project.R

Rishabh Rawat

2021-12-19

```
options(scipen = 999)

library(ggplot2)
library(quantmod)

## Warning: package 'quantmod' was built under R version 4.1.2

## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: TTR

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

library(xts)
library(rvest)
library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse
1.3.1 --

## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1
## v purrr   0.3.4

## Warning: package 'dplyr' was built under R version 4.1.2

## -- Conflicts ------------------------------------------
tidyverse_conflicts() --
## x dplyr::filter()         masks stats::filter()
## x dplyr::first()          masks xts::first()
## x readr::guess_encoding() masks rvest::guess_encoding()
```

```
## x dplyr::lag()            masks stats::lag()
## x dplyr::last()           masks xts::last()

library(PerformanceAnalytics)

##
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':
##
##     legend

library(corrplot)

## corrplot 0.90 loaded

library(GGally)

## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2

library(cluster)
library(factoextra)

## Warning: package 'factoextra' was built under R version 4.1.2

## Welcome! Want to learn more? See two factoextra-related books at
https://goo.gl/ve3WBa

library(cluster.datasets)
library(tidyverse)
library(gridExtra)

##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine

library(factoextra)
library(plotly)

##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

## The following object is masked from 'package:stats':
##
##     filter
```

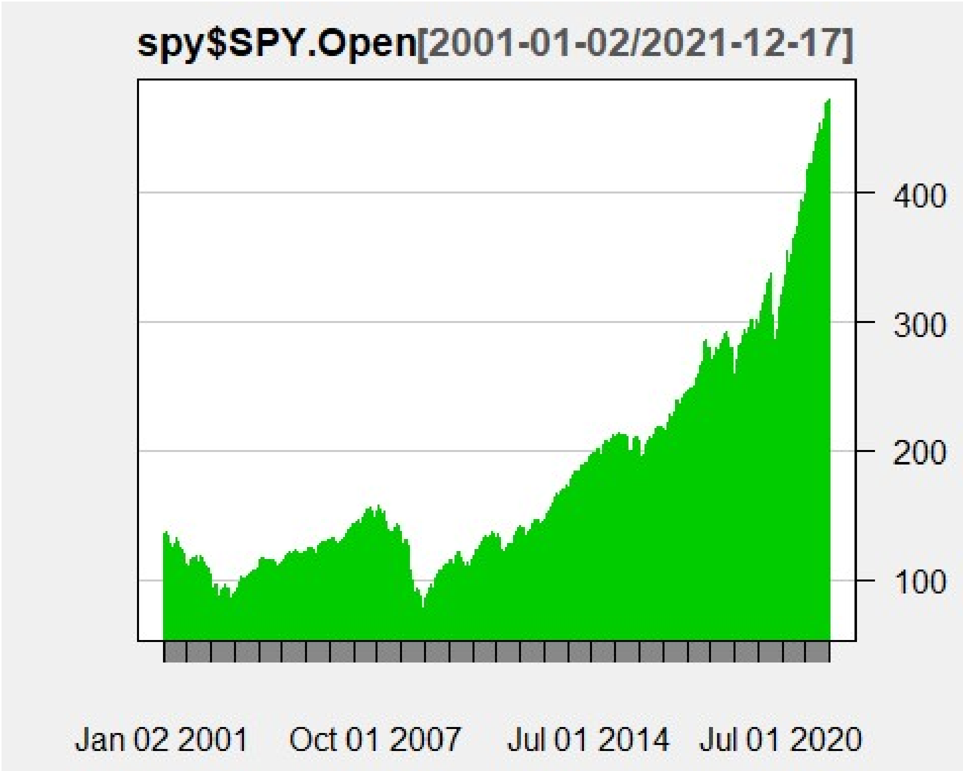
```
## The following object is masked from 'package:graphics':
##
##     layout

brk<-read.csv("BRK-A.csv")
brk<-data.frame(brk)
head(brk)

##       Date  Open  High   Low Close Adj.Close Volume
## 1 1/2/2001 71500 74600 71500 72400     72400   1230
## 2 1/3/2001 72400 73000 70000 70000     70000    680
## 3 1/4/2001 69900 70600 68300 69000     69000    650
## 4 1/5/2001 68600 68600 67500 67800     67800    270
## 5 1/8/2001 67600 68600 66200 67700     67700    550
## 6 1/9/2001 67900 69300 66500 66800     66800    390

#Exploratory Analysis
#Converting Date into appropriate format

head(brk$Date)

## [1] "1/2/2001" "1/3/2001" "1/4/2001" "1/5/2001" "1/8/2001" "1/9/2001"

strDates <- c(brk$Date)
brk$Date <- as.Date(strDates, "%m/%d/%Y")


a<-read.csv("annual.csv")
strDate <- c(a$Date)
date<- as.Date(strDate, "%m/%d/%Y")
a$MarketCap<- as.numeric(gsub("[^[:digit:]]","",a$MarketCap))

spx <- read.csv("SPY.csv")
strDates <- c(spx$Date)
spx$Date <- as.Date(strDates)

data <- read.csv("Closing prices.csv")
head(data)

##    ï..Date    BRK   Apple   BOA  Amex    KO Moodys
## 1 1/2/2015 223600 27.3325 17.90 93.02 42.14  95.89
## 2 1/5/2015 220980 26.5625 17.38 90.56 42.14  94.01
## 3 1/6/2015 220450 26.5650 16.86 88.63 42.46  93.02
## 4 1/7/2015 223480 26.9375 16.94 90.30 42.99  94.44
## 5 1/8/2015 226680 27.9725 17.29 91.58 43.51  95.36
## 6 1/9/2015 224675 28.0025 16.98 90.42 43.03  93.82

summary(brk)

##       Date                 Open             High             Low
##  Min.   :2001-01-02   Min.   : 60000   Min.   : 61700   Min.   : 59000
```

```
##  1st Qu.:2006-03-26   1st Qu.: 89986   1st Qu.: 90419   1st Qu.: 89600
##  Median :2011-06-11   Median :123898   Median :124900   Median :122985
##  Mean   :2011-06-12   Mean   :168130   Mean   :169193   Mean   :166960
##  3rd Qu.:2016-08-29   3rd Qu.:222934   3rd Qu.:223963   3rd Qu.:221827
##  Max.   :2021-11-15   Max.   :441063   Max.   :445000   Max.   :439132
##      Close          Adj.Close          Volume
##  Min.   : 61200   Min.   : 61200   Min.   :    1.0
##  1st Qu.: 90000   1st Qu.: 90000   1st Qu.:    4.0
##  Median :123932   Median :123932   Median :  210.0
##  Mean   :168110   Mean   :168110   Mean   :  355.7
##  3rd Qu.:222910   3rd Qu.:222910   3rd Qu.:  500.0
##  Max.   :439460   Max.   :439460   Max.   :27402.0
```

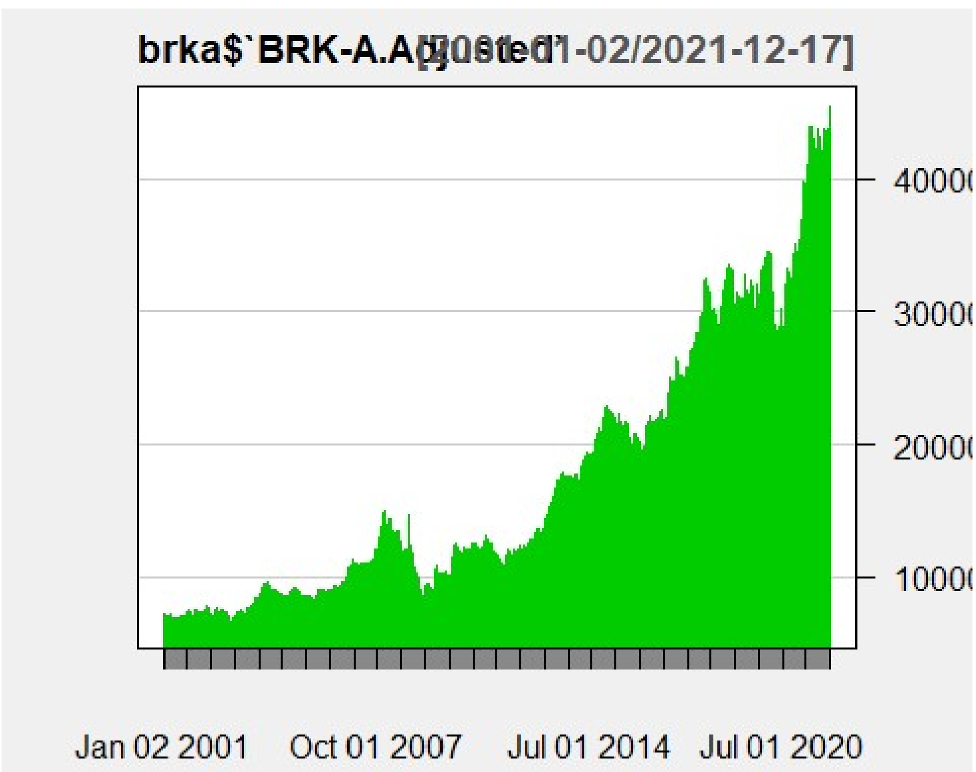
```r
# RETURN SCREENSHOT

brka <- getSymbols("BRK-A",from="2000-12-30",auto.assign = F)
```

```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.
```

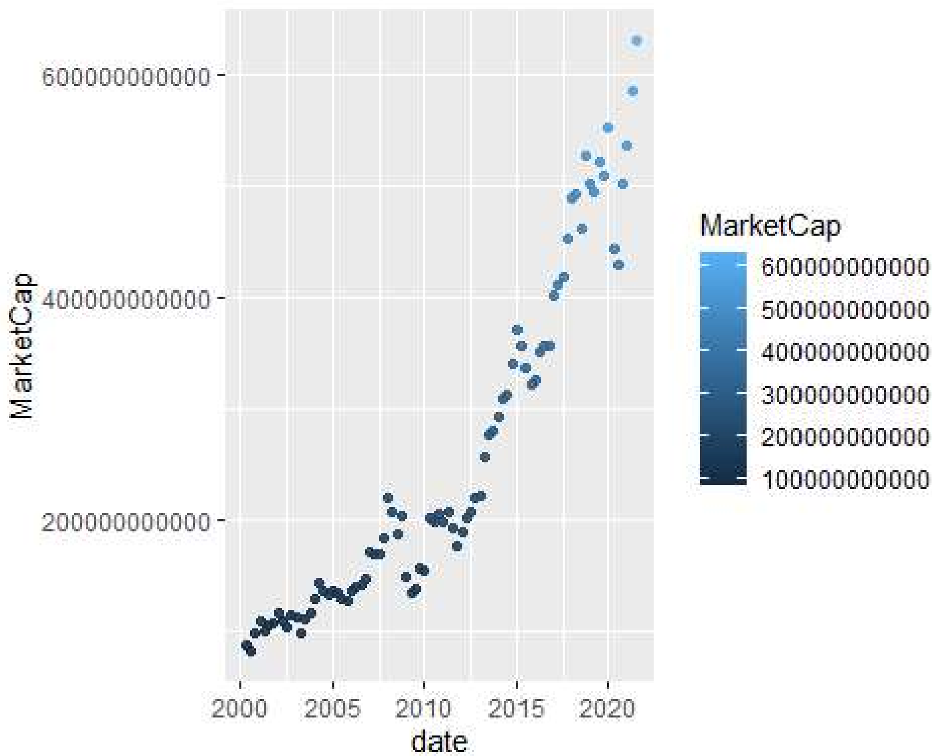
```r
spy <- getSymbols("SPY",from="2000-12-31",auto.assign = F)
lineChart(spy$SPY.Open, line.type = 'h', theme = 'white', TA = NULL)
```



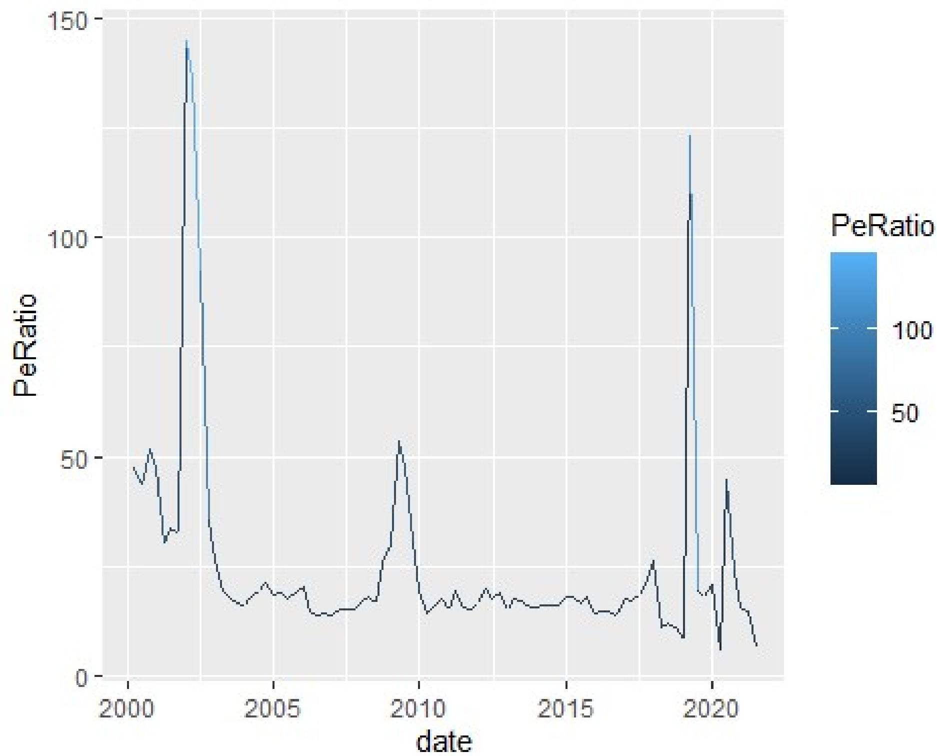

```
lineChart(brka$`BRK-A.Adjusted`, line.type = 'h', theme = 'white', TA = NULL)
```



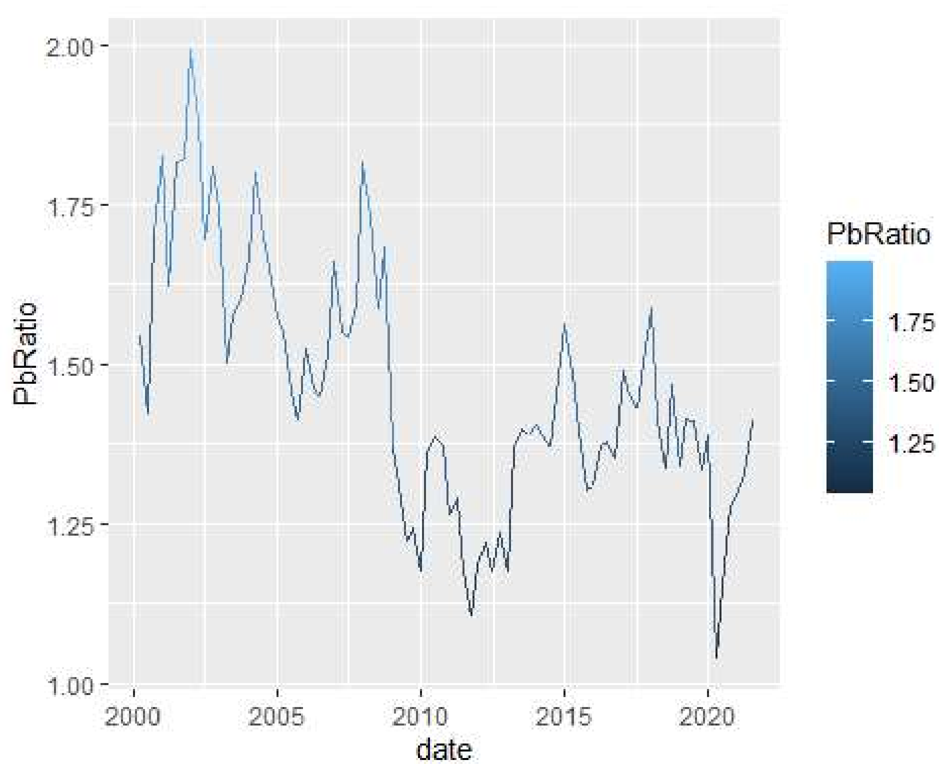
```r
#Market cap , PE ratio, PB Ratios

ggplot(data = a, mapping = aes(x = date, y = MarketCap)) +
  geom_point(alpha = 0.9, aes(color = MarketCap))
```




```r
ggplot(data = a, mapping = aes(x = date, y = PeRatio)) +
  geom_line(alpha = 0.9, aes(color = PeRatio))
```




```r
ggplot(data = a, mapping = aes(x = date, y = PbRatio)) +
  geom_line(alpha = 0.9, aes(color = PbRatio))
```

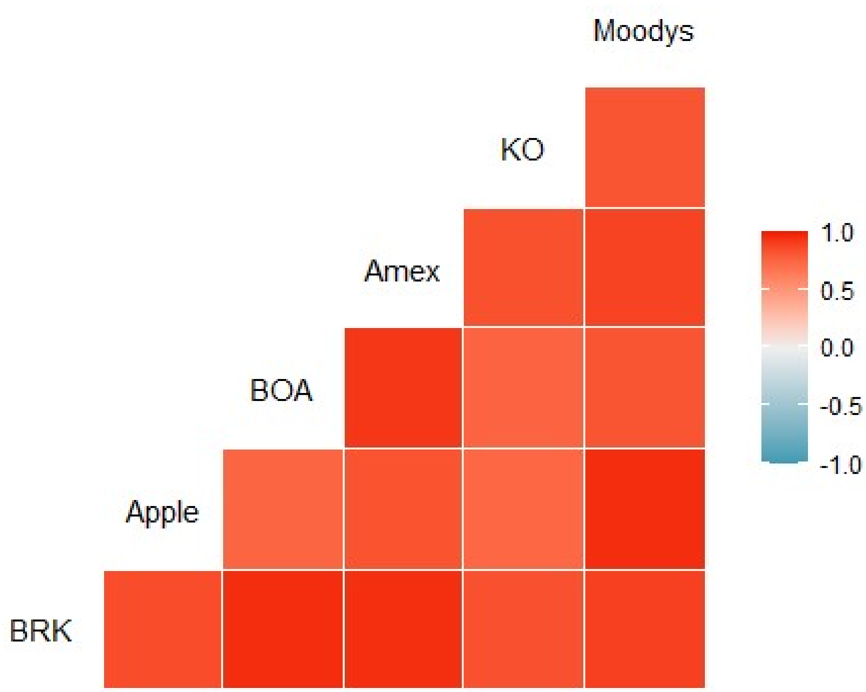


```r
ret <- read.csv("returns.csv")
head(ret)
```

```
##       spy brka Year
## 1 -10.14   27 2000
## 2 -13.04    6 2001
## 3 -23.37   -4 2002
## 4  26.38   16 2003
## 5   8.99    4 2004
## 6   3.00    1 2005
```

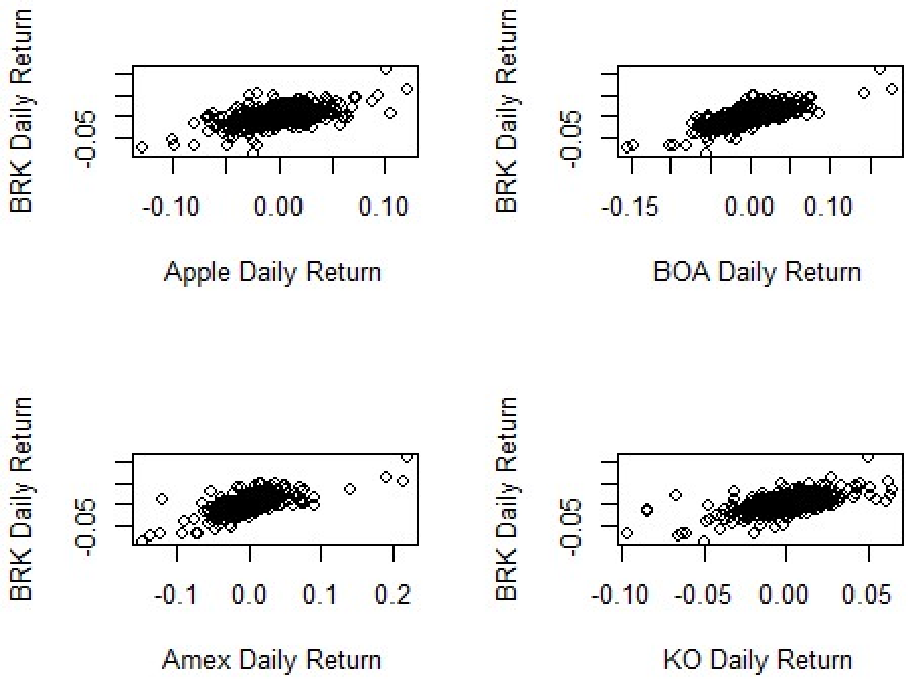
```r
g <- ggplot(ret, aes(x=Year))
g <- g + geom_line(aes(y=brka), colour="blue")
g <- g + geom_line(aes(y=spy), colour="darkgreen")



# corrplot of holdings and BRK_A [STRONG CORRELATION]

ggcorr(data)
```

```
## Warning in ggcorr(data): data in column(s) 'ï..Date' are not numeric and
were
## ignored
```




```r
#Performing linear and multiple regression
```

```r
#Data Description -
library(quantmod)
library(car)

## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode

## The following object is masked from 'package:purrr':
##
##     some

library(MASS)

##
## Attaching package: 'MASS'

## The following object is masked from 'package:plotly':
##
##     select

## The following object is masked from 'package:dplyr':
##
##     select

library(ggplot2)
data <- read.csv("dailyreturns.csv")
y <- data$BRK
x1 <- data$Apple
x2 <- data$BOA
x3 <- data$Amex
x4 <- data$KO
x5 <- data$Moodys

#scatterplots
par(mfrow=c(2,2))
plot(y ~ x1, xlab = "Apple Daily Return", ylab = "BRK Daily Return")
plot(y ~ x2, xlab = "BOA Daily Return", ylab = "BRK Daily Return")
plot(y ~ x3, xlab = "Amex Daily Return", ylab = "BRK Daily Return")
plot(y ~ x4, xlab = "KO Daily Return", ylab = "BRK Daily Return")
```




```
plot(y ~ x5, xlab = "Moodys Daily Return", ylab = "BRK Daily Return")


#Linear regression
linear.model1 <- lm(formula = y ~ x1)
abline(linear.model1, col = "red")
print(linear.model1)

##
## Call:
## lm(formula = y ~ x1)
##
## Coefficients:
## (Intercept)           x1
##  0.00007043   0.33039489

summary(linear.model1)

##
## Call:
## lm(formula = y ~ x1)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.077435 -0.005346 -0.000395  0.005326  0.079658
##
## Coefficients:
```

```
##               Estimate Std. Error t value            Pr(>|t|)
## (Intercept) 0.00007043 0.00025408   0.277               0.782
## x1          0.33039489 0.01386886  23.823 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01055 on 1728 degrees of freedom
## Multiple R-squared:  0.2472, Adjusted R-squared:  0.2468
## F-statistic: 567.5 on 1 and 1728 DF,  p-value: < 0.00000000000000022
```

```
linear.model2 <- lm(formula = y ~ x2)
abline(linear.model2, col = "red")
print(linear.model2)
```

```
##
## Call:
## lm(formula = y ~ x2)
##
## Coefficients:
## (Intercept)           x2
##   0.0001192    0.4313438
```

```
summary(linear.model2)
```

```
##
## Call:
## lm(formula = y ~ x2)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.062219 -0.004749 -0.000061  0.004353  0.049144
##
## Coefficients:
##              Estimate Std. Error t value            Pr(>|t|)
## (Intercept) 0.0001192  0.0002002   0.596               0.552
## x2          0.4313438  0.0097419  44.277 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.008321 on 1728 degrees of freedom
## Multiple R-squared:  0.5315, Adjusted R-squared:  0.5312
## F-statistic:  1960 on 1 and 1728 DF,  p-value: < 0.00000000000000022
```

```
linear.model3 <- lm(formula = y ~ x3)
abline(linear.model3, col = "red")
print(linear.model3)
```

```
##
## Call:
## lm(formula = y ~ x3)
##
```

```
## Coefficients:
## (Intercept)           x3
##   0.0002055    0.4189309

summary(linear.model3)

##
## Call:
## lm(formula = y ~ x3)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.046452 -0.004716 -0.000209  0.004229  0.060734
##
## Coefficients:
##              Estimate Std. Error t value            Pr(>|t|)
## (Intercept) 0.0002055  0.0002130   0.965               0.335
## x3          0.4189309  0.0107161  39.094 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.008856 on 1728 degrees of freedom
## Multiple R-squared:  0.4693, Adjusted R-squared:  0.469
## F-statistic:  1528 on 1 and 1728 DF,  p-value: < 0.00000000000000022

linear.model4 <- lm(formula = y ~ x4)
abline(linear.model4, col = "red")
print(linear.model4)

##
## Call:
## lm(formula = y ~ x4)
##
## Coefficients:
## (Intercept)           x4
##    0.000309     0.590927

summary(linear.model4)

##
## Call:
## lm(formula = y ~ x4)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.056579 -0.005365 -0.000194  0.004959  0.082831
##
## Coefficients:
##              Estimate Std. Error t value            Pr(>|t|)
## (Intercept) 0.0003090  0.0002398   1.289               0.198
## x4          0.5909275  0.0203963  28.972 <0.0000000000000002 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.009973 on 1728 degrees of freedom
## Multiple R-squared:  0.3269, Adjusted R-squared:  0.3266
## F-statistic: 839.4 on 1 and 1728 DF,  p-value: < 0.00000000000000022
```

```
linear.model5 <- lm(formula = y ~ x5)
abline(linear.model5, col = "red")
print(linear.model5)
```

```
##
## Call:
## lm(formula = y ~ x5)
##
## Coefficients:
## (Intercept)           x5
##  0.00002002   0.44745545
```

```
summary(linear.model5)
```

```
##
## Call:
## lm(formula = y ~ x5)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.041282 -0.004897 -0.000447  0.004505  0.076603
##
## Coefficients:
##               Estimate Std. Error t value            Pr(>|t|)
## (Intercept) 0.00002002 0.00022515   0.089               0.929
## x5          0.44745545 0.01295553  34.538 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.00935 on 1728 degrees of freedom
## Multiple R-squared:  0.4084, Adjusted R-squared:  0.4081
## F-statistic:  1193 on 1 and 1728 DF,  p-value: < 0.00000000000000022
```

```
#Multiple regression analysis
multiple.model <- lm(formula = y ~ x1+x2+x3+x4+x5)
abline(multiple.model, col = "red")
```

```
## Warning in abline(multiple.model, col = "red"): only using the first two
of 6
## regression coefficients
```

```
print(multiple.model)
```

```
##
## Call:
```

```
## lm(formula = y ~ x1 + x2 + x3 + x4 + x5)
##
## Coefficients:
## (Intercept)           x1           x2           x3           x4
x5
## -0.00002084   0.06569813   0.22102775   0.10852446   0.20245395
0.11824603
```

```
summary(multiple.model)
```

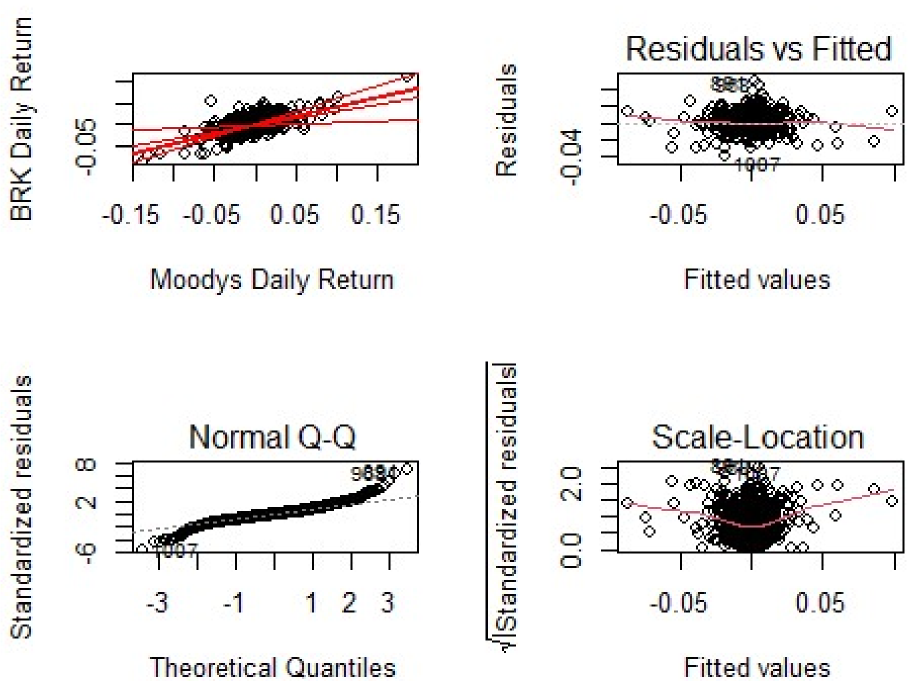
```
##
## Call:
## lm(formula = y ~ x1 + x2 + x3 + x4 + x5)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.038267 -0.003972 -0.000169  0.003386  0.047938
##
## Coefficients:
##                Estimate  Std. Error t value             Pr(>|t|)
## (Intercept) -0.00002084  0.00017027  -0.122                0.903
## x1           0.06569813  0.01138029   5.773 0.000000009217680434 ***
## x2           0.22102775  0.01273726  17.353 < 0.0000000000000002 ***
## x3           0.10852446  0.01337851   8.112 0.000000000000000937 ***
## x4           0.20245395  0.01766502  11.461 < 0.0000000000000002 ***
## x5           0.11824603  0.01394717   8.478 < 0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.007065 on 1724 degrees of freedom
## Multiple R-squared:  0.663,  Adjusted R-squared:  0.6621
## F-statistic: 678.5 on 5 and 1724 DF,  p-value: < 0.00000000000000022
```

```
#annova test
anova(multiple.model, test="Chisq")
```

```
## Analysis of Variance Table
##
## Response: y
##             Df   Sum Sq  Mean Sq  F value                Pr(>F)
## x1           1 0.063134 0.063134 1264.899 < 0.00000000000000022 ***
## x2           1 0.083096 0.083096 1664.832 < 0.00000000000000022 ***
## x3           1 0.009880 0.009880  197.939 < 0.00000000000000022 ***
## x4           1 0.009619 0.009619  192.715 < 0.00000000000000022 ***
## x5           1 0.003588 0.003588   71.879 < 0.00000000000000022 ***
## Residuals 1724 0.086049 0.000050
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#ANOVA helps you find out whether the differences between groups of data are
statistically significant.
```

```r
#Model Diagnostic Comparison

plot(multiple.model)
```

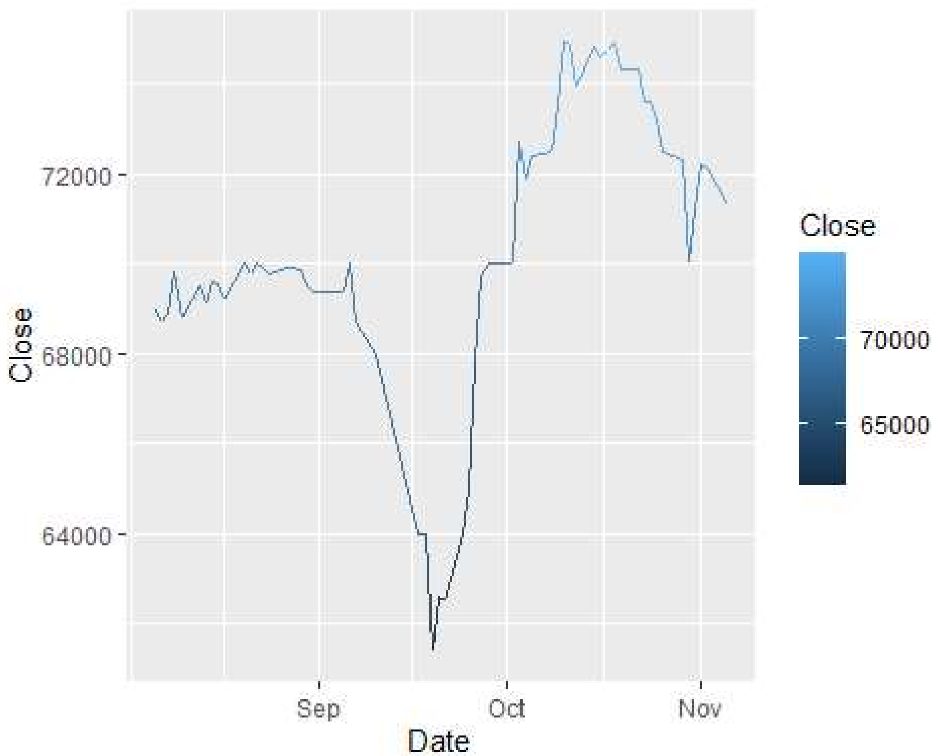



```r
#finding the best model
full.model.formula <- y ~ x1+x2+x3+x4+x5
full.model <- lm(y ~ x1+x2+x3+x4+x5, data = data)
null.model <- lm(y ~ 1, data = data)
step(null.model, full.model.formula, direction = "both")

## Start:  AIC=-15258.22
## y ~ 1
##
##        Df Sum of Sq     RSS    AIC
## + x2    1  0.135731 0.11964 -16568
## + x3    1  0.119853 0.13551 -16352
## + x5    1  0.104290 0.15108 -16164
## + x4    1  0.083490 0.17188 -15941
## + x1    1  0.063134 0.19223 -15748
## <none>              0.25537 -15258
##
## Step:  AIC=-16567.99
## y ~ x2
##
##        Df Sum of Sq      RSS    AIC
## + x4    1  0.019980 0.099655 -16882
```

```
## + x5    1  0.018654 0.100981 -16859
## + x3    1  0.012566 0.107069 -16758
## + x1    1  0.010500 0.109136 -16725
## <none>              0.119635 -16568
## - x2    1  0.135731 0.255366 -15258
##
## Step:  AIC=-16882.11
## y ~ x2 + x4
##
##        Df Sum of Sq      RSS    AIC
## + x5    1  0.008656 0.090999 -17037
## + x3    1  0.005492 0.094163 -16978
## + x1    1  0.005243 0.094412 -16974
## <none>              0.099655 -16882
## - x4    1  0.019980 0.119635 -16568
## - x2    1  0.072221 0.171876 -15941
##
## Step:  AIC=-17037.31
## y ~ x2 + x4 + x5
##
##        Df Sum of Sq      RSS    AIC
## + x3    1  0.003286 0.087713 -17099
## + x1    1  0.001665 0.089334 -17067
## <none>              0.090999 -17037
## - x5    1  0.008656 0.099655 -16882
## - x4    1  0.009982 0.100981 -16859
## - x2    1  0.039147 0.130146 -16420
##
## Step:  AIC=-17098.95
## y ~ x2 + x4 + x5 + x3
##
##        Df Sum of Sq      RSS    AIC
## + x1    1 0.0016634 0.086049 -17130
## <none>              0.087713 -17099
## - x3    1 0.0032864 0.090999 -17037
## - x5    1 0.0064506 0.094163 -16978
## - x4    1 0.0072358 0.094949 -16964
## - x2    1 0.0164518 0.104165 -16804
##
## Step:  AIC=-17130.07
## y ~ x2 + x4 + x5 + x3 + x1
##
##        Df Sum of Sq      RSS    AIC
## <none>              0.086049 -17130
## - x1    1 0.0016634 0.087713 -17099
## - x3    1 0.0032844 0.089334 -17067
## - x5    1 0.0035877 0.089637 -17061
## - x4    1 0.0065559 0.092605 -17005
## - x2    1 0.0150298 0.101079 -16854
```

```
##
## Call:
## lm(formula = y ~ x2 + x4 + x5 + x3 + x1, data = data)
##
## Coefficients:
## (Intercept)           x2           x4           x5           x3
x1
## -0.00002084   0.22102775   0.20245395   0.11824603   0.10852446
0.06569813

# From the multiple regression performed, we see that the model has p value
which is most significant.
#We can consider a linear model to be statistically significant only when
both these p-Values are less
#that the pre-determined statistical significance level, which is ideally
0.05
#Residual standard error - 10310 on 1725 degrees of freedom
#Multiple R squared : Higher the better and should be close to 1
#F-Statistic: Higher the better and Standard error closer to zero the better


#VOULME VISUALIZATION

ggplot(data = brk, mapping = aes(x = Close, y = Volume)) +
  geom_point(alpha = 0.2, aes(color = Volume))

#CURRENT VOLUME
ggplot(data = brk[5200:5250,], mapping = aes(x = Date, y = Volume)) +
  geom_point(alpha = 0.2, aes(color = Volume))

#DURING 2008 CRISIS
#High volume due and high volatility.
ggplot(data = brk[1500:2200,], mapping = aes(x = Date, y = Volume)) +
  geom_line(alpha = 0.2, aes(color = Volume))

# Date vs Close during 2008
#Stock halved

ggplot(data =brk[1700:2100,] , mapping = aes(x = Date, y = Close)) +
  geom_line(alpha = 0.9, aes(color = Close))

# Date vs Close during 9/11/2001

#High volume due and high volatility.
ggplot(data = brk[150:210,], mapping = aes(x = Date, y = Volume)) +
  geom_line(alpha = 0.9, aes(color = Volume))

# V shaped recovery
ggplot(data = brk[150:210,], mapping = aes(x = Date, y = Close)) +
```

```
  geom_line(alpha = 0.9, aes(color = Close))


chartSeries(brka,
            type = c("auto", "matchsticks"),
            subset = '2016-01::',
            show.grid = TRUE,
            major.ticks='auto', minor.ticks=TRUE,
            multi.col = FALSE,
            TA=c(addMACD(),addVo(),addSMA(n=250,col =
'blue'),addSMA(n=100,col = 'red'),addSMA(n=20,col = 'green'),
                 addROC(n=250,col = 'blue'),addROC(n=100,col =
'red'),addROC(n=20,col = 'green'))) # rate of change
```

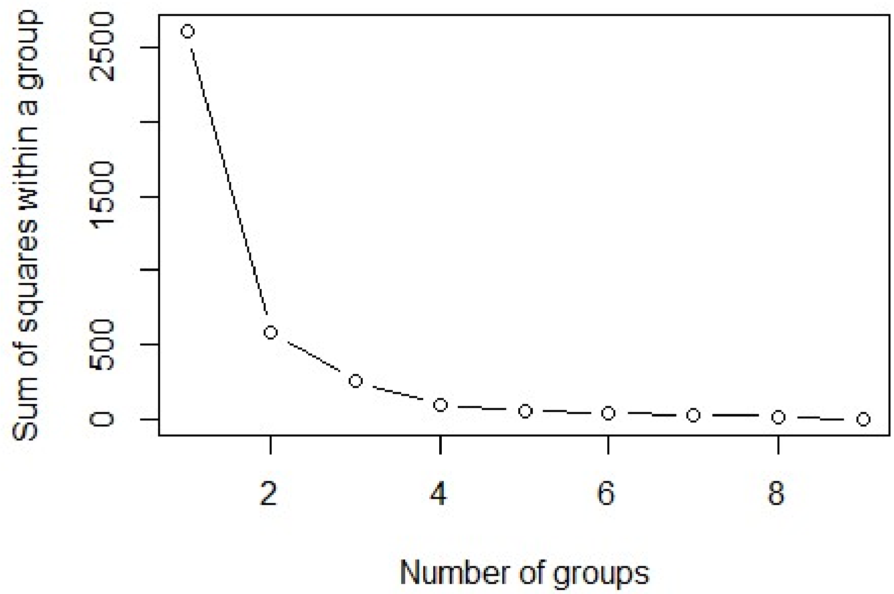



```
#Measuring crisis using bollinger bands

chartSeries(brka, theme="white",
            TA="addVo();addBBands();addCCI()", subset = '2001-08::2001-12')
```


```
chartSeries(brka, theme="white",
            TA="addVo();addBBands();addCCI()", subset = '2007-01::2009-05')
```

```
chartSeries(brka, theme="white",
            TA="addVo();addBBands();addCCI()", subset = '2020-01::2020-06')
```


```
#K Means Clustering
data <- read.csv("clustering.csv")
# Let's remove the column with the stock names, so it won't be used in the
clustering

input<-data[1:10,2:5]

# A function to check the best K value
wssplot <- function(data, nc, seed=123){
  wss <- (nrow(data)-1)*sum(apply(data,2,var))
  for (i in 2:nc){
    set.seed(seed)
    wss[i] <- sum(kmeans(data, centers=i)$withinss)}
  plot(1:nc, wss, type="b", xlab="Number of groups",
       ylab="Sum of squares within a group")}

wssplot(input, nc =9)
```

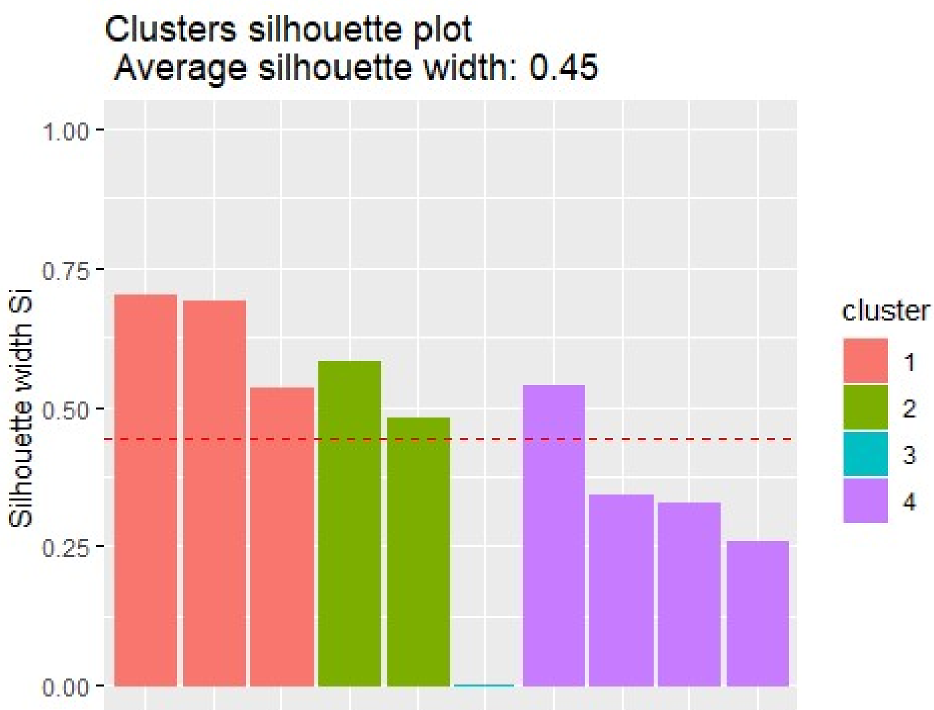



```
set.seed(123)
clustering <- kmeans(input, centers = 4, nstart = 20)
clustering
```

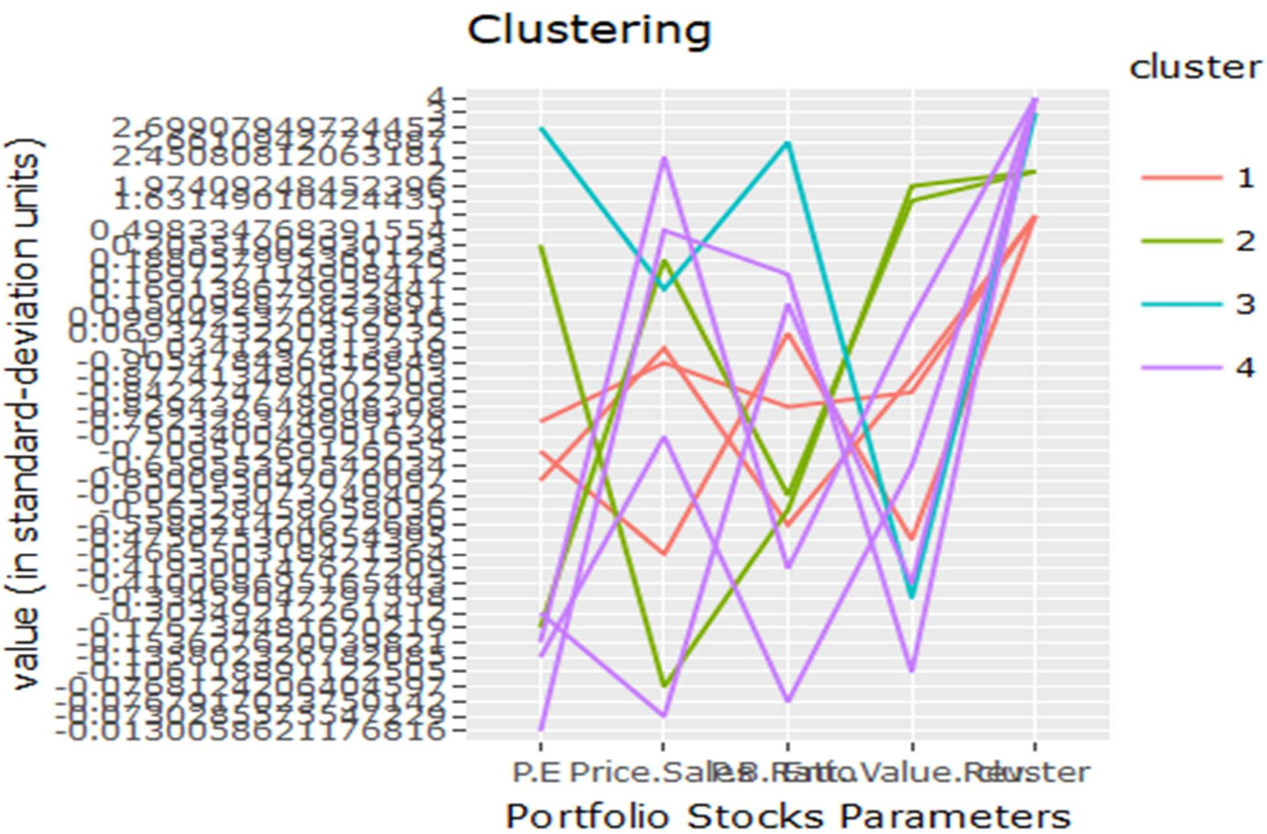
```
## K-means clustering with 4 clusters of sizes 3, 2, 1, 4
##
## Cluster means:
##         P.E Price.Sales P.B.Ratio Ent..Value.Rev.
## 1  6.543333   0.9333333  2.066667          1.0800
## 2 17.475000   3.2000000  1.410000         15.5050
## 3 58.140000   3.5000000 16.280000          3.3400
## 4 14.962500   4.4575000  3.880000          3.7125
##
## Clustering vector:
##  1  2  3  4  5  6  7  8  9 10
##  4  4  1  1  2  2  4  1  3  4
##
## Within cluster sum of squares by cluster:
## [1] 14.83660 18.92450  0.00000 65.10022
##  (between_SS / total_SS =  96.2 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
"tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
sil <- silhouette(clustering$cluster, dist(input))
fviz_silhouette(sil)
```

```
##   cluster size ave.sil.width
## 1       1    3          0.64
## 2       2    2          0.53
## 3       3    1          0.00
## 4       4    4          0.37
```




```
input$cluster <- as.factor(clustering$cluster)

p <- ggparcoord(data = input, groupColumn = "cluster", scale = "std") +
  labs(x = "Portfolio Stocks Parameters", y = "value (in standard-deviation
units)", title = "Clustering")
ggplotly(p)
```


Clustering
cluster
1
2
3
4
value (in standard-deviation units)
Portfolio Stocks Parameters
cluster