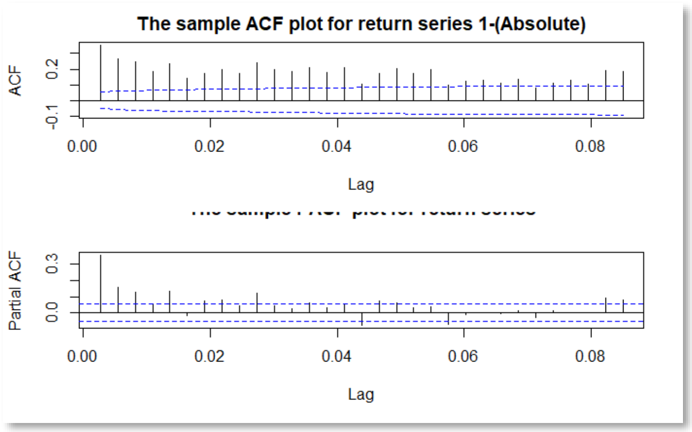


Figure 14 Absolute transformed - Split 1

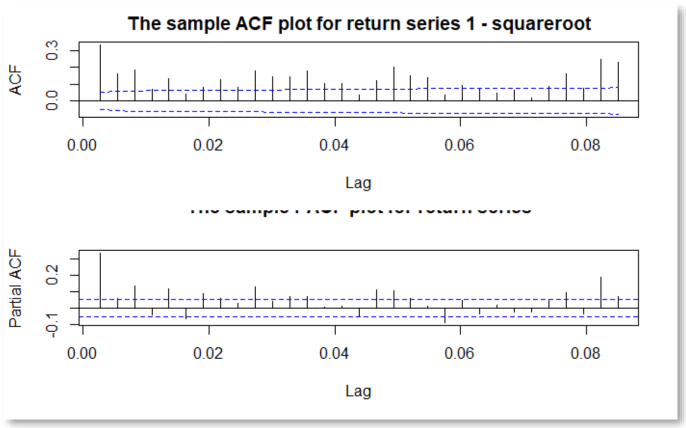
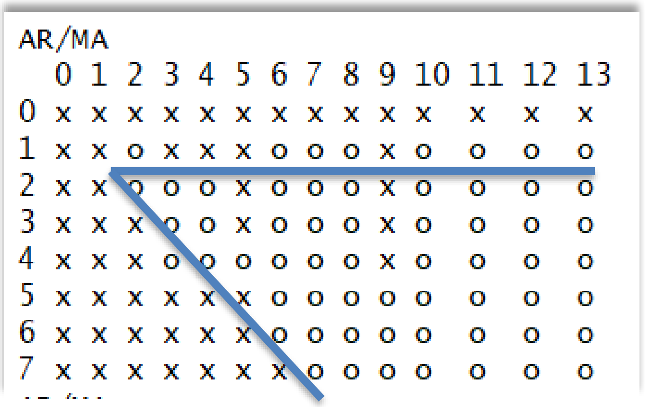


Figure 15 Square root transformed - Split 1

From the plots we can observe significant serial correlations in the transformed series, we infer that the simulated process is serially dependent, and the lag 1 autocorrelation is significance.

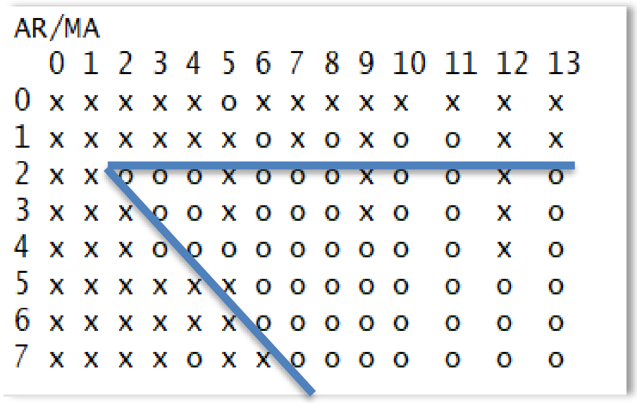
To illustrate the specification of a GARCH model, we obtain the EACF of the absolute and squared values from the simulated series 1.

EACF- Absolute Split 1



```
AR/MA
  0 1 2 3 4 5 6 7 8 9 10 11 12 13
0 x x x x x x x x x x x  x  x  x
1 x x o x x x o o o x o  o  o  o
2 x x o o o x o o o x o  o  o  o
3 x x x o o x o o o x o  o  o  o
4 x x x o o o o o o x o  o  o  o
5 x x x x x x o o o o o  o  o  o
6 x x x x x x o o o o o  o  o  o
7 x x x x x x x o o o o  o  o  o
```

EACF-Square root Split 1



```
AR/MA
  0 1 2 3 4 5 6 7 8 9 10 11 12 13
0 x x x x x o x x x x x  x  x  x
1 x x x x x x o x o x o  o  x  x
2 x x o o o x o o o x o  o  x  o
3 x x x o o x o o o x o  o  x  o
4 x x x o o o o o o o o  o  x  o
5 x x x x x x o o o o o  o  o  o
6 x x x x x x o o o o o  o  o  o
7 x x x x o x x o o o o  o  o  o
```

From the above ACF, PACF and EACF for the split 1 we get the following models

Candidate model: ARMA (2,2), ARMA (2,3), ARMA (3,3)

GARCH(2,2), GARCH(3,2), GARCH(3,3)

Residual Analysis Series 1

To check if the models are a right fit to the series we will do the Residual Analysis**.**

GARCH MODEL (2,2)

GARCH MODEL (3,2)

GARCH MODEL (3,3)

Split 2

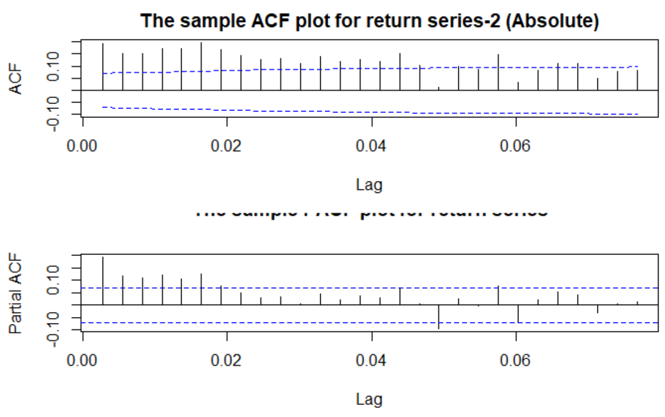


Figure 16 Absolute transformed - Split 2

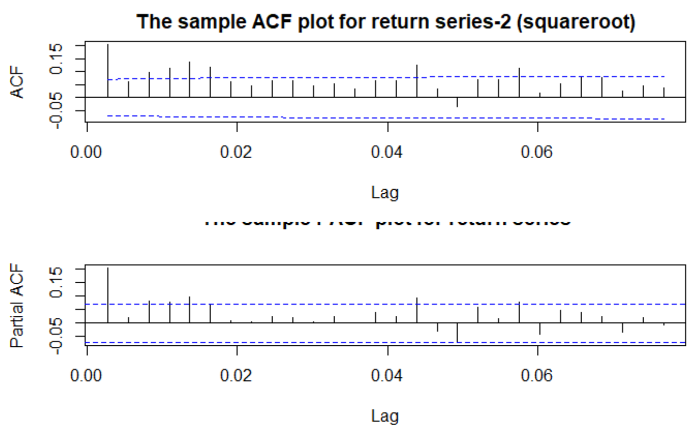
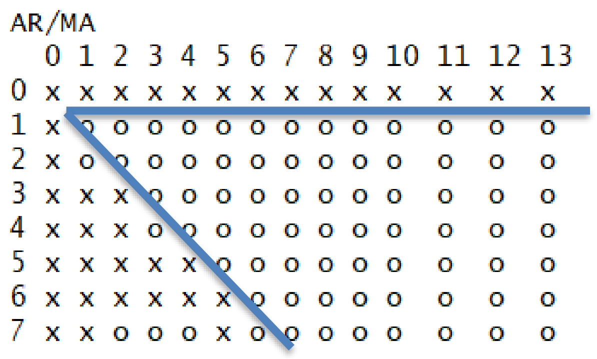


Figure 17 Square root transformed Split 2

From the above plots we can see that there are significant correlations in the transformed series 2

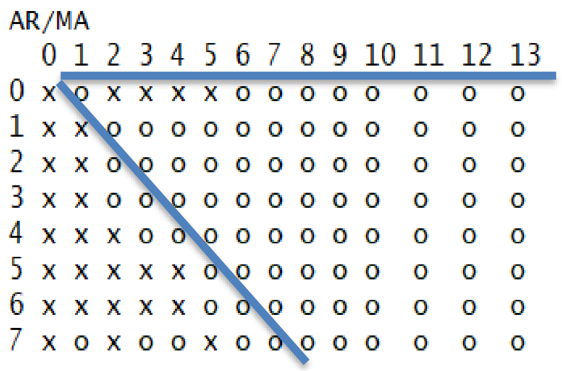
To illustrate the specification of a GARCH model, we obtain the EACF of the absolute and squared values from the simulated series 2.

EACF- Absolute Split 2



AR/MA	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x
1	x	o	o	o	o	o	o	o	o	o	o	o	o	o
2	x	o	o	o	o	o	o	o	o	o	o	o	o	o
3	x	x	x	o	o	o	o	o	o	o	o	o	o	o
4	x	x	x	o	o	o	o	o	o	o	o	o	o	o
5	x	x	x	x	x	o	o	o	o	o	o	o	o	o
6	x	x	x	x	x	x	o	o	o	o	o	o	o	o
7	x	x	o	o	o	x	o	o	o	o	o	o	o	o

EACF-Square root Split 2



AR/MA	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	x	o	x	x	x	x	o	o	o	o	o	o	o	o
1	x	x	o	o	o	o	o	o	o	o	o	o	o	o
2	x	x	o	o	o	o	o	o	o	o	o	o	o	o
3	x	x	o	o	o	o	o	o	o	o	o	o	o	o
4	x	x	x	o	o	o	o	o	o	o	o	o	o	o
5	x	x	x	x	x	o	o	o	o	o	o	o	o	o
6	x	x	x	x	x	o	o	o	o	o	o	o	o	o
7	x	o	x	o	o	x	o	o	o	o	o	o	o	o

From the above ACF, PACF and EACF for the split 2 we get the following models

Candidate model: ARMA (1,1), ARMA (1,2), ARMA (2,2), ARMA (1,3), ARMA (2,3)

GARCH(1,1), GARCH(2,1), GARCH(2,2), GARCH(3,1), GARCH(3,2)

Residual analysis series 2

To check if the models are a right fit to the series we will do the Residual Analysis**.**

GARCH (1,1)

GARCH (2,1)

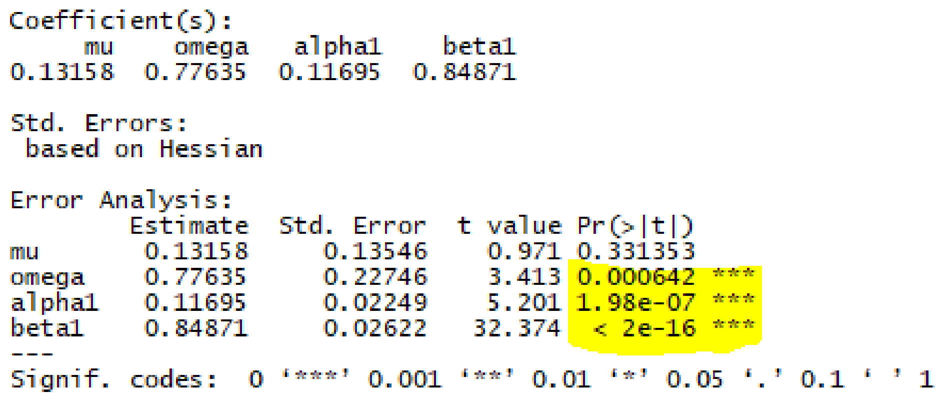
GARCH (2,2)

GARCH (3,1)

GARCH (3,2)

Henceforth from the split 2 time series after applying residual analysis for all the above models the GRACH (1,1) was the best among other models as per the residual analysis and significant tests.

The model diagnostics of GARCH (1,1) is as follows:



```
Coefficient(s):
     mu    omega   alpha1    beta1
0.13158  0.77635  0.11695  0.84871

Std. Errors:
 based on Hessian

Error Analysis:
        Estimate  Std. Error  t value Pr(>|t|)
mu       0.13158     0.13546    0.971 0.331353
omega    0.77635     0.22746    3.413 0.000642 ***
alpha1   0.11695     0.02249    5.201 1.98e-07 ***
beta1    0.84871     0.02622   32.374  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

From the above results we get all the values as significant at 5% level of significance.

Residual Analysis GARCH(1,1)

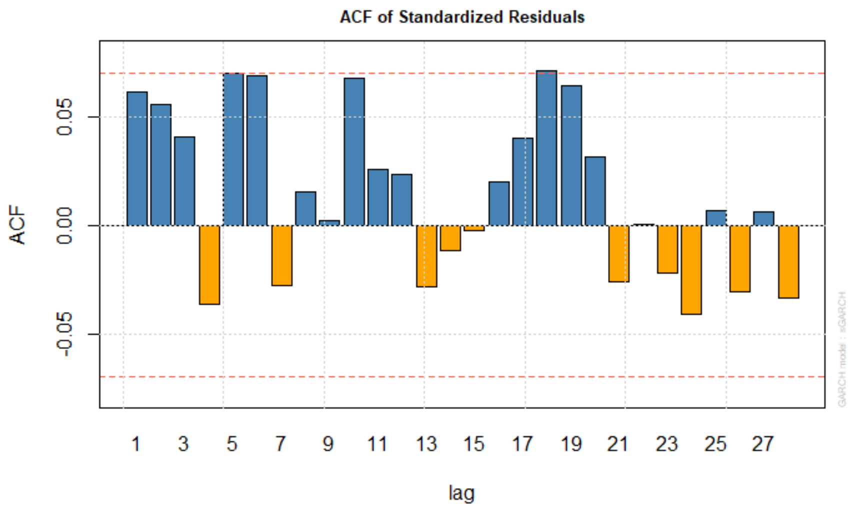


Figure 18 ACF of Standardized Residuals

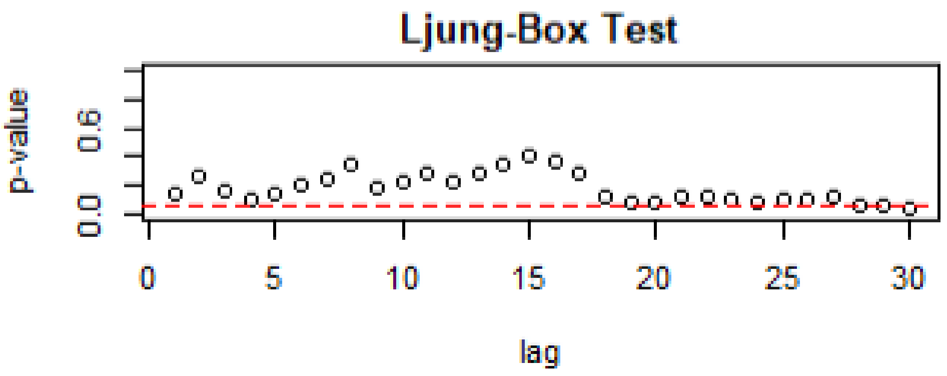


Figure 19 Ljung-Box Test

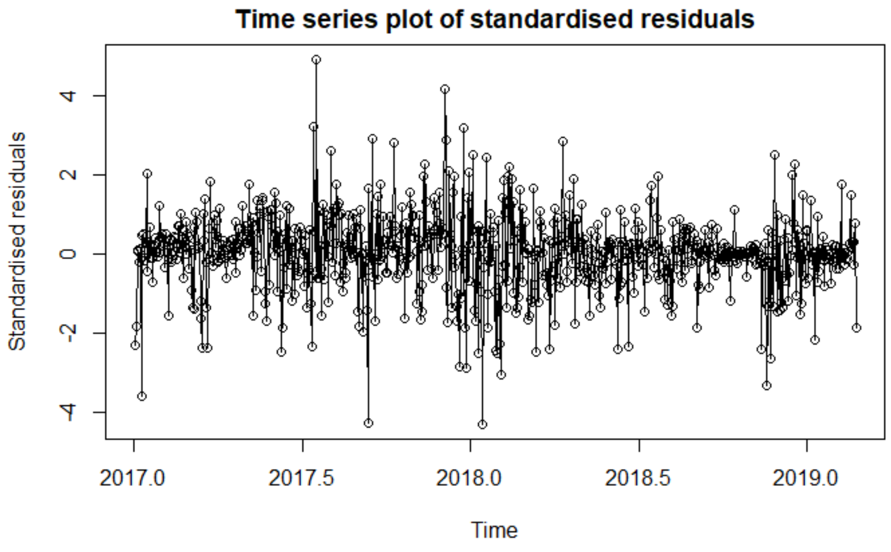


Figure 20 Time Series Plot of Standardized Residuals

In the ACF plot we can see that there is no lag exceeding the limits, next the Ljung-Box suggest that the residuals or P values in the plot are higher than 5% which confirms the squared residuals are uncorrelated over time and hence the standardized residuals may be independent.

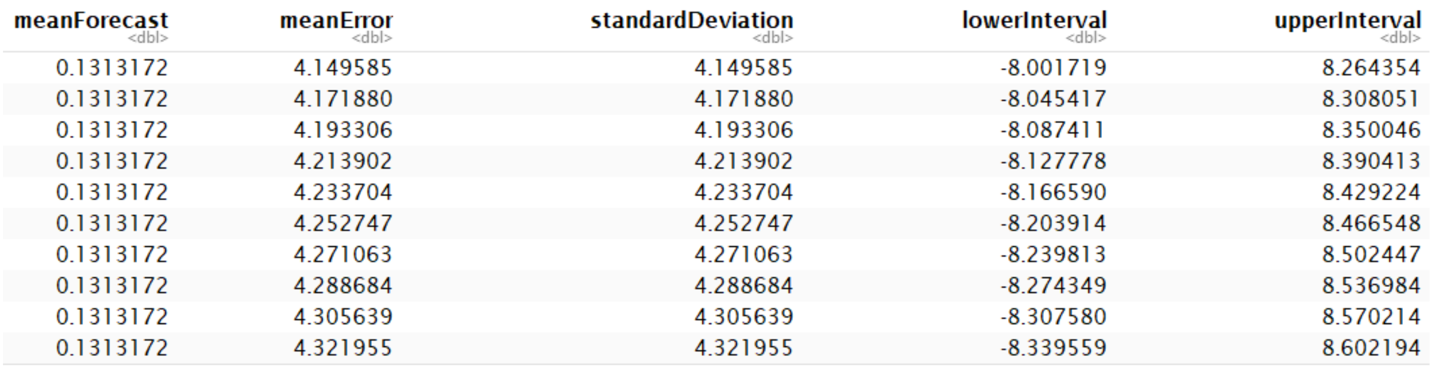
RESULTS

From all the analysis done above we can conclude that the prediction was manipulated using the adequate model GARCH(1,1) with the parameters.

FORECAST

The next aim is to forecast the values for the next years using the best fit model which is GARCH(1,1) which is as follows:

Prediction for the next 10 days



meanForecast	meanError	standardDeviation	lowerInterval	upperInterval
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
0.1313172	4.149585	4.149585	-8.001719	8.264354
0.1313172	4.171880	4.171880	-8.045417	8.308051
0.1313172	4.193306	4.193306	-8.087411	8.350046
0.1313172	4.213902	4.213902	-8.127778	8.390413
0.1313172	4.233704	4.233704	-8.166590	8.429224
0.1313172	4.252747	4.252747	-8.203914	8.466548
0.1313172	4.271063	4.271063	-8.239813	8.502447
0.1313172	4.288684	4.288684	-8.274349	8.536984
0.1313172	4.305639	4.305639	-8.307580	8.570214
0.1313172	4.321955	4.321955	-8.339559	8.602194

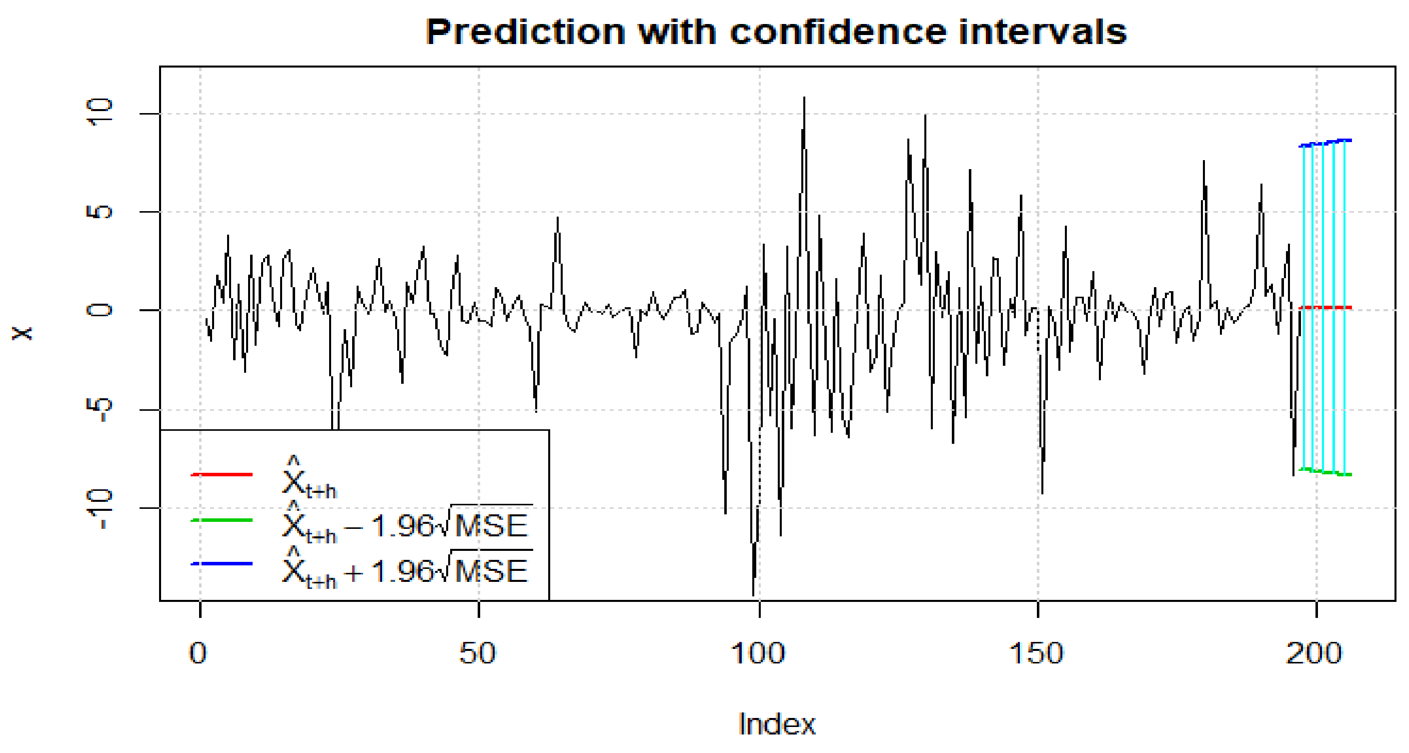


Figure 21 **Prediction of the next 10 years.**

From the fore cast table we can see that when we fit the GARCH model the forecast gives the upper and lower interval which is calculated on the sigma values but not the mean values as when we apply ARIMA models , hence it is not possible to fit the MASE for the Bitcoin Series.

DISCUSSION

The Bitcoin Series was is used to predict the Bitcoin value for the next ten days based on the past time series data. For this first we analysed the series by using various tests like correlation test and Dickey-Fuller Test, further the series was decomposed which gave more dept information of the series. From analyzing the series transformation and differencing was applied and ARIMA was used to fit the model, as the results of the test and the model fitting was not suitable the data was split into two parts to forecast the series.

The split series were also analyzed using the ACF, PACF and Dickey-Fuller Test and transformation and differencing was applied to make the series stationary the McLeod-Li test was applied which indicated presence of ARCH in the series and hence GARCH models were used to fit the series.

After considering the GARCH models' residual analysis was carried on all the models obtained through which the best model was GARCH(1,1) and was used to forecast the values of next 10 days.

CONCLUSION

The time series data was split into two and the GARCH model was selected as a bet model for the second split and the prediction was done using the above said model. This can be improved by tuning the model and better prediction can be done for future reference.

Despite the bitcoin time series data set was fitted on ARCH and GARCH model, further models ARIMA and GARCH could be fitted by splitting the data set from the top intervention point.

ARIMA and GARCH would predict the values with mean average and the squared error could be computed over the forecast values. Nevertheless, the squared error has been computed over the fitted values.

REFERENCE

https://www.quantstart.com/articles/White-Noise-and-Random-Walks-in-Time-Series-Analysis

https://www.kaggle.com/kp4920/s-p-500-stock-data-time-series-analysis

APPENDIX

#load the packages

```
#LOAD THE PACKAGES
library(TSA)

library(ks)
library(readr)

library(tseries)
library(FitAR)

library(lmtest)

library(plyr)
library(base)
library(devtools)
library(forecast)
library( fUnitRoots)

library(fGarch)
library(imputeTS)

library(stats)
source('C:/WorkingFolder/2ndyear/Time series/Lab/sort.score.R')

#Import and check the data
BitCoinData = read.csv("C:/WorkingFolder/2ndyear/Time Series/Project/Bitcoin_Historical_P
rice.csv", header = TRUE, stringsAsFactors = FALSE)

#checking structure of data
str(BitCoinData)

## 'data.frame':    2130 obs. of  2 variables:
##  $ Date : chr  "27/04/2013" "28/04/2013" "29/04/2013" "30/04/2013" ...
##  $ Close: num  134 145 139 117 105 ...

head(BitCoinData)

##         Date  Close
## 1 27/04/2013 134.21
## 2 28/04/2013 144.54
## 3 29/04/2013 139.00
## 4 30/04/2013 116.99
## 5  1/05/2013 105.21
## 6  2/05/2013  97.75

tail(BitCoinData)

##            Date   Close
## 2125 19/02/2019 3947.09
## 2126 20/02/2019 3999.82
## 2127 21/02/2019 3954.12
## 2128 22/02/2019 4005.53
## 2129 23/02/2019 4142.53
## 2130 24/02/2019 3810.43

length(BitCoinData$Close)

## [1] 2130

count(is.na(BitCoinData$Close))

##       x freq
## 1 FALSE 2130

#Convert data into time series
BitcoinDataTS<- ts(as.vector(BitCoinData[,-1]), start=c(2013,117),frequency=365)
#checking for NA Values
statsNA(BitcoinDataTS)
```

```
## [1] "Length of time series:"
## [1] 2130
## [1] "-------------------------"
## [1] "Number of Missing Values:"
## [1] 0
## [1] "-------------------------"
## [1] "Percentage of Missing Values:"
## [1] "0%"
## [1] "-------------------------"
## [1] "No NAs in the time Series."

## [1] "No NAs"

plot(BitcoinDataTS, main="Bitcoin Time series plot-2013 to 2019", col="Maroon", ylab="Coi
n Value (USD)")
```

```
#check the correlation between previous day and current day
y = BitcoinDataTS
x = zlag(BitcoinDataTS)
index = 2:length(x)    # Create an index to get rid of the first NA value in x
cor(y[index],x[index])

## [1] 0.9976471

plot(y=y,x=x,ylab='Bit Coin Values', xlab='Previous Day Values', col="Maroon")
```

```
#unit root test for stationarity
adf.test(BitcoinDataTS)

##
##  Augmented Dickey-Fuller Test
##
## data:  BitcoinDataTS
## Dickey-Fuller = -2.6913, Lag order = 12, p-value = 0.2856
## alternative hypothesis: stationary

#Indication of non-stationarity

#Plot of stationary series
plot(diff(log(BitcoinDataTS)), main="Daily return of Bitcoin value : April 27, 2013 to Fe
bruary 24, 2019",col="brown" )
```

```
#checking the best fit of ARIMA
auto.arima(diff(log(BitcoinDataTS)),max.d = 5,max.p =5,max.q=5)
```

```
## Series: diff(log(BitcoinDataTS))
## ARIMA(0,0,0) with non-zero mean
##
## Coefficients:
##         mean
##       0.0016
## s.e.  0.0009
##
## sigma^2 estimated as 0.001894:  log likelihood=3653.2
## AIC=-7302.4   AICc=-7302.39   BIC=-7291.07
```

```
#Splitting the data into two
BitcoinData1<-BitCoinData[1:1345,]
BitcoinData2<-BitCoinData[1346:2130,]
#checking the length of dataset
length(BitcoinData2$Close)
```

```
## [1] 785
```

```
#converting the splited data sets into time series
BitcoinDataTS1 <- ts(as.vector(BitcoinData1[,-1]), start=c(2013,117),frequency=365)
BitcoinDataTS2 <- ts(as.vector(BitcoinData2[,-1]), start=c(2017),frequency=365)

#plot the time series data

par(mfrow=c(1,1))
plot(BitcoinDataTS1, main="Time series plot-2013 to 2016", col="Maroon", ylab="Coin Value
 (USD)")
```

```
plot(BitcoinDataTS2, main="Time series plot-2017 to 2019", col="Maroon", ylab="Coin Value
 (USD)")
```

```
#check the correlation between previous day and current day
y1 = BitcoinDataTS1
x1 = zlag(BitcoinDataTS1)
index = 2:length(x1)    # Create an index to get rid of the first NA value in x
cor(y1[index],x1[index])
```

```
## [1] 0.9944077
```

```
y2 = BitcoinDataTS2
x2 = zlag(BitcoinDataTS2)
index = 2:length(x2)    # Create an index to get rid of the first NA value in x
cor(y2[index],x2[index])
```

```
## [1] 0.9946823
```

```
#plot the lagged time series data to check the correlation
par(mfrow=c(1,2))
plot(y=y1,x=x1,ylab='Bit Coin Values', xlab='Previous Day Values', col="Maroon")
plot(y=y2,x=x2,ylab='Bit Coin Values', xlab='Previous Day Values', col="Maroon")
```

```
#test for stationarity
adfTest(BitcoinDataTS1)

##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 1
##   STATISTIC:
##     Dickey-Fuller: 0.3251
##   P VALUE:
##     0.72
##
## Description:
##  Fri Jun 07 01:21:21 2019 by user: clarine,shamini,tarun

adfTest(BitcoinDataTS2)

##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 1
##   STATISTIC:
##     Dickey-Fuller: -0.7665
##   P VALUE:
##     0.372
##
## Description:
##  Fri Jun 07 01:21:21 2019 by user: clarine,shamini,tarun

#Indication of non-stationarity

#Decomposing a time series involves separating the time series into trend and irregular c
omponents.
par(mfrow=c(2,2))
decbitcoin<-decompose(BitcoinDataTS1)
plot(decbitcoin, col = "red")
```

```
decbitcoin<-decompose(BitcoinDataTS2)
plot(decbitcoin, col = "red")
```

```
#ACF AND PACF plots to analyses non stationarity
par(mfrow=c(2,2))
#Trend is apparent from ACF and PACF plots
acf(BitcoinDataTS1)
pacf(BitcoinDataTS1)

acf(BitcoinDataTS2)
pacf(BitcoinDataTS2)
```

```
# Slowly decaying pattern in ACF and very high first correlation in PACF
# implies the existence of trend and non-stationary.

#Return series
y.coin1=diff(log(BitcoinDataTS1))*100
y.coin2=diff(log(BitcoinDataTS2))*100

#Plotting the return series
#par(mfrow=c(2,1))
plot(y.coin1, main="Daily return of Bitcoin value : 2013 to 2016",col="brown" )
```

```
plot(y.coin2, main="Daily return of Bitcoin value : 2017 to 2019",col="brown" )
```

```
#non-stationarity test
adfTest(y.coin1)

##
## Title:
##  Augmented Dickey-Fuller Test
m## Test Results:
##   PARAMETER:
##     Lag Order: 1
##   STATISTIC:
##     Dickey-Fuller: -27.1444
##   P VALUE:
##     0.01
##
## Description:
##  Fri Jun 07 01:21:22 2019 by user: clarine,shamini,tarun
```

```
adfTest(y.coin2)

## 
## Title:
##  Augmented Dickey-Fuller Test
## 
## Test Results:
##   PARAMETER:
##     Lag Order: 1
##   STATISTIC:
##     Dickey-Fuller: -19.1041
##   P VALUE:
##     0.01 
## 
## Description:
##  Fri Jun 07 01:21:22 2019 by user: clarine,shamini,tarun

#Checking for best arima on both series
auto.arima(y.coin1, max.d = 5,max.p =5,max.q=5)

## Series: y.coin1 
## ARIMA(0,0,0) with zero mean 
## 
## sigma^2 estimated as 17.98:  log likelihood=-3848.75
## AIC=7699.5   AICc=7699.5   BIC=7704.7

auto.arima(y.coin2, max.d = 5,max.p =5,max.q=5)

## Series: y.coin2 
## ARIMA(1,1,0) with drift 
## 
## Coefficients:
##           ar1    drift
##       -0.5121  -0.0117
## s.e.   0.0308   0.1298
## 
## sigma^2 estimated as 30.23:  log likelihood=-2444.71
## AIC=4895.42   AICc=4895.45   BIC=4909.41

#ACF AND PACF plots to analyses non stationarity
par(mfrow=c(2,2))

acf(y.coin1)
pacf(y.coin1)

acf(y.coin2)
pacf(y.coin2)
```

```
#extended ACF plot
eacf(y.coin1)

## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 o o o x o x o o o o x  o  o  o
## 1 o o o x o x o o o o x  o  o  o
## 2 x x o x o x o o o o x  o  o  o
## 3 x x o o o x o o o o x  x  o  o
## 4 x x x x o x o o o o x  x  o  o
## 5 x x x x x o o o o o x  o  o  o
## 6 o x x x x x o o o o o  o  o  o
## 7 o x x o x x x o o o o  o  o  o

eacf(y.coin2)

## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 o o o o o o o o o o o  o  o  o
## 1 x o o o o o o o o o o  o  o  o
## 2 x x o o o o o o o o o  o  o  o
## 3 x x x o o o o o o o o  o  o  o
## 4 x x x x o o o o o o o  o  o  o
## 5 x x x x x o o o o o o  o  o  o
## 6 x x x x x x o o o o o  o  o  o
## 7 x x o x x x x o o o o  o  o  o

#main="McLeod-Li test statistics
par(mfrow=c(2,1))
McLeod.Li.test(y=y.coin1,main="McLeod-Li test statistics split-1 series")
McLeod.Li.test(y=y.coin2,main="McLeod-Li test statistics split-2 series")
```

```
#normality test
par(mfrow=c(2,1))
qqnorm(y.coin1,main="Q-Q Normal Plot of Bit coin series -1 ")
qqline(y.coin1)

qqnorm(y.coin2,main="Q-Q Normal Plot of Bit coin series- 2 ")
qqline(y.coin2)
```

```
#use squared and absolute value series
abs.y.coin1 = abs(y.coin1)
sq.y.coin1 = y.coin1^2

abs.y.coin2 = abs(y.coin2)
sq.y.coin2 = y.coin2^2
```

```
#ACF and PACF plot for return
par(mfrow=c(2,1))
acf(abs.y.coin1, ci.type="ma",main="The sample ACF plot for return series 1-(Absolute)")
pacf(abs.y.coin1, main="The sample PACF plot for return series")
```

```
acf(sq.y.coin1, ci.type="ma",main="The sample ACF plot for return series 1 - (squareroot)
")
pacf(sq.y.coin1, main="The sample PACF plot for return series")
```

```
#eacf of return
eacf(abs.y.coin1)

## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x x x x x x x x x x x  x  x  x
## 1 x x o x x x o o o x o  o  o  o
## 2 x x o o o x o o o x o  o  o  o
## 3 x x x o o x o o o x o  o  o  o
## 4 x x x o o o o o o x o  o  o  o
## 5 x x x x x x o o o o o  o  o  o
## 6 x x x x x x o o o o o  o  o  o
## 7 x x x x x x x o o o o  o  o  o

eacf(sq.y.coin1)

## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x x x x x o x x x x x  x  x  x
## 1 x x x x x x o x o x o  o  x  x
## 2 x x o o o x o o o x o  o  x  o
## 3 x x x o o x o o o x o  o  x  o
## 4 x x x o o o o o o o o  o  x  o
## 5 x x x x x x o o o o o  o  o  o
## 6 x x x x x x o o o o o  o  o  o
## 7 x x x x o x x o o o o  o  o  o

par(mfrow=c(2,1))
acf(abs.y.coin2, ci.type="ma",main="The sample ACF plot for return series-2 (Absolute)")
pacf(abs.y.coin2, main="The sample PACF plot for return series")
```

```
acf(sq.y.coin2, ci.type="ma",main="The sample ACF plot for return series-2 (squareroot)")
pacf(sq.y.coin2, main="The sample PACF plot for return series")
```

```
eacf(abs.y.coin2)
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x x x x x x x x x x x  x  x  x
## 1 x o o o o o o o o o o  o  o  o
## 2 x x o o o o o o o o o  o  o  o
## 3 x x x o o o o o o o o  o  o  o
## 4 x x x o o o o o o o o  o  o  o
## 5 x x x x x o o o o o o  o  o  o
## 6 x o o x x x o o o o o  o  o  o
## 7 x x o o o x x o o o o  o  o  o
```

```
eacf(sq.y.coin2)
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x o x x x x o o o o o  o  o  o
## 1 x x o o o o o o o o o  o  o  o
## 2 x x o o o o o o o o o  o  o  o
## 3 x x o o o o o o o o o  o  o  o
## 4 x x x o o o o o o o o  o  o  o
## 5 x x x x o o o o o o o  o  o  o
## 6 x o x x x o o o o o o  o  o  o
## 7 x o x o o x o o o o o  o  o  o
```

```
#Residual Analysis Function
residual.analysis <- function(model, std = TRUE, start = 2, class = c("ARIMA","GARCH","AR
MA-GARCH")[1]){
  # If you have an output from arima() function use class = "ARIMA"
  # If you have an output from garch() function use class = "GARCH"
  # If you have an output from ugarchfit() function use class = "ARMA-GARCH"
  library(TSA)
  library(FitAR)
  if (class == "ARIMA"){
    if (std == TRUE){
      res.model = rstandard(model)
    }else{
      res.model = residuals(model)
    }
  }else if (class == "GARCH"){
    res.model = model$residuals[start:model$n.used]
  }else if (class == "ARMA-GARCH"){
    res.model = model@fit$residuals
  }else {
    stop("The argument 'class' must be either 'ARIMA' or 'GARCH' ")
  }
  par(mfrow=c(3,2))
  plot(res.model, type='o',ylab='Standardized residuals', main="Time series plot of stand
ardized residuals")
```

```
  abline(h=0)
  hist(res.model, main="Histogram of standardized residuals")
  acf(res.model, main="ACF of standardized residuals")
  pacf(res.model, main="PACF of standardized residuals")
  qqnorm(res.model, main="QQ plot of standardized residuals")
  qqline(res.model, col = 2)
  print(shapiro.test(res.model))
  k=0
  LBQPlot(res.model, lag.max = 30, StartLag = k + 1, k = 0, SquaredQ = FALSE)
}
```

RESIDUAL ANLAYSIS

SPLIT1 RESIDUAL CHECKS

GARCH (2,2),

```
m.22 = garch(y.coin1,order=c(2,2),trace = FALSE)
summary(m.22)# All the coefficients but aplha_2 are significant at 5% level of significan
ce.

##
## Call:
## garch(x = y.coin1, order = c(2, 2), trace = FALSE)
##
## Model:
## GARCH(2,2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.57356 -0.27650  0.03149  0.37649  5.94516
##
## Coefficient(s):
##     Estimate  Std. Error  t value Pr(>|t|)
## a0 1.438e+01   1.653e+00    8.694  < 2e-16 ***
## a1 1.506e-01   1.493e-02   10.093  < 2e-16 ***
## a2 1.127e-01   3.253e-02    3.464 0.000532 ***
## b1 9.472e-14   1.565e-01    0.000 1.000000
## b2 2.102e-03   7.238e-02    0.029 0.976838
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Diagnostic Tests:
##  Jarque Bera Test
##
## data:  Residuals
## X-squared = 3693.5, df = 2, p-value < 2.2e-16
##
```

```
##
##  Box-Ljung test
##
## data:  Squared. Residuals
## X-squared = 5.7807, df = 1, p-value = 0.0162
```

```
m.22_2 = garchFit(formula = ~garch(2,2), data =y.coin1, trace = FALSE )
summary(m.22_2)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(2, 2), data = y.coin1, trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(2, 2)
## <environment: 0x00000000215160f0>
##  [data = y.coin1]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##        mu      omega     alpha1     alpha2      beta1      beta2
## 0.1048049  0.3325119  0.1805395  0.0097485  0.1967100  0.6171910
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.104805    0.075653    1.385  0.16595
## omega   0.332512    0.108905    3.053  0.00226 **
## alpha1  0.180539    0.031043    5.816 6.04e-09 ***
## alpha2  0.009749    0.029116    0.335  0.73776
## beta1   0.196710    0.081901    2.402  0.01631 *
## beta2   0.617191    0.069795    8.843  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -3559.659    normalized:  -2.648556
##
## Description:
##  Fri Jun 07 01:21:26 2019 by user: clarine,shamini,tarun
##
```

```
##
## Standardized Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  6248.317  0
##  Shapiro-Wilk Test  R    W      0.8814262 0
##  Ljung-Box Test     R    Q(10)  27.98342  0.001816316
##  Ljung-Box Test     R    Q(15)  35.13613  0.002351602
##  Ljung-Box Test     R    Q(20)  41.13565  0.003578637
##  Ljung-Box Test     R^2  Q(10)  6.272409  0.7918809
##  Ljung-Box Test     R^2  Q(15)  8.119775  0.918891
##  Ljung-Box Test     R^2  Q(20)  9.841579  0.9709544
##  LM Arch Test       R    TR^2   7.194985  0.8444636
##
## Information Criterion Statistics:
##      AIC      BIC      SIC     HQIC
## 5.306040 5.329270 5.306001 5.314742
```

```
residual.analysis(m.22,class="GARCH”, start=3)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res.model
## W = 0.87544, p-value < 2.2e-16
```

GARCH(3,2)

```
m.32 = garch(y.coin1,order=c(3,2),trace = FALSE)
summary(m.32)# All the coefficients but aplha_2 are significant at 5% level of significan
ce.
```

```
##
## Call:
## garch(x = y.coin1, order = c(3, 2), trace = FALSE)
##
## Model:
## GARCH(3,2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.65847 -0.27995  0.03253  0.38570  5.99822
##
## Coefficient(s):
##     Estimate  Std. Error  t value Pr(>|t|)
## a0 1.348e+01   1.935e+00    6.964 3.31e-12 ***
## a1 1.486e-01   1.470e-02   10.107  < 2e-16 ***
## a2 1.140e-01   3.892e-02    2.930  0.00339 **
## b1 4.140e-13   1.943e-01    0.000  1.00000
## b2 5.601e-04   1.058e-01    0.005  0.99578
## b3 1.549e-02   3.721e-02    0.416  0.67726
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Diagnostic Tests:
##  Jarque Bera Test
##
## data:  Residuals
## X-squared = 3750, df = 2, p-value < 2.2e-16
##
##
##  Box-Ljung test
##
## data:  Squared. Residuals
## X-squared = 5.2169, df = 1, p-value = 0.02237
```

```
m.32_2 = garchFit(formula = ~garch(3,2), data =y.coin1, trace = FALSE )
summary(m.32_2)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(3, 2), data = y.coin1, trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(3, 2)
## <environment: 0x000000001fb929a8>
##  [data = y.coin1]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##         mu       omega      alpha1      alpha2      alpha3       beta1
## 0.09959612  0.32156660  0.18436717  0.00000001  0.00000001  0.21486638
##      beta2
## 0.60425825
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu     9.960e-02   7.572e-02    1.315  0.18840
## omega  3.216e-01   1.241e-01    2.592  0.00954 **
## alpha1 1.844e-01   3.661e-02    5.036 4.76e-07 ***
## alpha2 1.000e-08   2.974e-02    0.000  1.00000
## alpha3 1.000e-08   4.637e-02    0.000  1.00000
## beta1  2.149e-01   8.923e-02    2.408  0.01603 *
## beta2  6.043e-01   7.722e-02    7.826 5.11e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -3559.704    normalized:  -2.648589
##
## Description:
##  Fri Jun 07 01:21:27 2019 by user: clarine,shamini,tarun
##
##
## Standardized Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  6198.791  0
##  Shapiro-Wilk Test  R    W      0.8815626 0
##  Ljung-Box Test     R    Q(10)  27.81487  0.001932614
##  Ljung-Box Test     R    Q(15)  34.95357  0.00249674
##  Ljung-Box Test     R    Q(20)  40.91669  0.00381807
##  Ljung-Box Test     R^2  Q(10)  6.268784  0.7921983
##  Ljung-Box Test     R^2  Q(15)  8.063729  0.9212023
##  Ljung-Box Test     R^2  Q(20)  9.806012  0.9715546
##  LM Arch Test       R    TR^2   7.201078  0.8440443
##
## Information Criterion Statistics:
##      AIC      BIC      SIC     HQIC
## 5.307596 5.334697 5.307542 5.317747
```

```
residual.analysis(m.32,class="GARCH”, start=4)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res.model
## W = 0.87717, p-value < 2.2e-16
```

GARCH(3,3)

```
m.33 = garch(y.coin1,order=c(3,3),trace = FALSE)
summary(m.33)# All the coefficients but aplha_2 are significant at 5% level of
significance.
```

```
##
## Call:
## garch(x = y.coin1, order = c(3, 3), trace = FALSE)
##
## Model:
```

```
## GARCH(3,3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.73129 -0.27823  0.03347  0.39033  6.16400
##
## Coefficient(s):
##     Estimate  Std. Error  t value Pr(>|t|)
## a0 1.258e+01   3.220e+00    3.906 9.38e-05 ***
## a1 1.416e-01   1.393e-02   10.159  < 2e-16 ***
## a2 1.023e-01   4.825e-02    2.120    0.034 *
## a3 6.685e-02   4.707e-02    1.420    0.156
## b1 2.197e-13   3.133e-01    0.000    1.000
## b2 9.961e-03   2.464e-01    0.040    0.968
## b3 1.447e-02   1.291e-01    0.112    0.911
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Diagnostic Tests:
##  Jarque Bera Test
##
## data:  Residuals
## X-squared = 4081.6, df = 2, p-value < 2.2e-16
##
##
##  Box-Ljung test
##
## data:  Squared. Residuals
## X-squared = 5.2114, df = 1, p-value = 0.02244
```

```
m.33_2 = garchFit(formula = ~garch(3,3), data =y.coin1, trace = FALSE )
summary(m.33_2)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(3, 3), data = y.coin1, trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(3, 3)
## <environment: 0x0000000025639880>
##  [data = y.coin1]
##
## Conditional Distribution:
##  norm
##
```

```
## Coefficient(s):
##         mu       omega      alpha1      alpha2      alpha3       beta1
## 0.09964722  0.42325650  0.17787453  0.03775745  0.02580334  0.00000001
##      beta2       beta3
## 0.61954753  0.14535294
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu     9.965e-02   7.546e-02    1.320 0.186683
## omega  4.233e-01   1.050e-01    4.030 5.59e-05 ***
## alpha1 1.779e-01   3.193e-02    5.572 2.52e-08 ***
## alpha2 3.776e-02   1.071e-02    3.527 0.000421 ***
## alpha3 2.580e-02   7.910e-03    3.262 0.001106 **
## beta1  1.000e-08          NA       NA       NA
## beta2  6.195e-01   6.704e-02    9.241  < 2e-16 ***
## beta3  1.454e-01          NA       NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -3559.599    normalized:  -2.648511
##
## Description:
##  Fri Jun 07 01:21:27 2019 by user: clarine,shamini,tarun
##
##
## Standardized Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  6063.442  0
##  Shapiro-Wilk Test  R    W      0.8817022 0
##  Ljung-Box Test     R    Q(10)  27.89516  0.00187634
##  Ljung-Box Test     R    Q(15)  35.09135  0.002386435
##  Ljung-Box Test     R    Q(20)  41.17055  0.003541809
##  Ljung-Box Test     R^2  Q(10)  6.368516  0.7834105
##  Ljung-Box Test     R^2  Q(15)  8.238954  0.9138456
##  Ljung-Box Test     R^2  Q(20)  9.982256  0.9684925
##  LM Arch Test       R    TR^2   7.336078  0.8346287
##
## Information Criterion Statistics:
##      AIC      BIC      SIC     HQIC
## 5.308927 5.339899 5.308856 5.320529

residual.analysis(m.33,class="GARCH”, start=4)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res.model
## W = 0.87528, p-value < 2.2e-16
```

```
sort.score(AIC(m.32,m.33,m.22), score = "aic")
```

```
##      df      AIC
## m.33  7 7351.696
## m.32  6 7372.085
## m.22  5 7404.991
```

SPLIT 2 **RESIDUAL CHECKS**

GARCH (1,1)

```
m.11 = garch(y.coin2,order=c(1,1),trace = FALSE)
summary(m.11)# All the coefficients but aplha_2 are significant at 5% level of significan
ce.
```

```
##
## Call:
## garch(x = y.coin2, order = c(1, 1), trace = FALSE)
##
## Model:
## GARCH(1,1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4.28244 -0.38571  0.07028  0.51381  4.91338
##
## Coefficient(s):
##     Estimate  Std. Error  t value Pr(>|t|)
## a0 1.856e+01   3.546e+00    5.235 1.65e-07 ***
## a1 1.311e-01   3.195e-02    4.103 4.07e-05 ***
## b1 3.931e-13   1.644e-01    0.000        1
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Diagnostic Tests:
##  Jarque Bera Test
##
## data:  Residuals
## X-squared = 262.76, df = 2, p-value < 2.2e-16
##
##
```

```
##  Box-Ljung test
##
## data:  Squared. Residuals
## X-squared = 0.71452, df = 1, p-value = 0.3979
```

```
m.11_2 = garchFit(formula = ~garch(1,1), data =y.coin2, trace = FALSE )
summary(m.11_2)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(1, 1), data = y.coin2, trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(1, 1)
## <environment: 0x00000000255c4478>
##  [data = y.coin2]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##      mu    omega   alpha1    beta1
## 0.13132  0.76959  0.11625  0.84983
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu       0.13132     0.13540    0.970 0.332133
## omega    0.76959     0.22581    3.408 0.000654 ***
## alpha1   0.11625     0.02239    5.193 2.07e-07 ***
## beta1    0.84983     0.02605   32.618  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -2231.705    normalized:  -2.846562
##
## Description:
##  Fri Jun 07 01:21:27 2019 by user: clarine,shamini,tarun
##
##
## Standardized Residuals Tests:
##                                  Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  292.7791  0
##  Shapiro-Wilk Test  R    W      0.9557599 1.292842e-14
##  Ljung-Box Test     R    Q(10)  19.96982  0.0295394
##  Ljung-Box Test     R    Q(15)  21.65265  0.1172406
##  Ljung-Box Test     R    Q(20)  31.74189  0.04612741
##  Ljung-Box Test     R^2  Q(10)  14.54403  0.149591
##  Ljung-Box Test     R^2  Q(15)  18.36692  0.2438668
##  Ljung-Box Test     R^2  Q(20)  22.82332  0.2975531
##  LM Arch Test       R    TR^2   14.90809  0.2465012
##
## Information Criterion Statistics:
##      AIC      BIC      SIC     HQIC
## 5.703328 5.727126 5.703276 5.712479
```

```
residual.analysis(m.11,class="GARCH”, start=2)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res.model
## W = 0.95488, p-value = 9.127e-15
```

```
plot(m.11$residuals,type='o',ylab='Standardized residuals', main="Time series plot of sta
ndardized residuals")
  abline(h=0)
```

GARCH (2,1)

```
m.21 = garch(y.coin2,order=c(2,1),trace = FALSE)
summary(m.21)# All the coefficients but aplha_2 are significant at 5% level of significan
ce.
```

```
##
## Call:
## garch(x = y.coin2, order = c(2, 1), trace = FALSE)
##
## Model:
## GARCH(2,1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4.30565 -0.38616  0.06907  0.51450  4.78768
##
## Coefficient(s):
##     Estimate  Std. Error  t value Pr(>|t|)
## a0 1.753e+01   3.560e+00    4.924 8.46e-07 ***
## a1 1.417e-01   3.424e-02    4.138 3.50e-05 ***
## b1 2.540e-13   1.511e-01    0.000    1.000
## b2 4.149e-02   6.371e-02    0.651    0.515
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Diagnostic Tests:
##  Jarque Bera Test
##
## data:  Residuals
## X-squared = 262.16, df = 2, p-value < 2.2e-16
##
##
##  Box-Ljung test
##
## data:  Squared. Residuals
## X-squared = 0.29861, df = 1, p-value = 0.5848
```

```
m.21_2 = garchFit(formula = ~garch(2,1), data =y.coin2, trace = FALSE )
summary(m.21_2)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(2, 1), data = y.coin2, trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(2, 1)
## <environment: 0x0000000020e16450>
##  [data = y.coin2]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##         mu       omega      alpha1      alpha2       beta1
## 0.13337459  0.77661656  0.11690713  0.00000001  0.84870765
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu     1.334e-01   1.355e-01    0.984 0.324907
## omega  7.766e-01   2.486e-01    3.124 0.001783 **
## alpha1 1.169e-01   3.443e-02    3.395 0.000686 ***
## alpha2 1.000e-08   4.041e-02    0.000 1.000000
## beta1  8.487e-01   3.146e-02   26.975  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -2231.361    normalized:  -2.846124
##
## Description:
##  Fri Jun 07 01:21:28 2019 by user: clarine,shamini,tarun
##
##
## Standardized Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  292.6325  0
##  Shapiro-Wilk Test  R    W      0.9558993 1.372625e-14
##  Ljung-Box Test     R    Q(10)  20.03189  0.02895254
##  Ljung-Box Test     R    Q(15)  21.74056  0.1147894
##  Ljung-Box Test     R    Q(20)  31.86438  0.04476505
##  Ljung-Box Test     R^2  Q(10)  14.40951  0.1551185
##  Ljung-Box Test     R^2  Q(15)  18.24074  0.2502208
##  Ljung-Box Test     R^2  Q(20)  22.69436  0.3040499
##  LM Arch Test       R    TR^2   14.8335   0.2506687
##
## Information Criterion Statistics:
##      AIC      BIC      SIC     HQIC
## 5.705002 5.734750 5.704922 5.716441
```

```
residual.analysis(m.21,class="GARCH”, start=3)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res.model
## W = 0.95477, p-value = 8.924e-15
```

GARCH (2,2),

```
m.22 = garch(y.coin2,order=c(2,2),trace = FALSE)
summary(m.22)# All the coefficients but aplha_2 are significant at 5% level of significan
ce.
```

```
##
## Call:
## garch(x = y.coin2, order = c(2, 2), trace = FALSE)
##
## Model:
## GARCH(2,2)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4.38264 -0.37757  0.07115  0.50447  4.78126
##
## Coefficient(s):
##     Estimate  Std. Error  t value Pr(>|t|)
## a0 1.650e+01   3.163e+00    5.216 1.83e-07 ***
## a1 1.346e-01   3.246e-02    4.148 3.36e-05 ***
## a2 8.034e-02   5.244e-02    1.532    0.126
## b1 2.754e-13   2.934e-01    0.000    1.000
## b2 3.946e-02   1.483e-01    0.266    0.790
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Diagnostic Tests:
##  Jarque Bera Test
##
## data:  Residuals
## X-squared = 289.47, df = 2, p-value < 2.2e-16
##
##
##  Box-Ljung test
##
## data:  Squared. Residuals
## X-squared = 0.66722, df = 1, p-value = 0.414
```

```
m.22_2 = garchFit(formula = ~garch(2,2), data =y.coin2, trace = FALSE )
summary(m.22_2)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(2, 2), data = y.coin2, trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(2, 2)
## <environment: 0x0000000026394e68>
##  [data = y.coin2]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##         mu       omega      alpha1      alpha2       beta1       beta2
## 0.12526544  1.04655302  0.16294649  0.00000001  0.30164517  0.48898102
##
```

```
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu     1.253e-01   1.354e-01    0.925  0.35502
## omega  1.047e+00   3.588e-01    2.917  0.00353 **
## alpha1 1.629e-01   4.107e-02    3.967 7.27e-05 ***
## alpha2 1.000e-08   5.312e-02    0.000  1.00000
## beta1  3.016e-01   2.182e-01    1.382  0.16684
## beta2  4.890e-01   1.873e-01    2.610  0.00905 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -2230.035    normalized:  -2.844433
##
## Description:
##  Fri Jun 07 01:21:28 2019 by user: clarine,shamini,tarun
##
##
## Standardized Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  272.3555  0
##  Shapiro-Wilk Test  R    W      0.9568483 2.070839e-14
##  Ljung-Box Test     R    Q(10)  19.66261  0.03261023
##  Ljung-Box Test     R    Q(15)  21.35396  0.1258963
##  Ljung-Box Test     R    Q(20)  31.53996  0.04845379
##  Ljung-Box Test     R^2  Q(10)  12.40143  0.2590881
##  Ljung-Box Test     R^2  Q(15)  16.71158  0.3363968
##  Ljung-Box Test     R^2  Q(20)  21.20537  0.3851347
##  LM Arch Test       R    TR^2   13.21093  0.3538977
##
## Information Criterion Statistics:
##      AIC      BIC      SIC     HQIC
## 5.704172 5.739869 5.704056 5.717898
```

```
residual.analysis(m.22,class="GARCH", start=4)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res.model
## W = 0.95415, p-value = 7.063e-15
```

GARCH (3,1),

```
m.31 = garch(y.coin2,order=c(3,1),trace = FALSE)
summary(m.31)# All the coefficients but aplha_2 are significant at 5% level of significan
ce.
```

```
##
## Call:
## garch(x = y.coin2, order = c(3, 1), trace = FALSE)
##
## Model:
## GARCH(3,1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4.31795 -0.38526  0.07084  0.51727  4.79697
##
## Coefficient(s):
##     Estimate  Std. Error  t value Pr(>|t|)
## a0 1.650e+01   3.554e+00    4.643 3.44e-06 ***
## a1 1.459e-01   3.603e-02    4.049 5.14e-05 ***
## b1 1.067e-13   1.547e-01    0.000    1.000
## b2 4.164e-02   6.312e-02    0.660    0.509
## b3 4.457e-02   7.707e-02    0.578    0.563
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Diagnostic Tests:
##  Jarque Bera Test
##
## data:  Residuals
## X-squared = 269.62, df = 2, p-value < 2.2e-16
##
##
##  Box-Ljung test
##
## data:  Squared. Residuals
## X-squared = 0.18128, df = 1, p-value = 0.6703
```

```
m.31_2 = garchFit(formula = ~garch(3,1), data =y.coin2, trace = FALSE )
summary(m.31_2)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(3, 1), data = y.coin2, trace = FALSE)
##
## Mean and Variance Equation:
```

```
##  data ~ garch(3, 1)
## <environment: 0x0000000021e4e9d0>
##  [data = y.coin2]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##         mu       omega      alpha1      alpha2      alpha3       beta1
## 0.11751040  0.91033172  0.09475631  0.00000001  0.04152701  0.82378176
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu     1.175e-01   1.358e-01    0.865  0.38701
## omega  9.103e-01   3.063e-01    2.972  0.00296 **
## alpha1 9.476e-02   3.001e-02    3.157  0.00159 **
## alpha2 1.000e-08   5.804e-02    0.000  1.00000
## alpha3 4.153e-02   5.560e-02    0.747  0.45515
## beta1  8.238e-01   4.093e-02   20.127  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -2231.989    normalized:  -2.846925
##
## Description:
##  Fri Jun 07 01:21:28 2019 by user: clarine,shamini,tarun
##
##
## Standardized Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  289.9588  0
##  Shapiro-Wilk Test  R    W      0.9552147 1.024135e-14
##  Ljung-Box Test     R    Q(10)  20.63159  0.0238141
##  Ljung-Box Test     R    Q(15)  22.30138  0.1001417
##  Ljung-Box Test     R    Q(20)  32.5653   0.03763487
##  Ljung-Box Test     R^2  Q(10)  13.79311  0.1826394
##  Ljung-Box Test     R^2  Q(15)  17.63676  0.2822434
##  Ljung-Box Test     R^2  Q(20)  22.08602  0.3358611
##  LM Arch Test       R    TR^2   13.73863  0.3177097
##
## Information Criterion Statistics:
##      AIC      BIC      SIC     HQIC
## 5.709157 5.744854 5.709041 5.722883
```

```
residual.analysis(m.31,class="GARCH”, start=4)

##
##  Shapiro-Wilk normality test
##
## data:  res.model
## W = 0.95492, p-value = 9.765e-15
```

GARCH(3,2)

```
m.32 = garch(y.coin2,order=c(3,2),trace = FALSE)
summary(m.32)# All the coefficients but aplha_2 are significant at 5% level of significan
ce.

##
## Call:
## garch(x = y.coin2, order = c(3, 2), trace = FALSE)
##
## Model:
## GARCH(3,2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4.42764 -0.37776  0.07298  0.50102  4.77528
##
## Coefficient(s):
##     Estimate  Std. Error  t value Pr(>|t|)
## a0 1.547e+01   4.252e+00    3.638 0.000275 ***
## a1 1.328e-01   3.361e-02    3.953 7.73e-05 ***
## a2 9.150e-02   6.217e-02    1.472 0.141077
## b1 3.796e-13   3.188e-01    0.000 1.000000
## b2 3.183e-02   2.356e-01    0.135 0.892537
## b3 4.325e-02   1.436e-01    0.301 0.763259
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Diagnostic Tests:
##  Jarque Bera Test
##
## data:  Residuals
## X-squared = 293.43, df = 2, p-value < 2.2e-16
##
##
##  Box-Ljung test
##
```

```
## data:  Squared. Residuals
## X-squared = 0.63678, df = 1, p-value = 0.4249
```

```
m.32_2 = garchFit(formula = ~garch(3,2), data =y.coin2, trace = FALSE )
summary(m.32_2)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(3, 2), data = y.coin2, trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(3, 2)
## <environment: 0x0000000025ea2718>
##  [data = y.coin2]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##         mu       omega      alpha1      alpha2      alpha3       beta1
## 0.11675721  1.11016468  0.13514107  0.00000001  0.03480404  0.34857043
##      beta2
## 0.43177891
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu     1.168e-01   1.367e-01    0.854  0.39302
## omega  1.110e+00   3.944e-01    2.815  0.00488 **
## alpha1 1.351e-01   4.067e-02    3.323  0.00089 ***
## alpha2 1.000e-08   9.251e-02    0.000  1.00000
## alpha3 3.480e-02   7.909e-02    0.440  0.65989
## beta1  3.486e-01   4.140e-01    0.842  0.39986
## beta2  4.318e-01   3.796e-01    1.137  0.25540
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -2231.118    normalized:  -2.845814
##
## Description:
##  Fri Jun 07 01:21:28 2019 by user: clarine,shamini,tarun
##
```

```
##
## Standardized Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  274.9673  0
##  Shapiro-Wilk Test  R    W      0.9559701 1.415108e-14
##  Ljung-Box Test     R    Q(10)  20.02926  0.02897719
##  Ljung-Box Test     R    Q(15)  21.62061  0.1181446
##  Ljung-Box Test     R    Q(20)  31.78176  0.04568006
##  Ljung-Box Test     R^2  Q(10)  11.96404  0.2874707
##  Ljung-Box Test     R^2  Q(15)  16.31047  0.361723
##  Ljung-Box Test     R^2  Q(20)  20.78938  0.4096214
##  LM Arch Test       R    TR^2   12.32059  0.4202883
##
## Information Criterion Statistics:
##      AIC      BIC      SIC     HQIC
## 5.709485 5.751132 5.709328 5.725499
```

```
residual.analysis(m.32,class="GARCH”, start=4)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res.model
## W = 0.95475, p-value = 9.078e-15
```

*Models are GARCH(1,1), GARCH(2,1), ARMA(2,2), ARMA(3,1), ARMA(3,2)

```
sort.score(AIC(m.11,m.21,m.22,m.31,m.32), score = "aic")
```

```
##      df      AIC
## m.32  6 4536.153
## m.31  5 4544.839
## m.22  5 4549.953
## m.21  4 4556.948
## m.11  3 4567.986
```

PREDICTION FOR THE NEXT 10 DAYS

```
#Prediction for the next 10 days
par(mfrow=c(1,1))
plot((fitted(m.11)[,1])^2,type='l',ylab='Conditional Variance',xlab='t',main="Estimated C
onditional Variances of the Daily Returns")
```

```
# Changes in conditional variance at the beginning of the series and between observations
 300 and 400, then the conditional variance settles down.
```

```r
####We are using the m.11_2 instead of m.11 model in order to get the plot
fGarch::predict(m.11_2,n.ahead=10,trace=FALSE, plot=TRUE)
```

```r
# Forecasts for the confidence limits are based on the forecasts of conditional variance.

#Fitting GARCH and Predicting for next 10 days
library(rugarch)

model<-ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 1)), mean.mod
el = list(armaOrder = c(0,0), include.mean = FALSE),
                  distribution.model = "norm")

#Fitting model
Finalmodel<-ugarchfit(spec=model,data=y.coin2)

par(mfrow=c(1,1))
#plot(Finalmodel)

par(mfrow=c(1,1))

fittedmase1= diffinv(fitted(Finalmodel),d=1)

##Prediction
forc = ugarchforecast(Finalmodel, data = y.coin2, n.ahead = 10)
#plot(forc, which="all")

forecastsmase = forc@forecast$seriesFor

BitCoinForecast<-read.csv("C:/WorkingFolder/2ndyear/Time Series/Project/Bitcoin_Prices_Fo
recasts.csv", header = TRUE, stringsAsFactors = FALSE)

View(BitCoinForecast)
```

MASE TABLE

```r
observedmase1 <- BitcoinData2$Close
observedmase2 <- BitCoinForecast$Closing.price

View(observedmase1)

MASE = function(observed, fitted ){
  # observed: Observed series on the forecast period
  # fitted: Forecast values by your model
  Y.t = observed
  n = length(fitted)
  e.t = Y.t - fitted
  sum = 0
  for (i in 2:n){
    sum = sum + abs(Y.t[i] - Y.t[i-1] )
  }
  q.t = e.t / (sum/(n-1))
  MASE = data.frame( MASE = mean(abs(q.t)))
  return(list(MASE = MASE))
}
```

```
MASE(observedmase1,fittedmase1) ;
```

```
## $MASE
##       MASE
## 1 28.52268
```

```
MASE(observedmase2,forecastsmase)
```

```
## $MASE
##       MASE
## 1 119.6113
```

`