S&P500_Final

April 23, 2018

```
In [1]: # Importing the libraries
        library(ggplot2)
        library(gridExtra)
        library(grid)
        library(lattice)
        library(reshape)
        library(dplyr)
        library(forecast)
        library(tseries)
        library(portes)
        library(quantmod)
```

```
Attaching package: dplyr

The following object is masked from package:reshape:

    rename

The following object is masked from package:gridExtra:

    combine

The following objects are masked from package:stats:

    filter, lag

The following objects are masked from package:base:

    intersect, setdiff, setequal, union

Loading required package: parallel
Loading required package: xts
Loading required package: zoo

Attaching package: zoo

The following objects are masked from package:base:
```

```
    as.Date, as.Date.numeric


Attaching package: xts

The following objects are masked from package:dplyr:

    first, last

Loading required package: TTR
Version 0.4-0 included new data defaults. See ?getSymbols.
```

In [2]:
```
# Import and read the data
stock <- read.csv("all_stocks_5yr.csv")
attach(stock)
head(stock,5)
```

date	open	high	low	close	volume	Name
2013-02-08	15.07	15.12	14.63	14.75	8407500	AAL
2013-02-11	14.89	15.01	14.26	14.46	8882000	AAL
2013-02-12	14.45	14.51	14.10	14.27	8126000	AAL
2013-02-13	14.30	14.94	14.25	14.66	10259500	AAL
2013-02-14	14.94	14.96	13.16	13.99	31879900	AAL

In [3]:
```
# Using the summary and the string to look about the dataset
summary(stock)
str(stock)
```

```
         date                 open               high               low
 2017-12-05:   505   Min.   :   1.62   Min.   :   1.69   Min.   :   1.50
 2017-12-06:   505   1st Qu.:  40.22   1st Qu.:  40.62   1st Qu.:  39.83
 2017-12-07:   505   Median :  62.59   Median :  63.15   Median :  62.02
 2017-12-08:   505   Mean   :  83.02   Mean   :  83.78   Mean   :  82.26
 2017-12-11:   505   3rd Qu.:  94.37   3rd Qu.:  95.18   3rd Qu.:  93.54
 2017-12-12:   505   Max.   :2044.00   Max.   :2067.99   Max.   :2035.11
 (Other)   :616010   NA's   :11        NA's   :8         NA's   :8
     close              volume                Name
 Min.   :   1.59   Min.   :        0   A       :  1259
 1st Qu.:  40.24   1st Qu.:  1070320   AAL     :  1259
 Median :  62.62   Median :  2082094   AAP     :  1259
 Mean   :  83.04   Mean   :  4321823   AAPL    :  1259
 3rd Qu.:  94.41   3rd Qu.:  4284509   ABBV    :  1259
 Max.   :2049.00   Max.   :618237630   ABC     :  1259
                                       (Other):611486


'data.frame':        619040 obs. of  7 variables:
 $ date  : Factor w/ 1259 levels "2013-02-08","2013-02-11",..: 1 2 3 4 5 6 7 8 9 10 ...
```

```
$ open  : num  15.1 14.9 14.4 14.3 14.9 ...
$ high  : num  15.1 15 14.5 14.9 15 ...
$ low   : num  14.6 14.3 14.1 14.2 13.2 ...
$ close : num  14.8 14.5 14.3 14.7 14 ...
$ volume: int  8407500 8882000 8126000 10259500 31879900 15628000 11354400 14725200 11922100 60
$ Name  : Factor w/ 505 levels "A","AAL","AAP",..: 2 2 2 2 2 2 2 2 2 2 ...
```

In [4]:
```
#Now the dataset has some missing values and now we will try to
#remove all those to have a efficient analysis
stock[is.na(stock)] <- 0
```

0.1 We are going to Proceed Further on our Analysis On Banking & Finance Companies

In [5]:
```
# Filtering to only the Bank & Financial Sector companies
names <- c('BAC','BK','BBT','COF','SEHW','C','CFG','CMA','FITB','GS','HBAN','JPM','KEY',
banks <- subset(stock , Name == names)
banks$year <- strftime(banks$date,'%y')
banks$days <- strftime(banks$date , '%A')
banks$volume <- banks$volume/1000000
head(banks,5)
```

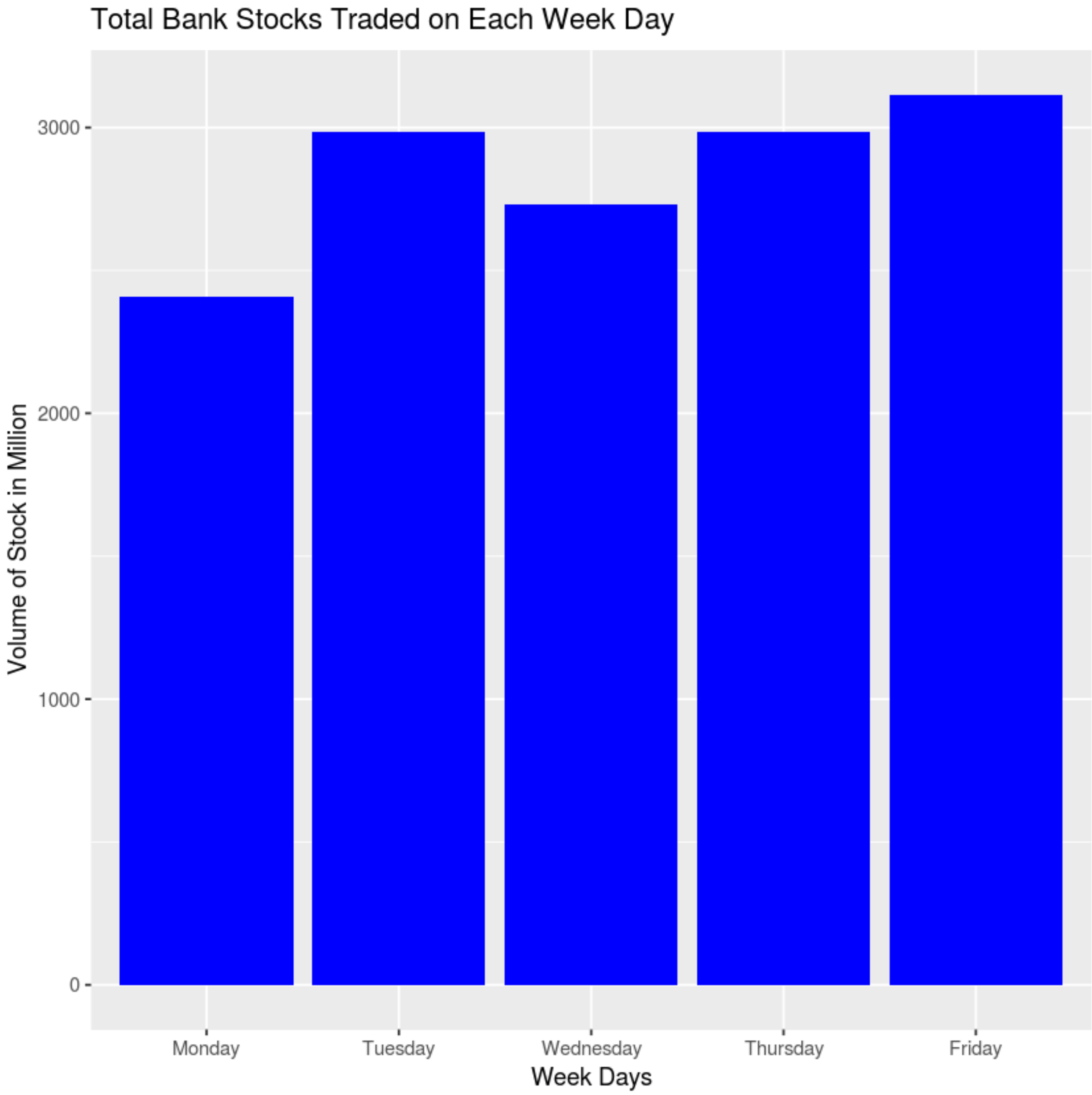
	date	open	high	low	close	volume	Name	year	days
72881	2013-02-25	11.600	11.61	10.98	11.03	205.9581	BAC	13	Monday
72901	2013-03-25	12.680	12.72	12.32	12.40	154.1264	BAC	13	Monday
72921	2013-04-23	11.920	12.16	11.90	12.07	176.5826	BAC	13	Tuesday
72941	2013-05-21	13.525	13.56	13.36	13.44	111.7066	BAC	13	Tuesday
72961	2013-06-19	13.285	13.40	13.17	13.19	103.6573	BAC	13	Wednesday

In [6]:
```
# Summary of the Banks dataframe Features
summary(banks)
```

```
        date            open             high             low
 2013-02-14:   2   Min.   :  7.07   Min.   :  7.17   Min.   :  7.05
 2013-02-26:   2   1st Qu.: 18.36   1st Qu.: 18.50   1st Qu.: 18.23
 2013-02-27:   2   Median : 40.55   Median : 40.88   Median : 40.19
 2013-03-04:   2   Mean   : 53.72   Mean   : 54.16   Mean   : 53.25
 2013-03-06:   2   3rd Qu.: 70.81   3rd Qu.: 71.74   3rd Qu.: 70.34
 2013-03-07:   2   Max.   :269.04   Max.   :273.79   Max.   :268.81
 (Other)   :1164
     close            volume                Name          year
 Min.   :  7.12   Min.   :  0.3574   BAC     : 63   Length:1176
 1st Qu.: 18.39   1st Qu.:  2.5010   BBT     : 63   Class :character
 Median : 40.44   Median :  4.9470   BK      : 63   Mode  :character
 Mean   : 53.72   Mean   : 12.0972   C       : 63
 3rd Qu.: 71.09   3rd Qu.: 11.8393   CMA     : 63
 Max.   :272.48   Max.   :231.4992   COF     : 63
                                     (Other):798
```

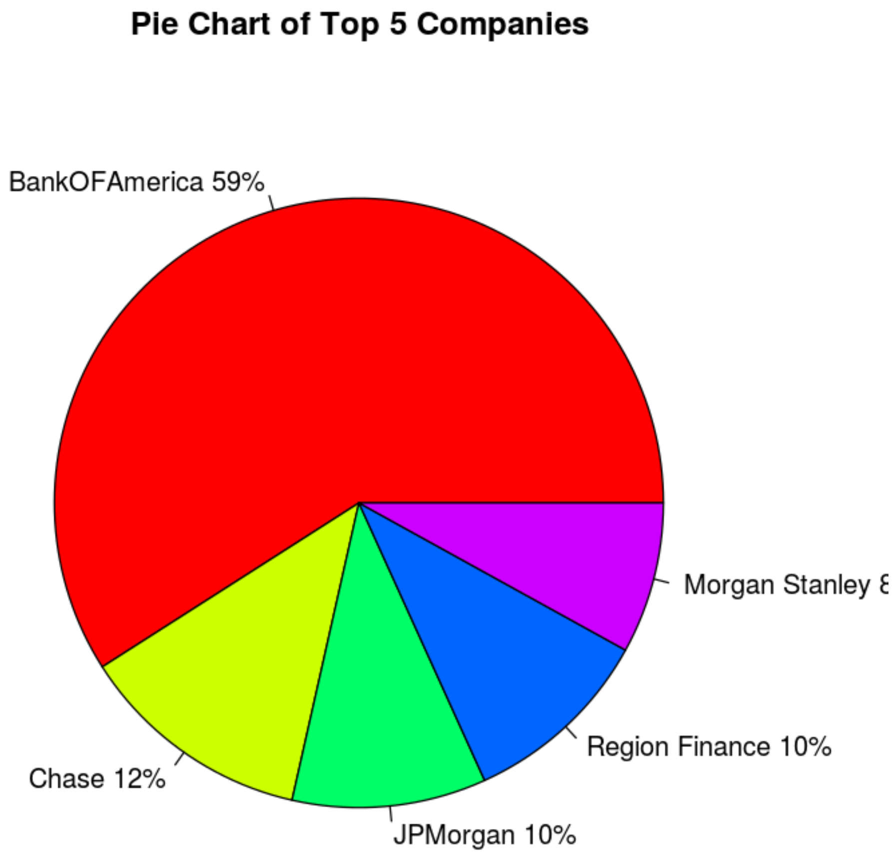
```
     days
 Length:1176
 Class :character
 Mode  :character
```

```
In [7]: # Total Stock Volume for each week day for the last five years for All the Banking & Fin
        banks$day<- factor(banks$day , levels = c('Monday', 'Tuesday', 'Wednesday', 'Thursday',

        ggplot(data = banks, aes(x= day , y= volume )) +
            geom_bar(stat= "identity", fill ='blue') +
            labs( x = 'Week Days',y = "Volume of Stock in Million") + ggtitle("Total Bank Stocks
```

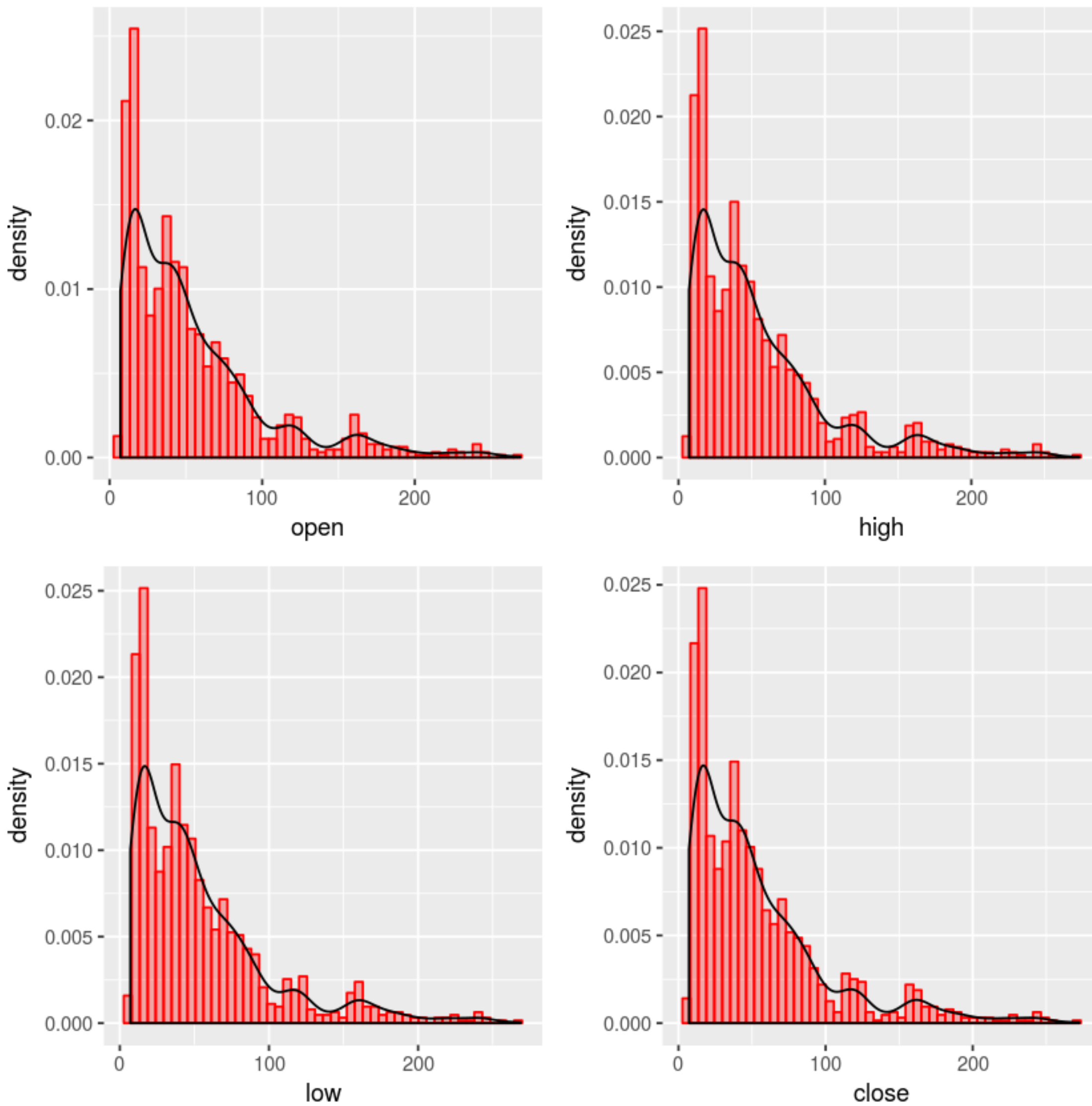


In [9]:
```
# To obtaint the total volume of stocks bought from each Bankind & Finance Company
options(warn = -1 )
Banks <- as.vector(unique(banks$Name))
tot_stocks <- c()
for (i in 1:19){
  a <- filter(banks, Name == Banks[i])
  tot <- sum(a$volume)
  tot_stocks <- c(tot_stocks, tot)
}

# Converting to Dataframe of each Banking Company and total stock volume
bank_stocks <- data.frame(Banks,tot_stocks)
# Ordering the dataframe in descending order based on the volume of stock
tot_stocks <- bank_stocks[order(-bank_stocks$tot_stock),]
five_comp <- head(tot_stocks,5)
```

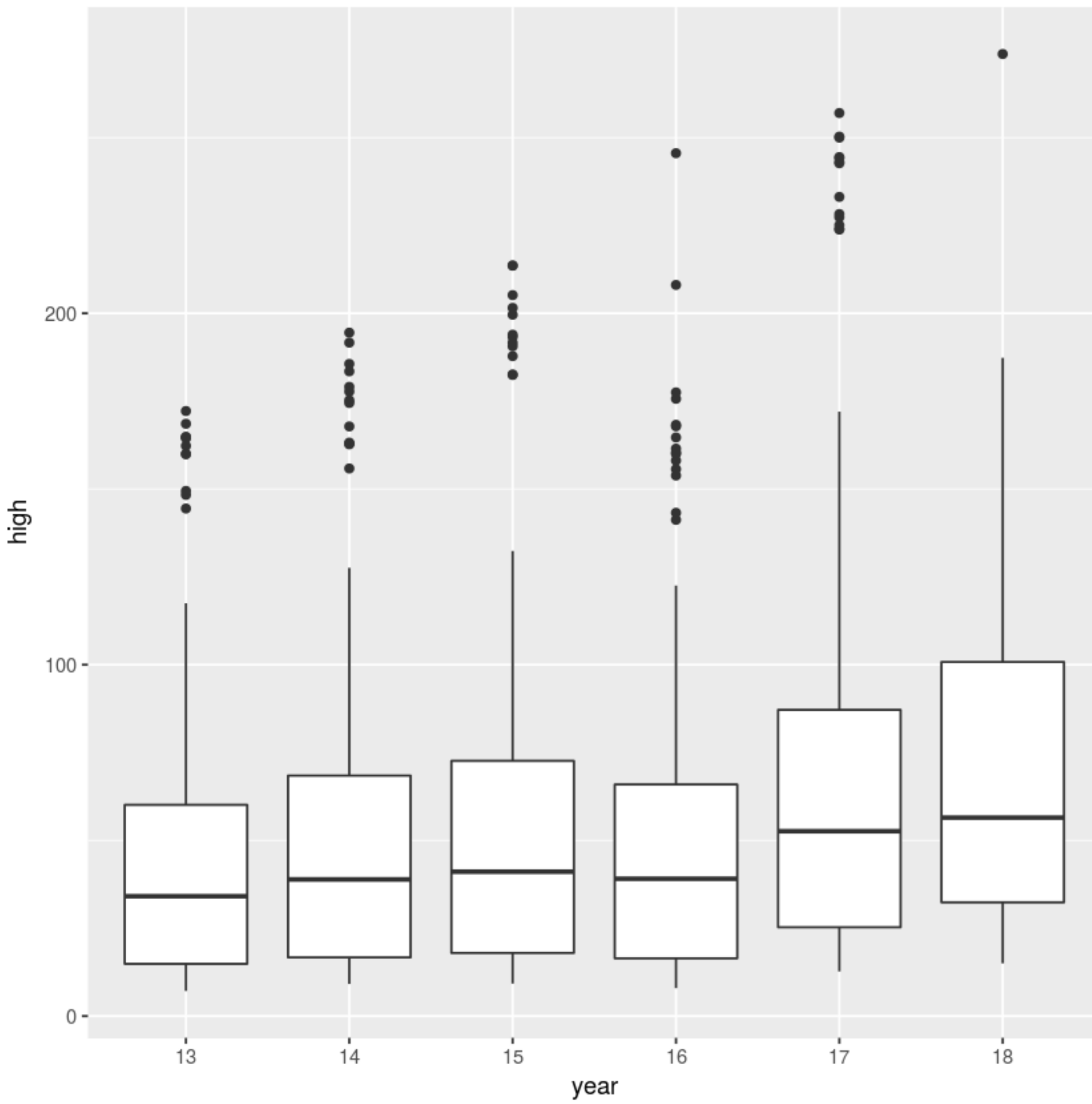
In [10]:
```
# Top 5 Banks
volume <- c(five_comp$tot_stocks)
company <- c("BankOFAmerica","Chase","JPMorgan","Region Finance","Morgan Stanley")
pct <- round(volume/sum(volume,is.na=FALSE) * 100)
company <- paste(company,pct)
company <- paste(company,"%",sep="")
pie(volume,labels = company, col=rainbow(length(company)),main="Pie Chart of Top 5 Comp
```




```
In [11]: bac <- filter(banks,Name == 'BAC')
         bac <- subset(banks, days!= '18')

In [12]: #Checking the Univariate Distribution
         p1 = ggplot(bac, aes(open)) + geom_histogram(bins = 50, aes(y = ..density..), col = "re
         p2 = ggplot(bac, aes(high)) + geom_histogram(bins = 50, aes(y = ..density..), col = "re

         p3 = ggplot(bac, aes(low)) + geom_histogram(bins = 50, aes(y = ..density..), col = "red

         p4 = ggplot(bac, aes(close)) + geom_histogram(bins = 50, aes(y = ..density..), col = "r

         grid.arrange(p1,p2,p3,p4, nrow=2,ncol=2)
```



In [13]:
```
# Plotting the Box plot for the BankofAmerica to check the Mean, Median, Mode, Q1 & Q3
bac$year <- strftime(bac$date , '%y')
ggplot(bac, aes(x = year, y= high))+ geom_boxplot()
```



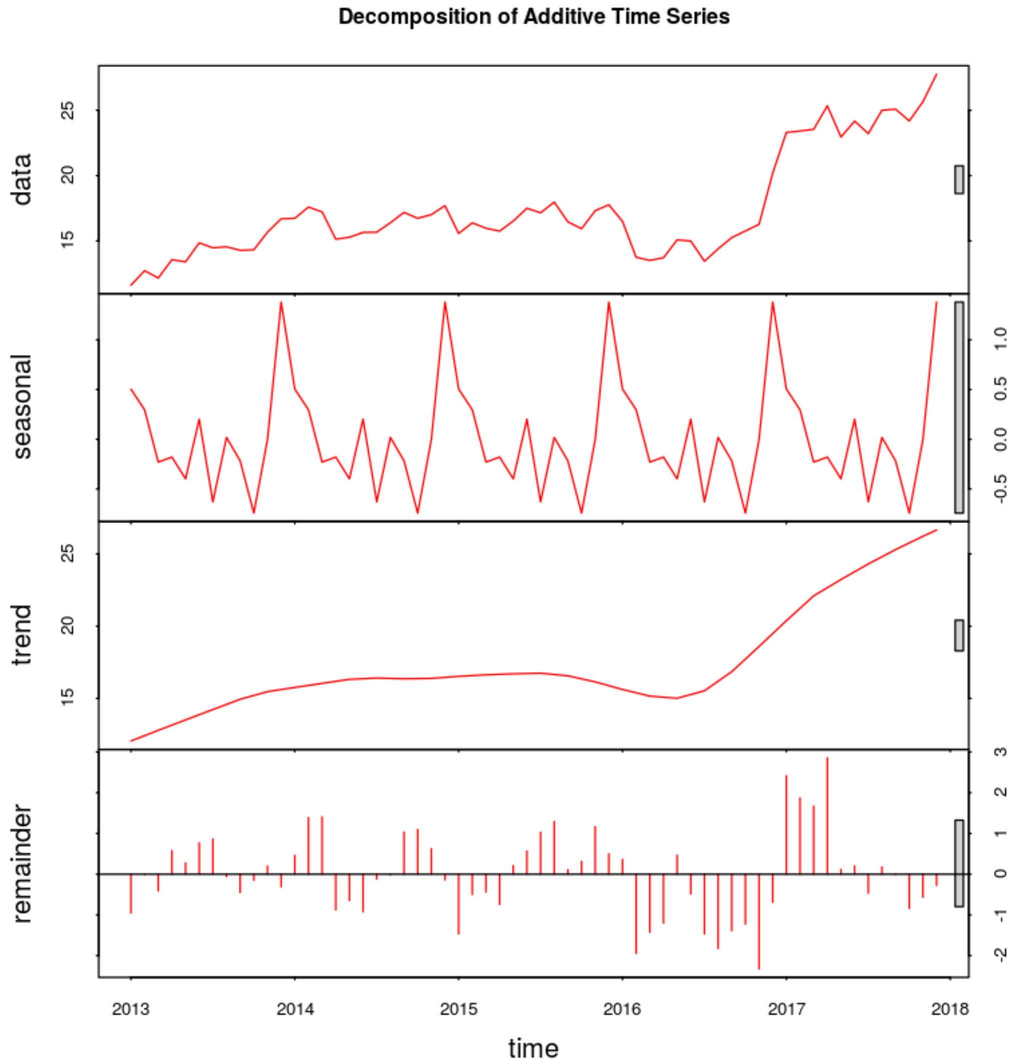

```
In [14]: summary(bac)
```

```
        date          open            high            low       
 2013-02-14:   2   Min.   :  7.07   Min.   :  7.17   Min.   :  7.05  
 2013-02-26:   2   1st Qu.: 18.36   1st Qu.: 18.50   1st Qu.: 18.23  
 2013-02-27:   2   Median : 40.55   Median : 40.88   Median : 40.19  
 2013-03-04:   2   Mean   : 53.72   Mean   : 54.16   Mean   : 53.25  
 2013-03-06:   2   3rd Qu.: 70.81   3rd Qu.: 71.74   3rd Qu.: 70.34  
 2013-03-07:   2   Max.   :269.04   Max.   :273.79   Max.   :268.81  
 (Other)   :1164                                                     
     close            volume               Name          year          
 Min.   :  7.12   Min.   :  0.3574   BAC     : 63   Length:1176       
 1st Qu.: 18.39   1st Qu.:  2.5010   BBT     : 63   Class :character  
```

```
 Median : 40.44   Median :  4.9470   BK      : 63   Mode  :character
 Mean   : 53.72   Mean   : 12.0972   C       : 63
 3rd Qu.: 71.09   3rd Qu.: 11.8393   CMA     : 63
 Max.   :272.48   Max.   :231.4992   COF     : 63
                                     (Other):798
     days                  day
 Length:1176        Monday   :228
 Class :character   Tuesday  :239
 Mode  :character   Wednesday:237
                    Thursday :240
                    Friday   :232
```

In [15]:
```
# Creating a daily date object
inds <- seq(as.Date("2012-08-13"), as.Date("2017-08-11"), by = "day")

# Creating a function for the time series
time_series <- function(col_idx){
  ## Create a time series object
    c_ts <- as.numeric(bac[,col_idx]) %>%
    tsclean(replace.missing = TRUE, lambda = NULL) %>%
    ts(start = c(2013,1) , end = c(2017,12), frequency = 12)
    #ts(start = c(2012, as.numeric(format(inds[1], "%j"))),end = 2017,frequency = 12)
    return(c_ts)
}
```

In [20]:
```
# Decompostion of the Time Series
c_ts <- time_series(which(colnames(bac)== 'high'))
x = stl(c_ts, s.window = 'periodic')
plot(x, main = "Decomposition of Additive Time Series" , col = 'red')
```

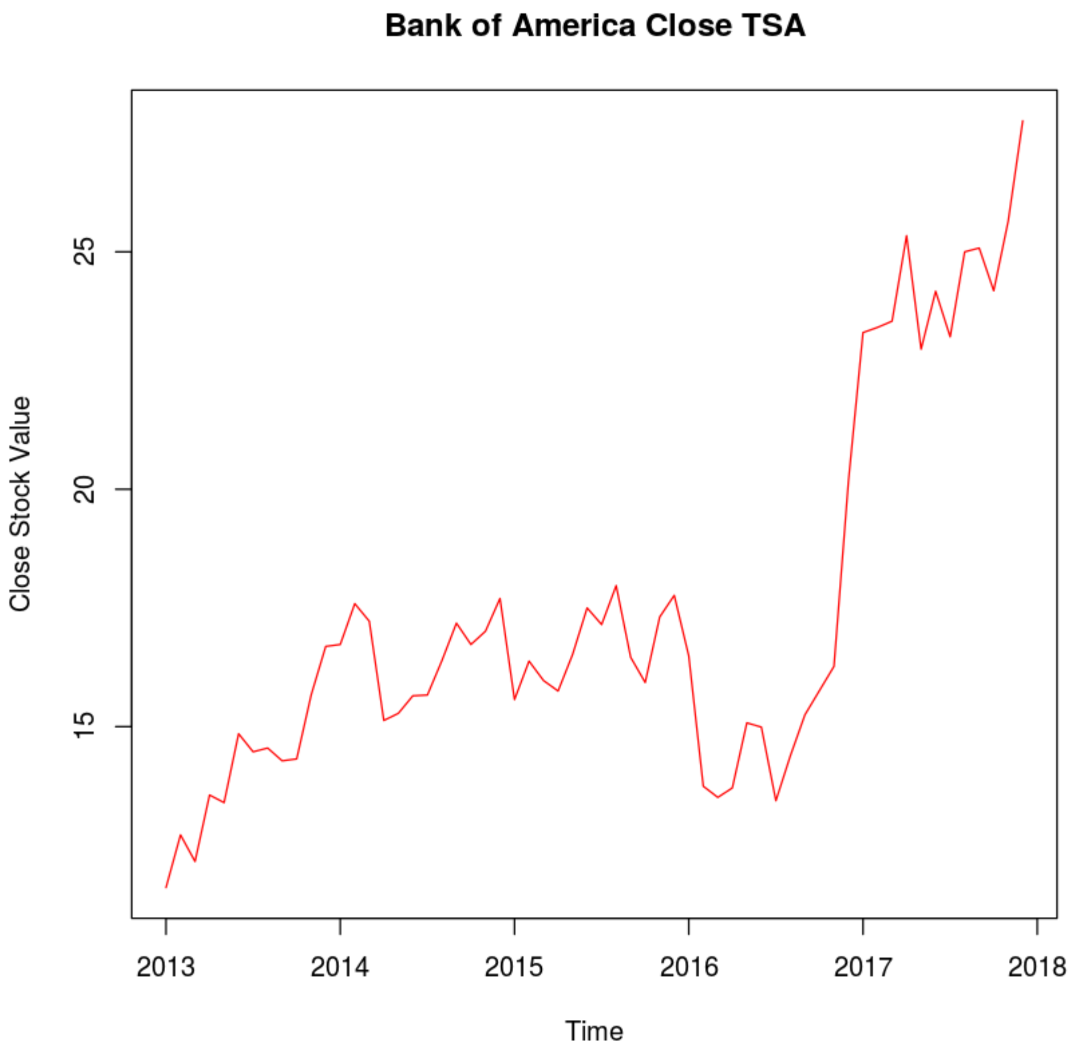


In [84]: # The Time Series Data of Bank of America Company
```
c_ts
```

```
          Jan    Feb    Mar    Apr    May    Jun    Jul    Aug    Sep    Oct
2013 11.610 12.720 12.160 13.560 13.400 14.850 14.470 14.550 14.280 14.320
2014 16.730 17.590 17.220 15.130 15.280 15.650 15.665 16.390 17.180 16.730
2015 15.570 16.380 15.970 15.750 16.520 17.500 17.150 17.970 16.460 15.930
2016 16.490 13.745 13.510 13.710 15.080 14.990 13.440 14.390 15.250 15.760
2017 23.300 23.409 23.540 25.340 22.950 24.170 23.210 25.000 25.080 24.180
          Nov    Dec
2013 15.670 16.690
2014 17.010 17.700
2015 17.310 17.765
```

```
2016 16.270 20.180
2017 25.650 27.760
```

```
In [48]: # Time series for the Bank of America Close Stock Value over the time period
         plot.ts(c_ts, xlab = "Time", ylab = "Close Stock Value", main = "Bank of America Close
```




```
In [49]: # Dickey-Fuller test for variable to check whether it is stationar or not , if the p-va
         # it means the dataset is stationary
         dick_fuller <- adf.test(c_ts, alternative = "stationary", k = 0)
         print(dick_fuller)
```

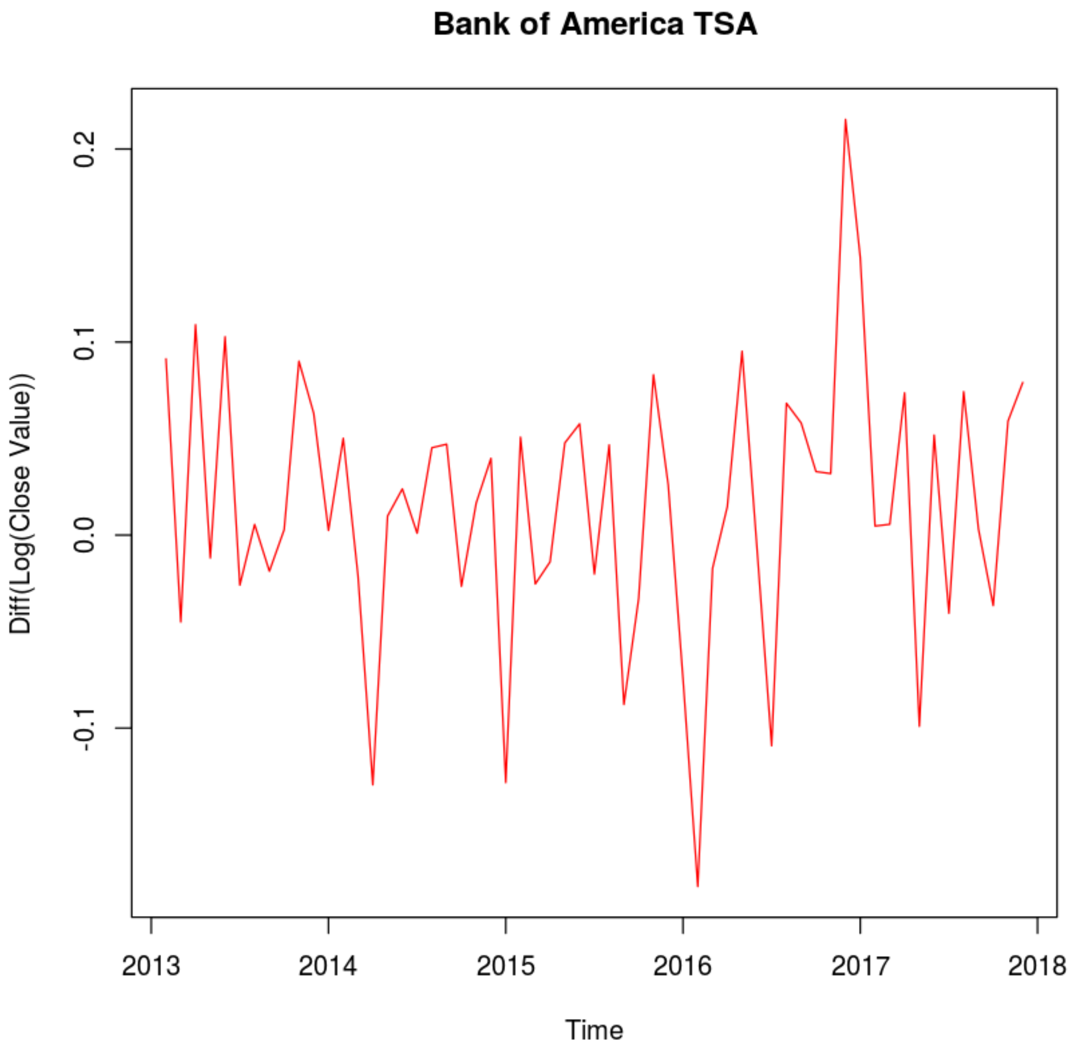
```
	Augmented Dickey-Fuller Test
```

```
data:  c_ts
Dickey-Fuller = -1.1402, Lag order = 0, p-value = 0.9086
alternative hypothesis: stationary
```

```
In [50]: # From the above p-value you can see that it is far away from the zero, therefore it is
         # Converting it to log so that the variance becomes equal over the time period
         # and Differencing is a common solution used to stationarize the mean of the variable a
         #and We will perform differencing using R function diff.
         stat_c_ts <- diff(log(c_ts))
```

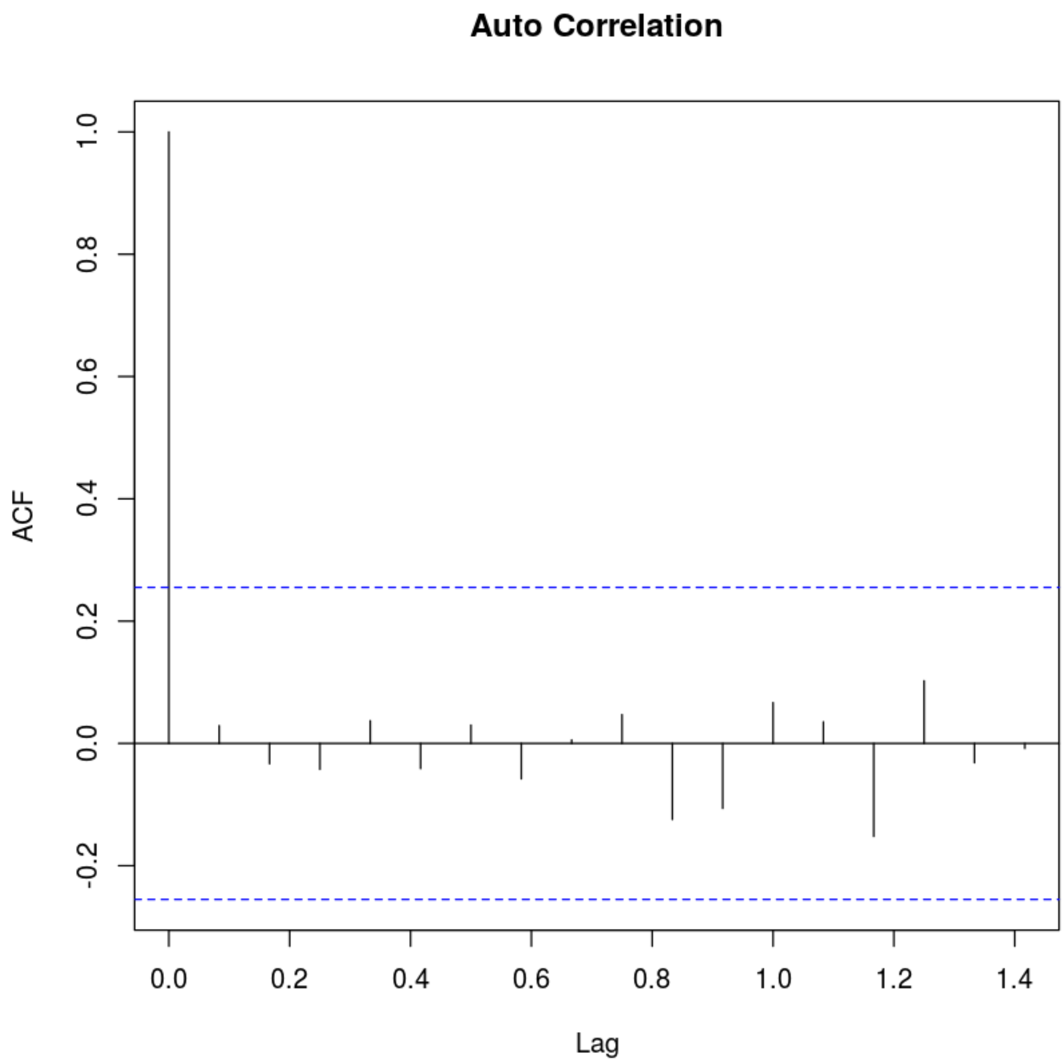
```
In [51]: plot(stat_c_ts, main = 'Bank of America TSA', ylab = 'Diff(Log(Close Value))',col = 're

         # To check if the time series is stationary after using difference function using the d
         adf.test(stat_c_ts,alternative = "stationary", k = 0)
```

```
Augmented Dickey-Fuller Test

data:  stat_c_ts
Dickey-Fuller = -7.2621, Lag order = 0, p-value = 0.01
alternative hypothesis: stationary
```



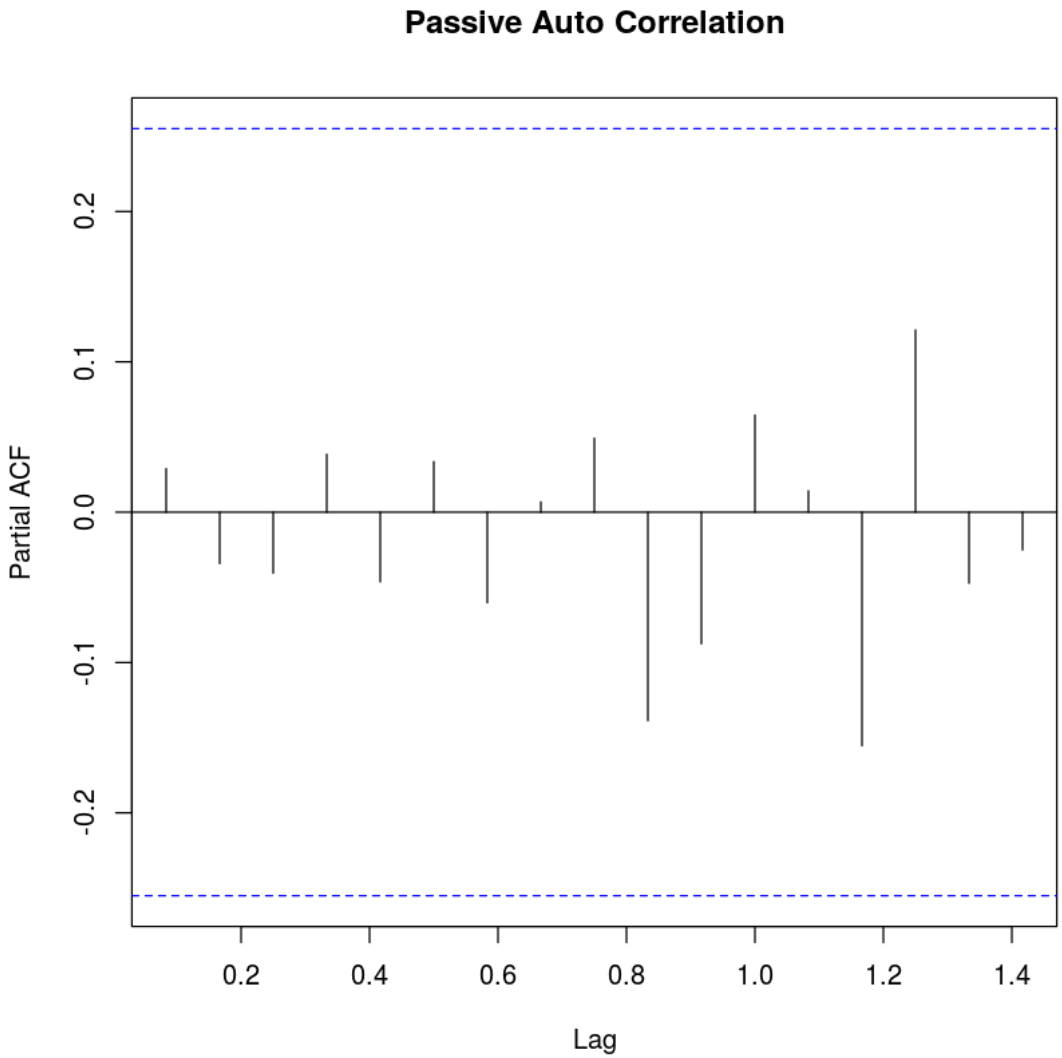
0.1.1 Now we can proceed with the ARIMA Prediction method, to do the ARIMA function we need three values, p,d,q and the values for p,q can be found using passive auto-correlation(acf) and auto-correlation(acf) function,and the d is the number of time the difference has been done to the variable

```
In [53]: # To determine the q value
         acf(stat_c_ts, main = 'Auto Correlation')
```



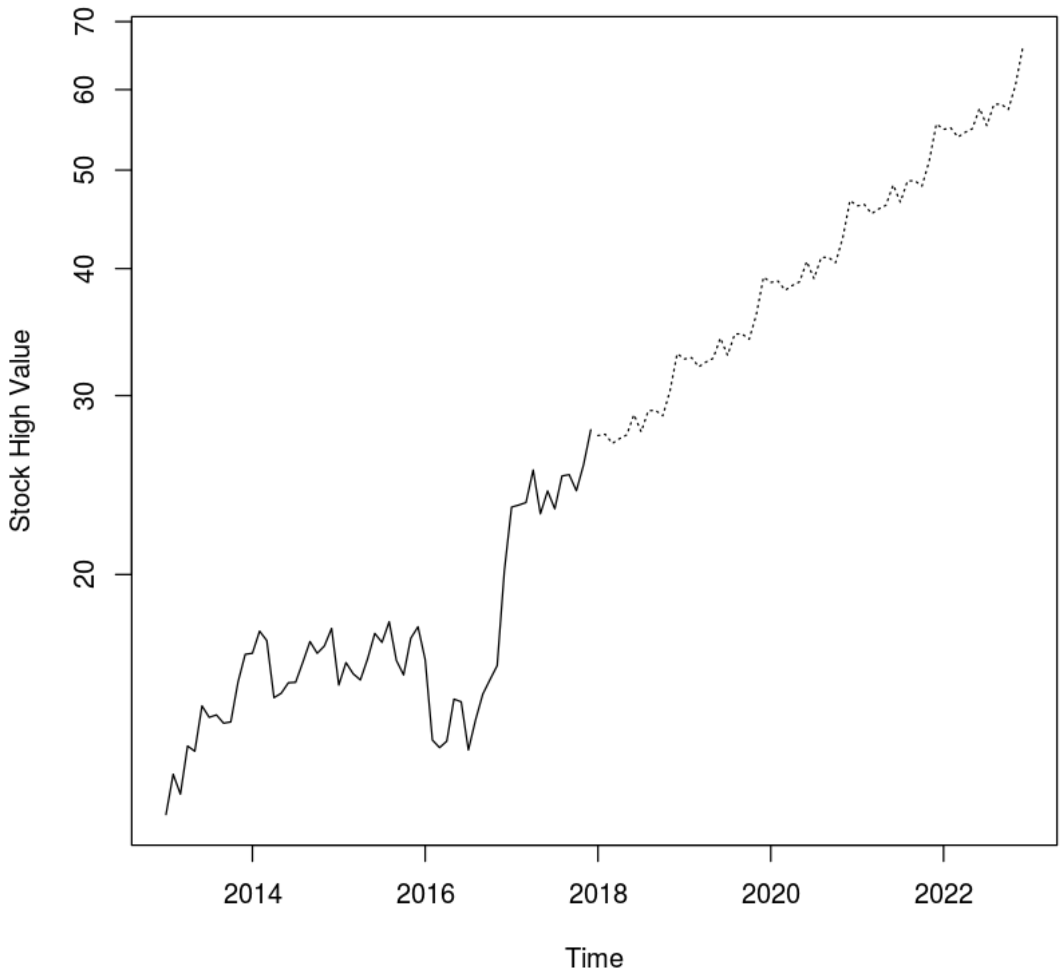
From the graph Above the q value is equal to 1

In [54]:
```
# To determain the p value
pacf(stat_c_ts, main = 'Passive Auto Correlation')
```



From the graph Above the p value is equal to 0

In [57]:
```
fit <- arima(log(c_ts), c(0,1,1), seasonal = list(order = c(0,1,1), period = 12))
pred <- predict(fit, n.ahead = 5*12)
pred1 <- 2.718^pred$pred
ts.plot(c_ts,pred1,log = "y",lty = c(1,3), ylab = "Stock High Value")
```



1 We have predicted the Company's High Stock Value for Next 5 Years

In [59]: pred1

```
          Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
2018 27.39929 27.48020 26.91841 27.21120 27.44027 28.73258 27.63466 28.99376
2019 32.59017 32.68640 32.01818 32.36644 32.63890 34.17605 32.87013 34.48671
2020 38.76447 38.87894 38.08411 38.49836 38.82244 40.65080 39.09746 41.02031
2021 46.10850 46.24466 45.29925 45.79198 46.17746 48.35221 46.50459 48.79173
2022 54.84389 55.00584 53.88132 54.46740 54.92591 57.51267 55.31501 58.03546
          Sep      Oct      Nov      Dec
2018 29.00442 28.65793 30.31241 32.98964
```

```
2019 34.49939 34.08726 36.05519 39.23962
2020 41.03540 40.54518 42.88594 46.67368
2021 48.80967 48.22658 51.01080 55.51614
2022 58.05680 57.36324 60.67495 66.03383
```

2 Accuracy Testing

We are going to take data till 2016 as a training set and we will test the predicted value of 2017 with the original high_stock value of 2017

```
In [60]: # Creating a daily date object
         inds <- seq(as.Date("2012-08-13"), as.Date("2017-08-11"), by = "day")

         # Creating a function for the time series
         test_series <- function(col_idx){
           ## Create a time series object
             c_ts <- as.numeric(bac[,col_idx]) %>%
             tsclean(replace.missing = TRUE, lambda = NULL) %>%
             ts(start = c(2013,1) , end = c(2016,12), frequency = 12)
             #ts(start = c(2012, as.numeric(format(inds[1], "%j"))),end = 2017,frequency = 12)
             return(c_ts)
         }

In [61]: test <- test_series(which(colnames(bac)== 'high'))
         test_stat <- diff(log(test))
         dick_fuller <- adf.test(test_stat, alternative = "stationary", k = 0)
         print(dick_fuller)
```

```
        Augmented Dickey-Fuller Test

data:  test_stat
Dickey-Fuller = -5.9805, Lag order = 0, p-value = 0.01
alternative hypothesis: stationary
```

```
In [86]: fit<- arima(log(test), c(0,1,0), seasonal = list(order = c(0,1,0), period = 12))
         pred <- predict(fit, n.ahead = 5*12)
         pred2 <- 2.718^pred$pred
         test_17 <- head(pred2,12)
         test_17 <- round(test_17, digits = 2)
         original_17 <- tail(c_ts,12)
         original_17 <- round(original_17,digits = 2)
```

3 Now Checking for the Accuracy

```
In [88]: accuracy <- sum(test_17)/sum(original_17)
         accuracy
```

0.707142613849246

4 Finally we where able to get an accuracy of 70.75%