Sharma_Kartik_Assignment1

Kartik Sharma

August 7, 2015

Importing the required libraries

```
library (ggplot2)
library(mosaic)

## Loading required package: car
## Loading required package: dplyr
##
## Attaching package: 'dplyr'
##
## 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: lattice
## Loading required package: mosaicData
##
## Attaching package: 'mosaic'
##
## The following objects are masked from 'package:dplyr':
##
##     count, do, tally
##
## The following object is masked from 'package:car':
##
##     logit
##
## The following objects are masked from 'package:stats':
##
##     binom.test, cor, cov, D, fivenum, IQR, median, prop.test,
##     quantile, sd, t.test, var
##
## The following objects are masked from 'package:base':
##
##     max, mean, min, prod, range, sample, sum

library(foreach)
library(fImport)
```

```
## Loading required package: timeDate
## Loading required package: timeSeries
```

Answer 1

Reading the data file

```
georgia=
read.csv("https://raw.githubusercontent.com/jgscott/STA380/master/data/georgi
a2000.csv",header = T)
head(georgia)
```

```
##     county ballots votes   equip poor urban atlanta perAA gore bush
## 1   APPLING    6617  6099   LEVER    1     0       0 0.182 2093 3940
## 2 ATKINSON    2149  2071   LEVER    1     0       0 0.230  821 1228
## 3    BACON     3347  2995   LEVER    1     0       0 0.131  956 2010
## 4    BAKER     1607  1519 OPTICAL    1     0       0 0.476  893  615
## 5  BALDWIN    12785 12126   LEVER    0     0       0 0.359 5893 6041
## 6    BANKS     4773  4533   LEVER    0     0       0 0.024 1220 3202
```

```
summary(georgia)
```

```
##       county       ballots           votes            equip
##  APPLING :  1   Min.   :   881   Min.   :   832   LEVER  :74
##  ATKINSON:  1   1st Qu.:  3694   1st Qu.:  3506   OPTICAL:66
##  BACON   :  1   Median :  6712   Median :  6299   PAPER  : 2
##  BAKER   :  1   Mean   : 16927   Mean   : 16331   PUNCH  :17
##  BALDWIN :  1   3rd Qu.: 12251   3rd Qu.: 11846
##  BANKS   :  1   Max.   :280975   Max.   :263211
##  (Other) :153
##       poor            urban            atlanta            perAA
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.1115
##  Median :0.0000   Median :0.0000   Median :0.00000   Median :0.2330
##  Mean   :0.4528   Mean   :0.2642   Mean   :0.09434   Mean   :0.2430
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:0.3480
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.00000   Max.   :0.7650
##
##       gore             bush
##  Min.   :   249   Min.   :   271
##  1st Qu.:  1386   1st Qu.:  1804
##  Median :  2326   Median :  3597
##  Mean   :  7020   Mean   :  8929
##  3rd Qu.:  4430   3rd Qu.:  7468
##  Max.   :154509   Max.   :140494
##
```

```
attach(georgia)
```

Creating indicators for counties having vote undercount

```
georgia$undercount=ifelse(georgia$ballots>georgia$votes,1,0)
```

Finding out the undercount counties on the basis of different equipments

```
xtabs(~equip+undercount,data=georgia)

##          undercount
## equip      0  1
##   LEVER    2 72
##   OPTICAL  0 66
##   PAPER    0  2
##   PUNCH    0 17
```

From the above table we can clearly see that Lever has least reported instances of undercounts

All other equipments have 100% undercounts.

Although Lever has highest efficiency, but its success rate is still too low.

In order to find out the efficiency of the equipment, we can use the percentage of undercount votes as the parameter.

Aggregating the counts of ballots and votes on the basis of equipment and merging them to form a new dataframe

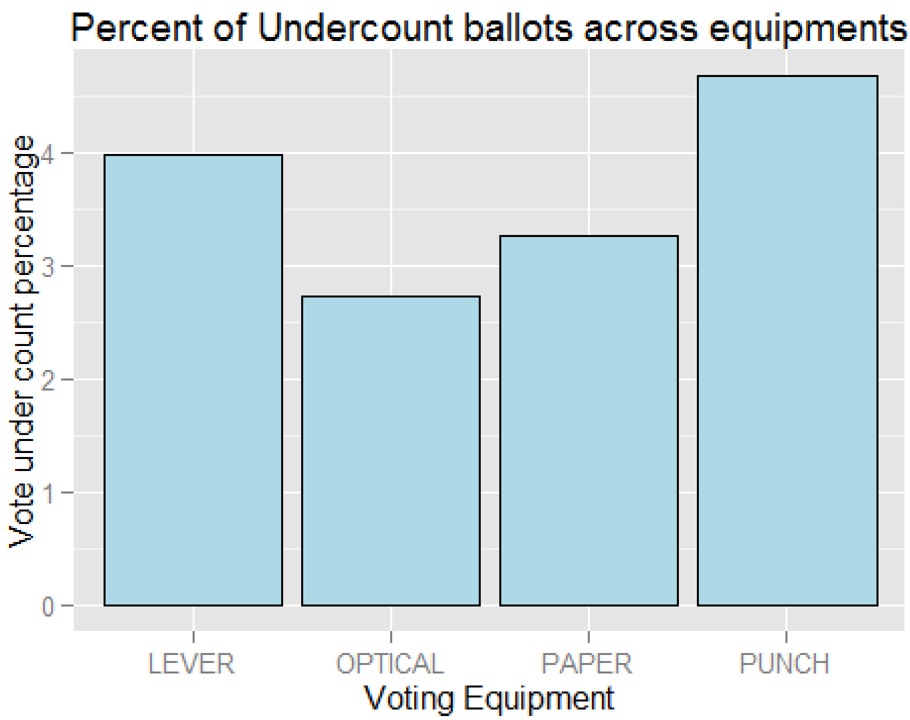
```
votes <-aggregate(votes ~ equip,data=georgia,FUN=sum, na.rm=TRUE)
ballots=aggregate(ballots~equip,data=georgia,FUN=sum,na.rm=TRUE)

ballot_undercount=merge(votes,ballots,by.x="equip",by.y="equip")
```

Finding the undercount for each equipment type

```
ballot_undercount$percent_ballot_diff= ((ballot_undercount$ballots -
ballot_undercount$votes)/ballot_undercount$ballots)*100
```

Plotting the undercount percentage for each equipment type

```
ggplot(ballot_undercount, aes(x=ballot_undercount$equip,
y=ballot_undercount$percent_ballot_diff)) +
geom_bar(stat="identity",fill="lightblue", colour="black")+
  labs(x="Voting Equipment",y="Vote under count percentage",title="Percent of
Undercount ballots across equipments")
```

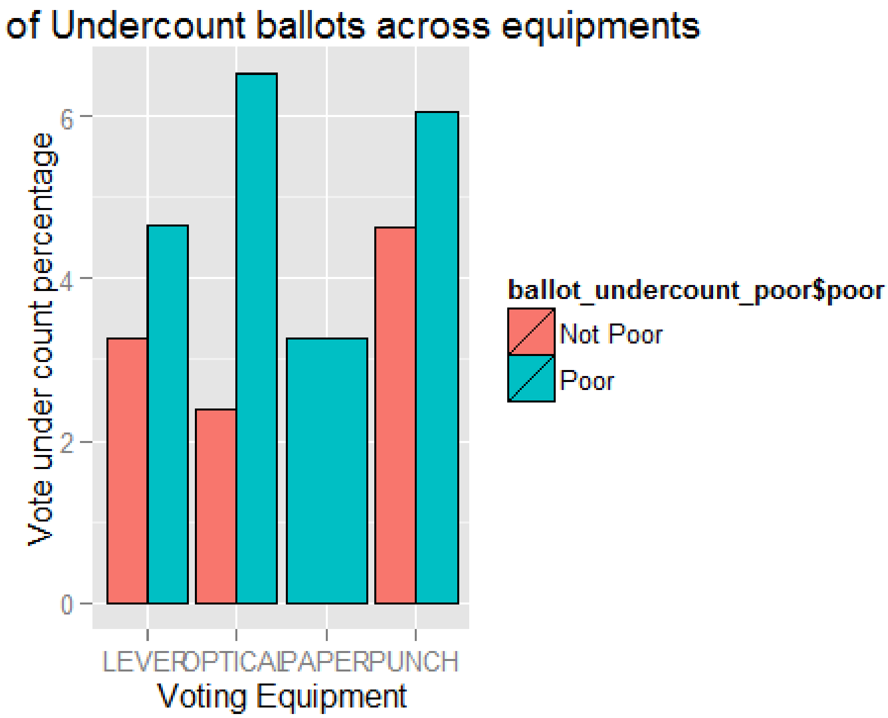


In order to find out the effect of ballot undercount on poor segments and minorities, we can aggregate the percent of undercount for the poor and non-poor counties

Creating a new data frame consisting of the counted votes, ballots and their percentage on the basis of poor and non-poor counties

```
votes_poor <-aggregate(votes ~ equip+poor,data=georgia,FUN=sum, na.rm=TRUE)
ballots_poor=aggregate(ballots~equip+poor,data=georgia,FUN=sum,na.rm=TRUE)
ballot_undercount_poor=merge(votes_poor,ballots_poor,by=c("equip","poor"))
ballot_undercount_poor$poor=ifelse(ballot_undercount_poor$poor==1,"Poor","Not
Poor")
ballot_undercount_poor$poor=factor(ballot_undercount_poor$poor)

# Creating variable for the percentage undercount
ballot_undercount_poor$percent_ballot_diff= ((ballot_undercount_poor$ballots
- ballot_undercount_poor$votes)/ballot_undercount_poor$ballots)*100
```

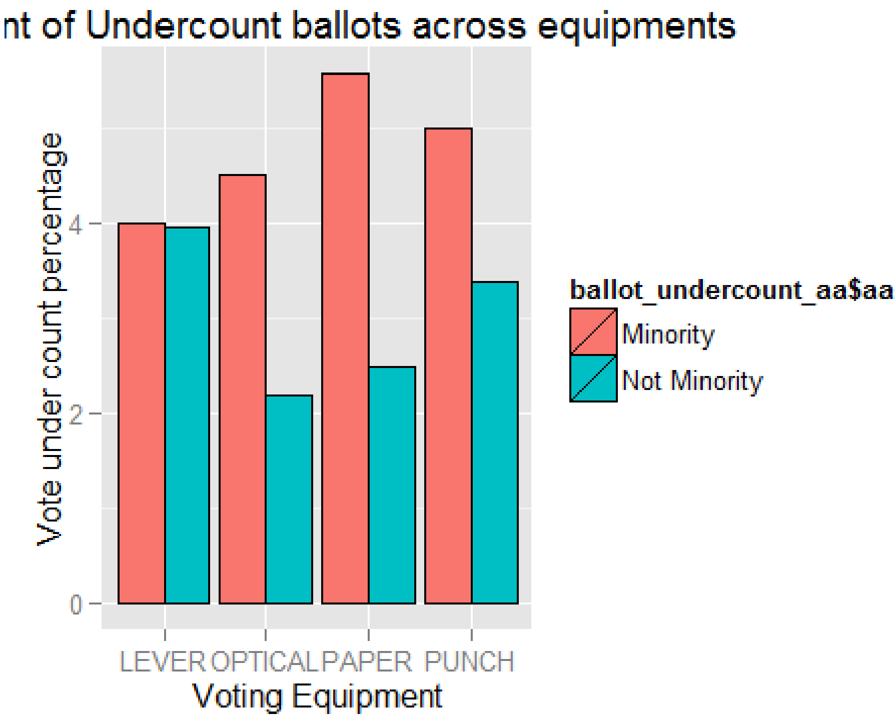
From the following plot , we can see that vote undercount is higher for poorer areas have higher rates of undercount. For optical equipment the undercount percent is very high compared to non-poor areas.



Creating a new data frame consisting of the counted votes, ballots and their percentage on the basis of percentage of African American and non-African American counties

```
georgia$aa=ifelse(georgia$perAA>0.25,"Minority","Not Minority")
votes_aa <-aggregate(votes ~ equip+aa,data=georgia,FUN=sum, na.rm=TRUE)
ballots_aa=aggregate(ballots~equip+aa,data=georgia,FUN=sum,na.rm=TRUE)
ballot_undercount_aa=merge(votes_aa,ballots_aa,by=c("equip","aa"))
ballot_undercount_aa$aa=factor(ballot_undercount_aa$aa)

# Creating variable for the percentage undercount
ballot_undercount_aa$percent_ballot_diff= ((ballot_undercount_aa$ballots -
ballot_undercount_aa$votes)/ballot_undercount_aa$ballots)*100
```

From the following plot , we can see that vote undercount is higher for minority areas have higher rates of undercount. Especially for paper and punch equipment the undercount percentage increases to a large extent.



Answer 2

Import the stocks

```
mystocks = c("SPY", "TLT", "LQD","EEM","VNQ")
myprices = yahooSeries(mystocks, from='2011-01-01', to='2015-08-05')
```

A Helper Function for calculating %age returns from a Yahoo Series

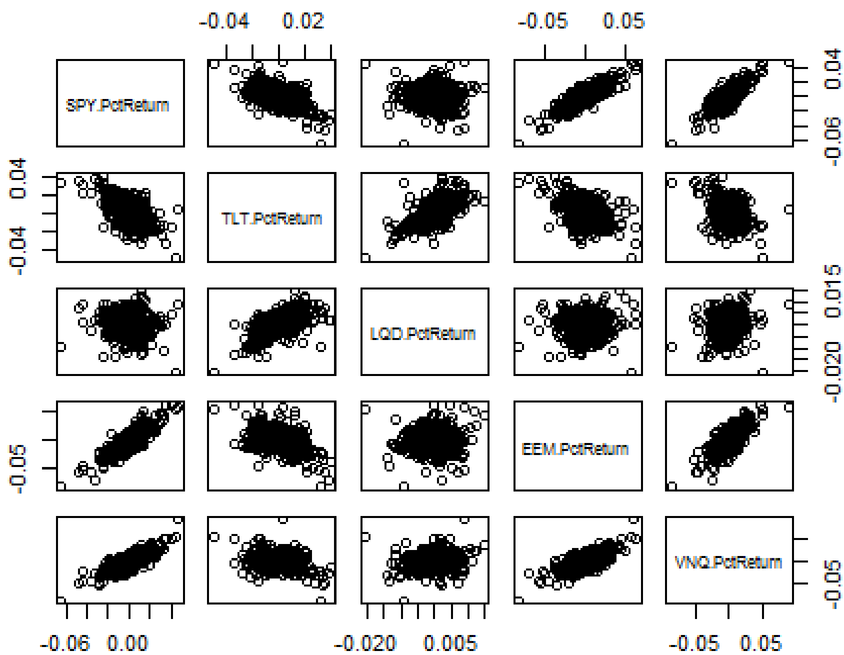
```
YahooPricesToReturns = function(series) {
  mycols = grep('Adj.Close', colnames(series))
  closingprice = series[,mycols]
  N = nrow(closingprice)
  percentreturn = as.data.frame(closingprice[2:N,]) /
as.data.frame(closingprice[1:(N-1),]) - 1
  mynames = strsplit(colnames(percentreturn), '.', fixed=TRUE)
  mynames = lapply(mynames, function(x) return(paste0(x[1], ".PctReturn")))
  colnames(percentreturn) = mynames
  as.matrix(na.omit(percentreturn))
}
```

Compute the returns from the closing prices

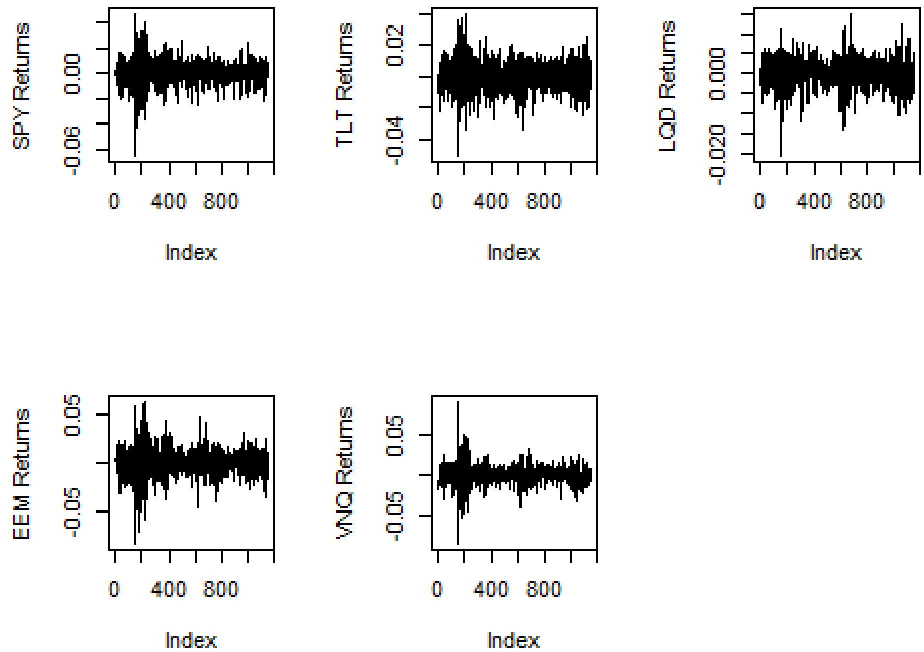
```
myreturns = YahooPricesToReturns(myprices)
```

These returns can be viewed as draws from the joint distribution

```
pairs(myreturns)
```




```
par(mfrow=c(2,3))
plot(myreturns[,1], type='l',ylab='SPY Returns')
plot(myreturns[,2], type='l',ylab='TLT Returns')
plot(myreturns[,3], type='l',ylab='LQD Returns')
plot(myreturns[,4], type='l',ylab='EEM Returns')
plot(myreturns[,5], type='l',ylab='VNQ Returns')
mu_SPY = mean(myreturns[,4])
sigma_SPY = sd(myreturns[,4])

mynames = sapply(data.frame(myreturns), function(x) sd(x))
```

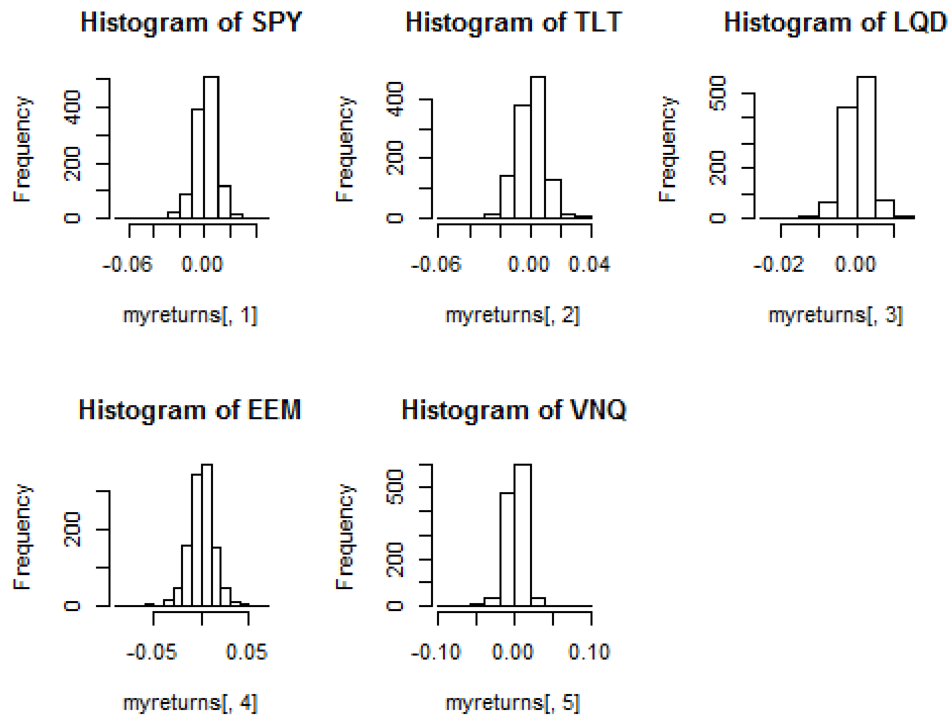


####Compute the moments of a one-day change in your portfolio

```
totalwealth = 100000
weights = c(0.20,0.20,0.20,0.20,0.20)     # What percentage of your wealth will you put in each stock?
```

How much money do we have in each stock?

```
holdings = weights * totalwealth
par(mfrow=c(2,3))
hist(myreturns[,1],main = paste("Histogram of SPY" ))
hist(myreturns[,2],main = paste("Histogram of TLT"))
hist(myreturns[,3],main = paste("Histogram of LQD" ))
hist(myreturns[,4],main = paste("Histogram of EEM" ))
hist(myreturns[,5],main = paste("Histogram of VNQ" ))
```



The standard deviation values helps in characterizing the risk/return properties for these stocks

LQD and and SPY safe stocks to purchase since they have smaller standard deviations

EEM and VNQ are riskier stocks to purchase since they have higher standard deviations

Portfolio with equal split amongst stocks

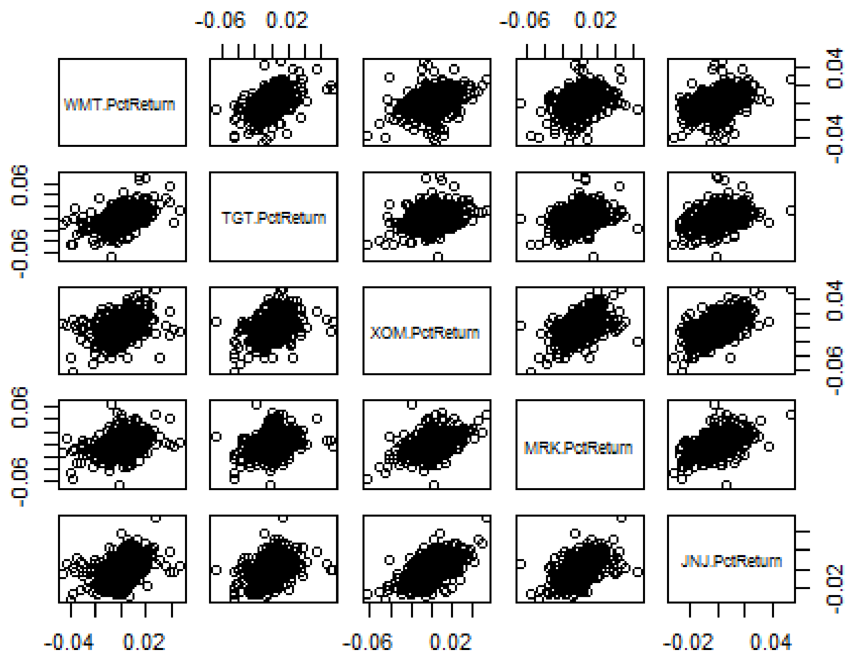
```
totalwealth = 100000
weights = c(0.20,0.20,0.20,0.20,0.20)
holdings = weights * totalwealth
```

Now use a bootstrap approach with more stocks

```
mystocks = c("WMT", "TGT", "XOM", "MRK", "JNJ")
myprices = yahooSeries(mystocks, from='2011-01-01', to='2015-07-30')
```

Compute the returns from the closing prices

```
myreturns = YahooPricesToReturns(myprices)
pairs(myreturns)
```



Sample a random return day

```
return.today = resample(myreturns, 1, orig.ids=FALSE)
```

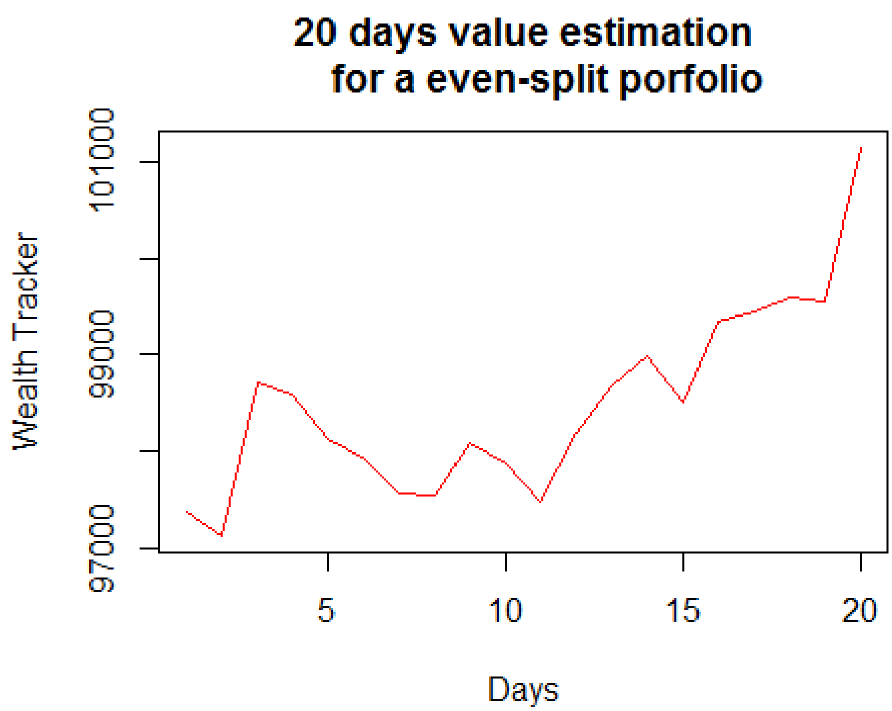
Update the value of the holdings and compute new wealth

```
holdings = holdings + holdings*return.today
totalwealth = sum(holdings)
par(mfrow=c(3,1))
```

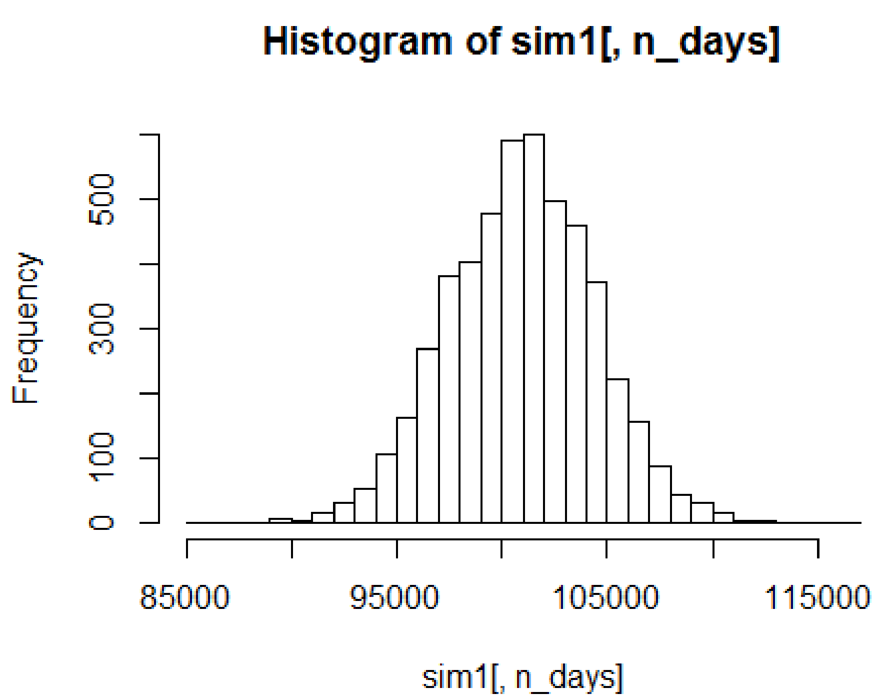
Bootstrapping for even split portfolio for a 20 day trading window

```
n_days=20
sim1 = foreach(i=1:5000, .combine='rbind') %do% {
  totalwealth = 100000
  weights = c(0.2, 0.2, 0.2, 0.2, 0.2)
  holdings = weights * totalwealth
  wealthtracker = rep(0, n_days)
  for(today in 1:n_days) {
    return.today = resample(myreturns, 1, orig.ids=FALSE)
    holdings = holdings + holdings*return.today
    totalwealth = sum(holdings)
    wealthtracker[today] = totalwealth
  }
  wealthtracker
}
plot(wealthtracker, type='l',xlab="Days",ylab="Wealth Tracker",main="20 days
```

```
value estimation
     for a even-split porfolio",col="red")
```

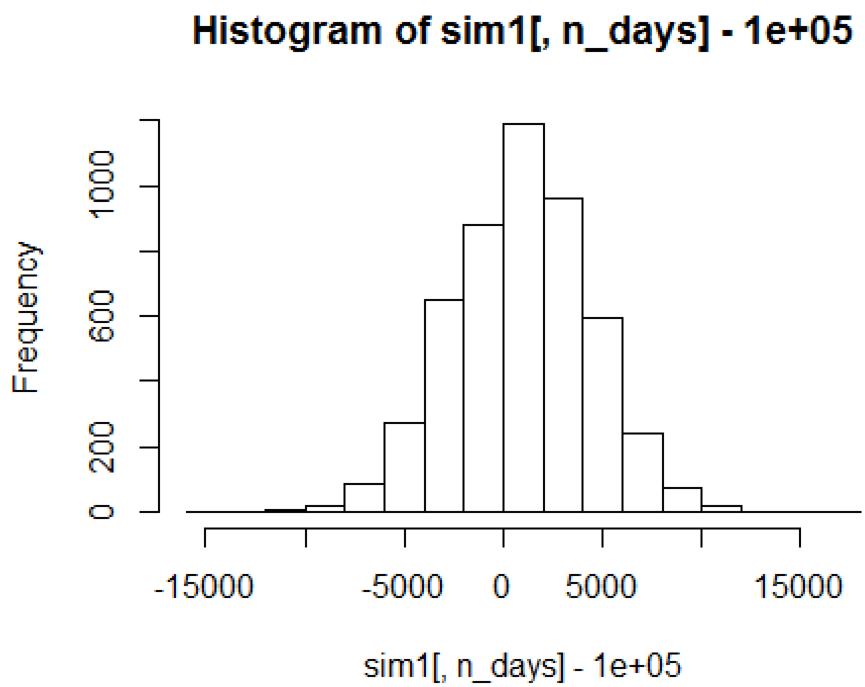



```
hist(sim1[,n_days], 25)
```



Find profit/loss and Calculate 5% value at risk

```
hist(sim1[,n_days]- 100000)
```



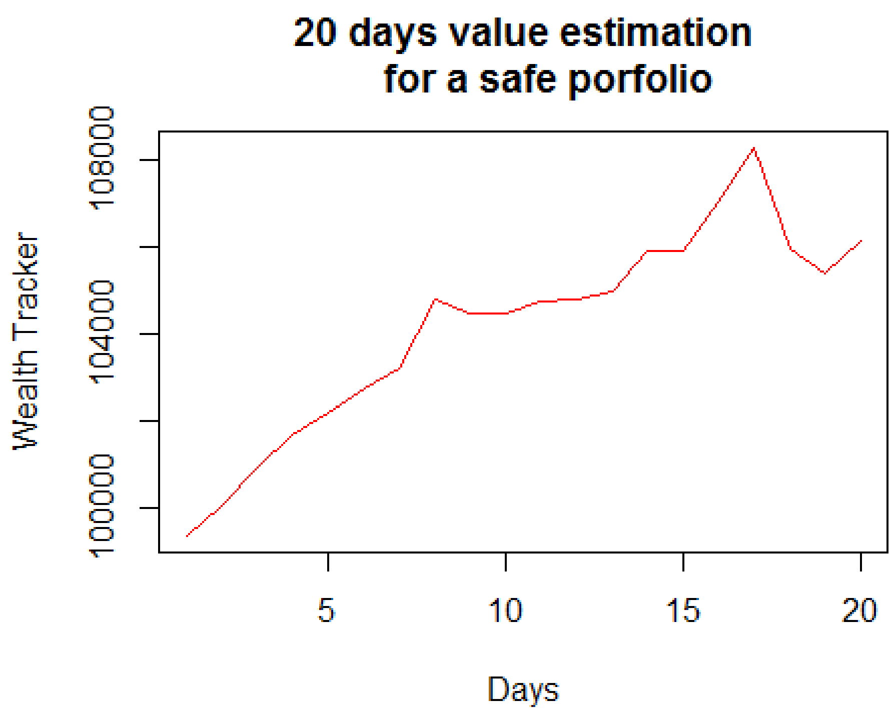

```
quantile(sim1[,n_days], 0.05) - 100000

##        5%
## -4772.255
```

Bootstrapping for safer portfolio over two trading weeks

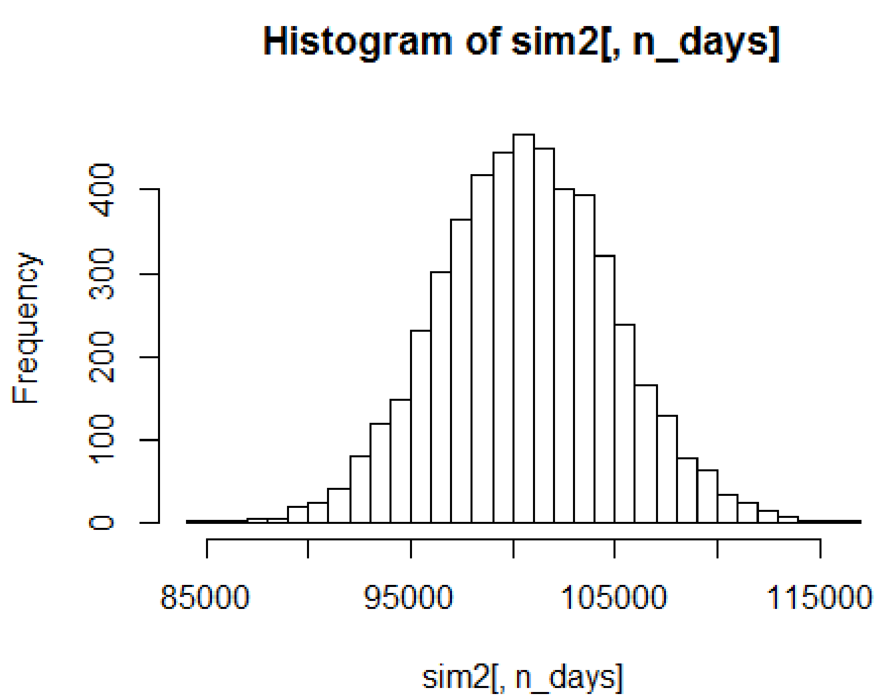
Considering the portfolio of SPY,TLT and LQD as a safe portfolio

```
n_days=20
sim2 = foreach(i=1:5000, .combine='rbind') %do% {
  totalwealth = 100000
  weights = c(0.15, 0.15, 0.70, 0, 0)
  holdings = weights * totalwealth
  wealthtracker = rep(0, n_days)
  for(today in 1:n_days) {
    return.today = resample(myreturns, 1, orig.ids=FALSE)
    holdings = holdings + holdings*return.today
    totalwealth = sum(holdings)
    wealthtracker[today] = totalwealth
  }
  wealthtracker
}
plot(wealthtracker, type='l',xlab="Days",ylab="Wealth Tracker",main="20 days
```

```
value estimation
     for a safe porfolio",col="red")
```

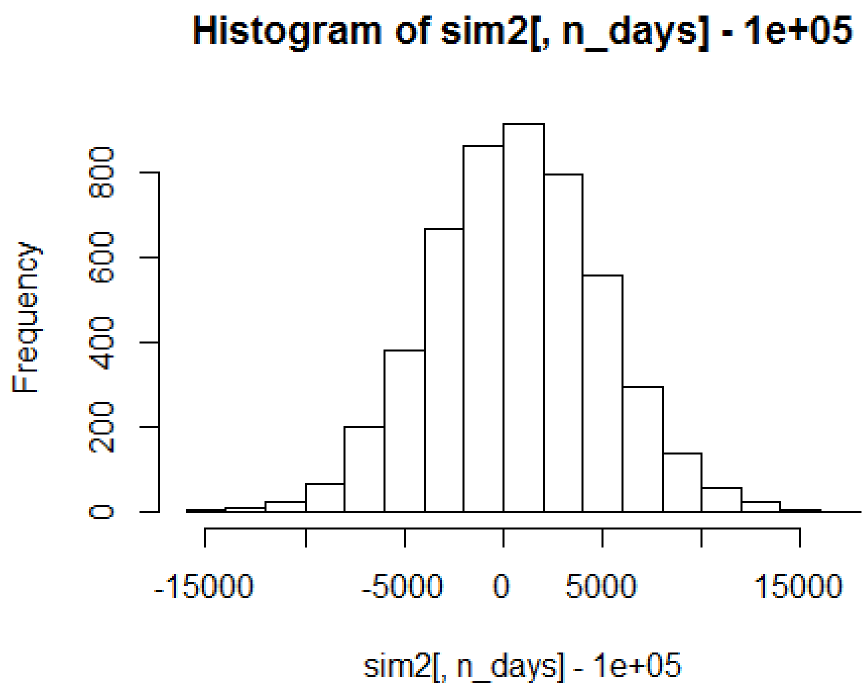



```
hist(sim2[,n_days], 25)
```



Find profit/loss and Calculate 5% value at risk

```
hist(sim2[,n_days]- 100000)
```



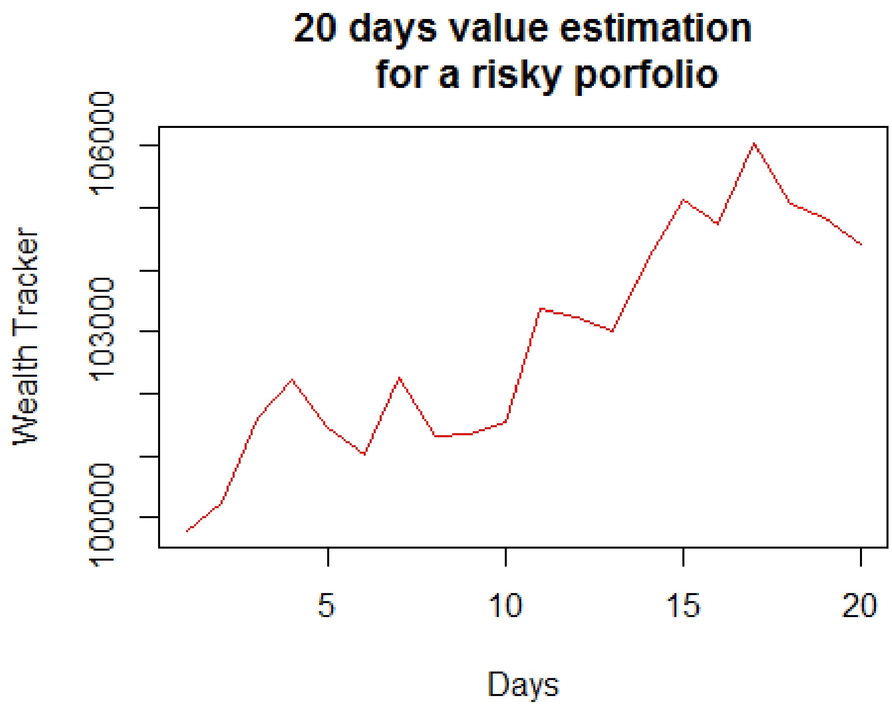

```
quantile(sim2[,n_days], 0.05) - 100000

##        5%
## -6388.769
```

Bootstrapping for riskier portfolio over two trading weeks

Considering the portfolio of EEM and VNQ as a risky portfolio

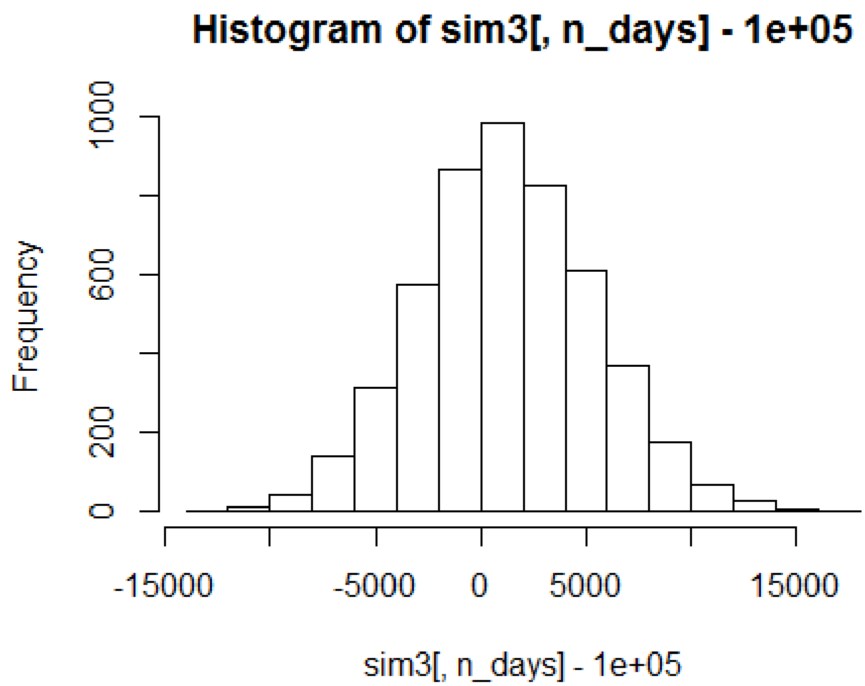
```
n_days=20
sim3 = foreach(i=1:5000, .combine='rbind') %do% {
  totalwealth = 100000
  weights = c(0,0,0,0.55, 0.45)
  holdings = weights * totalwealth
  wealthtracker = rep(0, n_days)
  for(today in 1:n_days) {
    return.today = resample(myreturns, 1, orig.ids=FALSE)
    holdings = holdings + holdings*return.today
    totalwealth = sum(holdings)
    wealthtracker[today] = totalwealth
  }
  wealthtracker
}
plot(wealthtracker, type='l',xlab="Days",ylab="Wealth Tracker",main="20 days
```

```
value estimation
     for a risky porfolio",col="red")
```



Find profit/loss and Calculate 5% value at risk

```
hist(sim3[,n_days]- 100000)
```




```
quantile(sim3[,n_days], 0.05) - 100000

##        5%
## -5569.765
```

Answer 3

Reading the wine.csv file

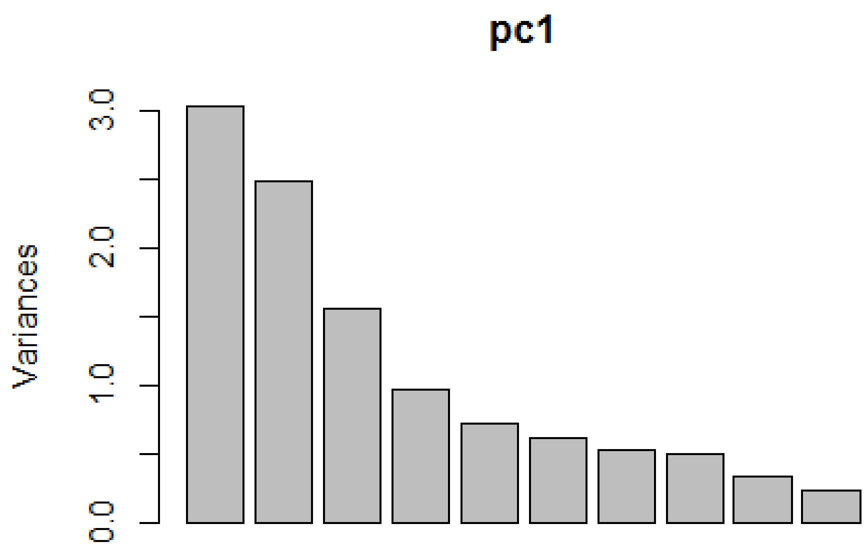
```
wine=
read.csv("https://raw.githubusercontent.com/jgscott/STA380/master/data/wine.c
sv",header = T)

# Creating a data frame with only numeric variables so that unsupervised
learning can be done.
X = wine[,(1:11)]
#Scaling the wine data around the mean so that the data points are on the
same level
wine_scaled = scale(X, center=TRUE, scale=TRUE)
```

Using the Principal Component Analysis to classify the data points

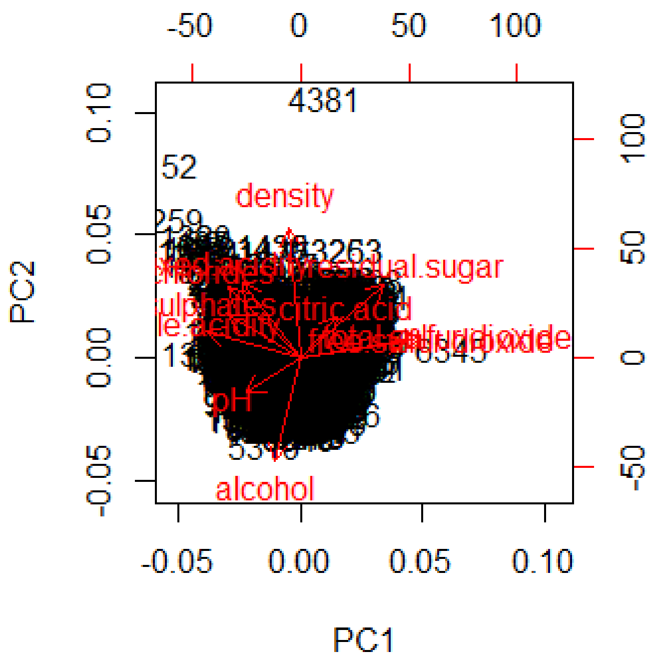
```
pc1 = prcomp(wine_scaled, scale.=TRUE,center.=TRUE)
summary(pc1)

## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6
## Standard deviation     1.7407 1.5792 1.2475 0.98517 0.84845 0.77930
```

```
## Proportion of Variance 0.2754 0.2267 0.1415 0.08823 0.06544 0.05521
## Cumulative Proportion  0.2754 0.5021 0.6436 0.73187 0.79732 0.85253
##                            PC7     PC8     PC9   PC10    PC11
## Standard deviation     0.72330 0.70817 0.58054 0.4772 0.18119
## Proportion of Variance 0.04756 0.04559 0.03064 0.0207 0.00298
## Cumulative Proportion  0.90009 0.94568 0.97632 0.9970 1.00000
```

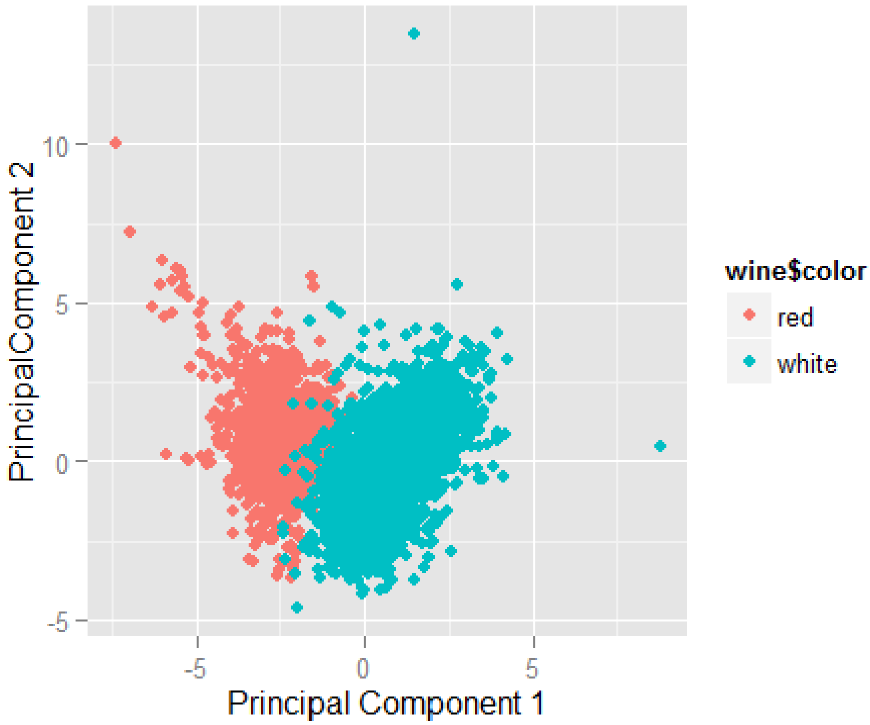
```r
plot(pc1)
```




```r
biplot(pc1)
```

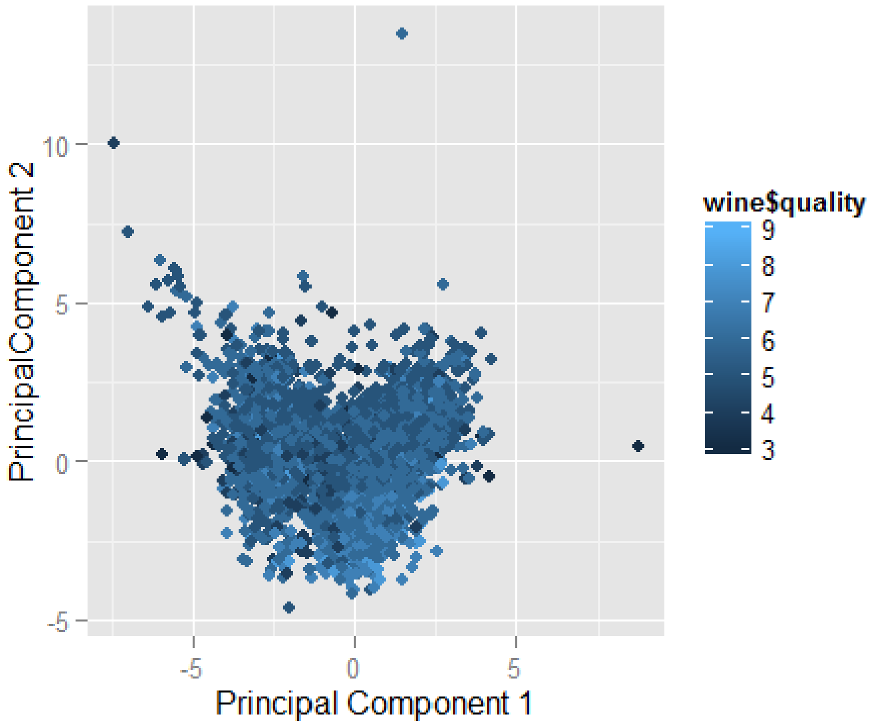



```r
# A more informative biplot
loadings = pc1$rotation
scores = pc1$x

#Plotting the PCA result to check its efficiency is discerning the wine
color.
qplot(scores[,1], scores[,2], color=wine$color, xlab='Principal Component 1',
ylab='PrincipalComponent 2')
```



Looking at above plot we can conclude that PCA helps us discern the color of wine effectively

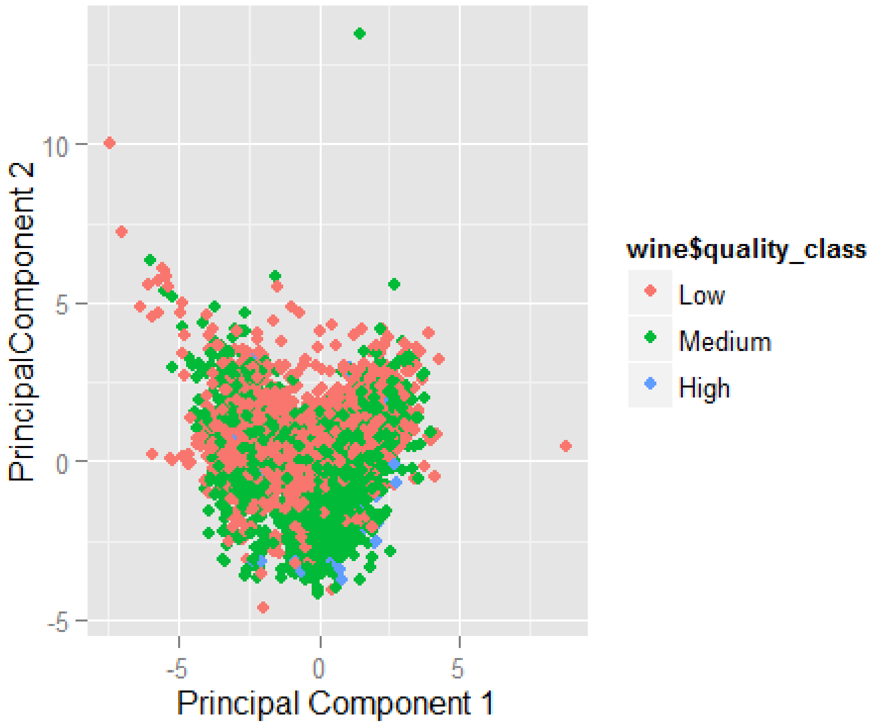
```
#Plotting the PCA result to check its efficiency is discerning the wine
quality.
qplot(scores[,1], scores[,2], color=wine$quality, xlab='Principal Component
1', ylab='PrincipalComponent 2')
```



Looking at above plot we are not able to discern the quality of wine effectively because we cannot find different clusters of different colors in the plot.

We can also check if the PCA is able to differentiate the the wine quality by creating bins of different qualities

```
# Creating factor dummy variable to create quality bins.
wine$quality_class=factor(rep(NA,length(wine$quality)),levels=c("Low","Medium
","High"))
wine$quality_class[wine$quality %in% c("3","4","5")]="Low"
wine$quality_class[wine$quality %in% c("6","7")]="Medium"
wine$quality_class[wine$quality %in% c("8","9")]="High"

# The following plot shows that PCA is unable to identify between different
wine qualities.
qplot(scores[,1], scores[,2], color=wine$quality_class, xlab='Principal
Component 1', ylab='PrincipalComponent 2')
```



Using clustering to classify different wine colors

```
set.seed(10)
# Creating two clusters to check if clusters can differentiate between wine colors.
clust1 = kmeans(wine_scaled, 2, nstart=25)

# Which color wines are in which clusters?
clust_out <- table(wine$color, clust1$cluster)
```

The following table shows that K means clustering is effective in differentiating between wine colors

```
clust_out

##
##             1    2
##   red      24 1575
##   white  4830   68
```

Using the K means to find out if it can be used to differentiate the wine quality.

```
# Creating 7 clusters for different ratings of wines
set.seed(10)
clust2 = kmeans(wine_scaled, 7, nstart=25)
clust_out <- table(wine$quality, clust2$cluster)
clust_out
```

```
##
##        1   2   3   4   5   6   7
##   3    4   2   6   4   5   7   2
##   4   20   2  64  15  63  24  28
##   5   77  27 472 198 413 652 299
##   6 528  16 343 263 538 645 503
##   7 451   2  43 140 144 122 177
##   8   96   0   2  14  30  21  30
##   9    4   0   0   0   0   1   0
```

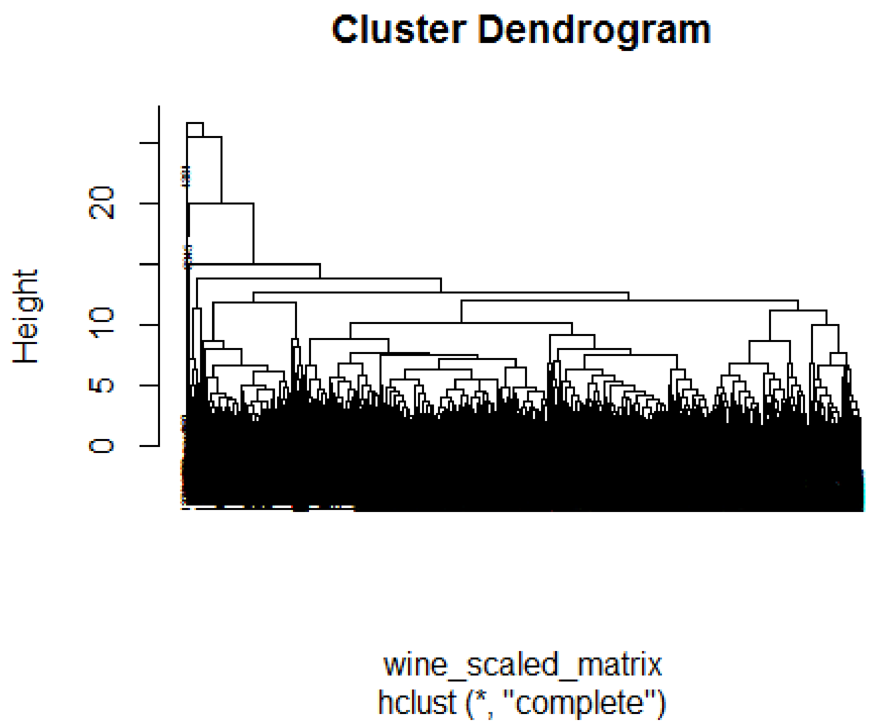
The above table shows that k means is not effective way to differentiate between the wine qualities.

Checking the efficiency of k-means for 3 quality subgroups differentiation

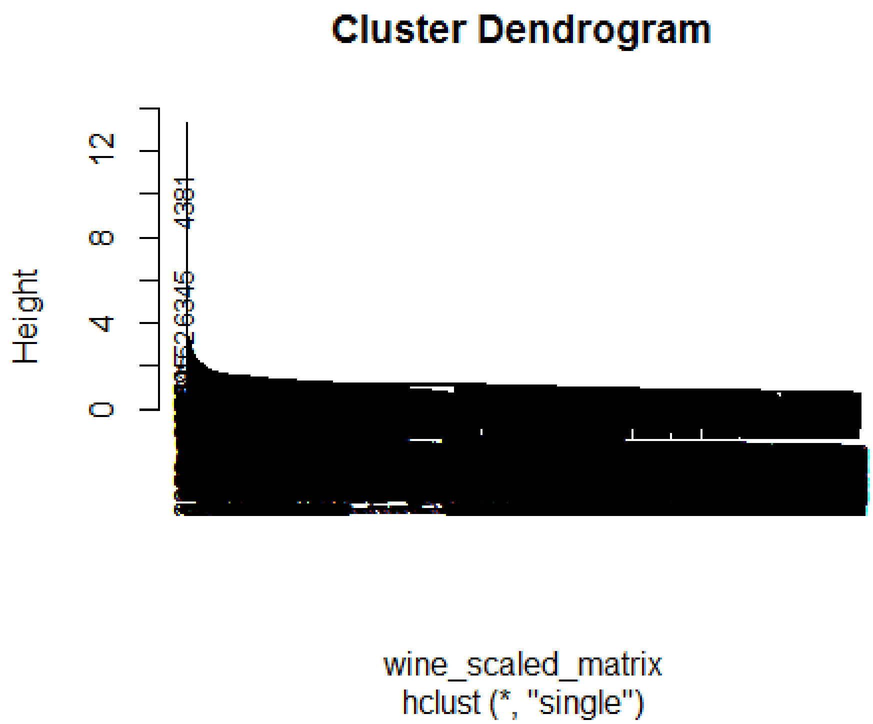
```
set.seed(10)
clust2 = kmeans(wine_scaled, 3, nstart=25)

# Which wine quality classes are in which clusters?

clust_out <- table(wine$quality_class, clust2$cluster)
clust_out

##
##              1    2    3
##   Low      746  774  864
##   Medium  2111  804 1000
##   High     152   15   31

# The following table shows that k means clustering is unable to distinguish
between the three quality classes of wine.
```

Using Hierarchical clusters for differentiating wines

```
# Form a pairwise distance matrix using the dist function
wine_scaled_matrix = dist(wine_scaled, method='euclidean')



# Now running hierarchical clustering
hier_wine = hclust(wine_scaled_matrix, method='complete')

new_tree=cutree(hier_wine,h=4)

# Plot the dendrogram
plot(hier_wine, cex=0.4)
```




```r
# Using single linkage instead
hier_wine2 = hclust(wine_scaled_matrix, method='single')

# Plot the dendrogram
plot(hier_wine2, cex=0.8)
```




```
cluster2 = cutree(hier_wine2, k=5)
```

The above dendogram shows that we cannot differentiate between various wine types.

The above figures and plots show that PCA and clustering techniques help us to classify wine colors but not the wine quality effectively.

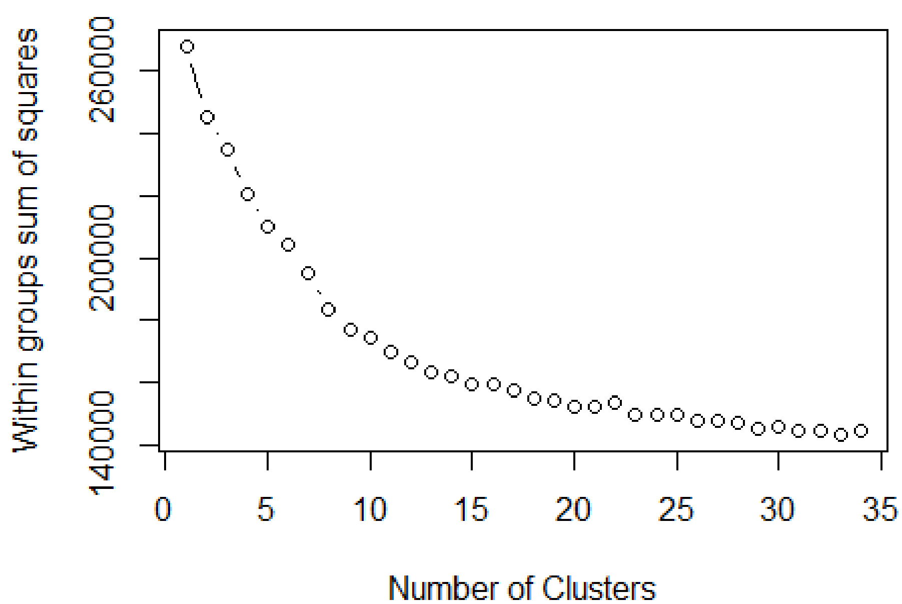
Answer 4

```
#Import the dataset and scaling the numeric dataset
soc_data=
read.csv("https://raw.githubusercontent.com/jgscott/STA380/master/data/social
_marketing.csv",header = T)
# Removing the customer code, spam and adult features
z = soc_data[,(2:35)]

# Scaling the data
z_scaled = scale(z, center=TRUE, scale=TRUE)
```

The denser the clusters and the more distant the clusters from each other the better

The scree plot shows that 'Within groups sum of Squares' value drops sharply with increasing no. of clusters. But it starts levelling around 12 clusters.Also, the 'Between groups sum of Squares' does not increase appreciably beyond 12 clusters

```
set.seed(10)
# Finding the optimum number of clusters

sum_squares_clust <- (nrow(z_scaled)-1)*sum(apply(z_scaled,2,var))
for (i in 1:34) sum_squares_clust[i] <- sum(kmeans(z_scaled,
                                                    centers=i)$withinss)
# plotting the sum_squares clusters to get optimum cluster number

plot(1:34, sum_squares_clust, type="b", xlab="Number of Clusters",
     ylab="Within groups sum of squares")
```



Cluster using k=12

```
set.seed(10)
cluster_var = kmeans(z_scaled, centers=12, nstart=50)

#calculate RSS
cluster_var$betweenss/cluster_var$totss

## [1] 0.3821614
```

```r
#Extracting attributes that would help us characterize the clusters from the output

mu=attr(z_scaled,"scaled:center")
sigma=attr(z_scaled,"scaled:scale")
```

To characterize each cluster, it helps to look at the scaled and unscaled center value of each cluster with repect to all of the twitter interests. If the standard deviation is greater than 2 then that interest can be labeled significant for that particular cluster.

Cluster1

```r
rbind(cluster_var$center[1,],(cluster_var$center[1,]*sigma + mu))
```

```
##        chatter current_events     travel photo_sharing uncategorized
## [1,] -0.369563     -0.2196938 -0.2129955    -0.4164946    -0.1952561
## [2,]  3.094522      1.2474950  1.0981964     1.5591182     0.6302605
##         tv_film sports_fandom   politics       food     family
## [1,] -0.2483568    -0.4061140 -0.2913631 -0.4413226 -0.3643344
## [2,]  0.6583166     0.7164329  0.9054776  0.6138945  0.4512358
##      home_and_garden      music       news online_gaming   shopping
## [1,]      -0.2121439 -0.2917548 -0.3026139    -0.2290798 -0.3928900
## [2,]       0.3643955  0.3787575  0.5698063     0.5931864  0.6786907
##      health_nutrition college_uni sports_playing    cooking        eco
## [1,]        -0.305563  -0.2769334     -0.2703801 -0.3255042 -0.2854544
## [2,]         1.193387   0.7471610      0.3754175  0.8817635  0.2925852
##       computers   business   outdoors     crafts automotive        art
## [1,] -0.2583557 -0.2707157 -0.3284311 -0.3174961 -0.3150457 -0.2275283
## [2,]  0.3443554  0.2358049  0.3854375  0.2565130  0.3994656  0.3540414
##        religion     beauty  parenting     dating     school
## [1,] -0.3922291 -0.2923284 -0.3946435 -0.2144380 -0.3905641
## [2,]  0.3443554  0.3169673  0.3233133  0.3286573  0.3036072
##      personal_fitness    fashion small_business
## [1,]       -0.3268054 -0.3048342     -0.2107220
## [2,]        0.6760187  0.4392118      0.2060788
```

Cluster2

```r
rbind(cluster_var$center[2,],(cluster_var$center[2,]*sigma + mu))
```

```
##          chatter current_events    travel photo_sharing uncategorized
## [1,] -0.02655763      0.2419996 0.0316027     0.1409417   -0.05218839
## [2,]  4.30503145      1.8333333 1.6572327     3.0817610    0.76415094
##         tv_film sports_fandom   politics     food   family home_and_garden
## [1,] 0.01260456      2.890259 -0.1284219 2.560271 2.019383       0.3475669
## [2,] 1.09119497      7.839623  1.3993711 5.943396 3.150943       0.7767296
##          music      news online_gaming  shopping health_nutrition
## [1,] 0.2259001 0.0353881    0.06036877 0.1359165     -0.003065448
## [2,] 0.9119497 1.2798742    1.37106918 1.6352201      2.553459119
##      college_uni sports_playing   cooking       eco computers  business
## [1,] 0.004629803      0.2506055 0.1068695 0.4538358 0.2362104 0.2698234
```

```
## [2,] 1.562893082      0.8836478 2.3647799 0.8616352 0.9276730 0.6100629
##          outdoors    crafts automotive        art religion    beauty
## [1,] 0.008088958 0.9930679   0.343210 0.08396001 3.050768 0.5985747
## [2,] 0.792452830 1.3270440   1.298742 0.86163522 6.937107 1.5000000
##       parenting      dating   school personal_fitness   fashion
## [1,]   3.019428 0.001669871 2.370869        0.0994462 0.1962202
## [2,]   5.496855 0.713836478 3.584906        1.7012579 1.3553459
##      small_business
## [1,]      0.2138934
## [2,]      0.4685535
```

Cluster3

```r
rbind(cluster_var$center[3,],(cluster_var$center[3,]*sigma + mu))
```

```
##         chatter current_events     travel photo_sharing uncategorized
## [1,] -0.2750332     -0.0383591 -0.2620461     -0.310541    -0.1668143
## [2,]  3.4281298      1.4775889  0.9860896      1.848532     0.6568779
##         tv_film sports_fandom   politics     food    family
## [1,] -0.2315206      1.065489 -0.3269773 0.946935 0.7902616
## [2,]  0.6862442      3.896445  0.7975270 3.078825 1.7588872
##      home_and_garden      music       news online_gaming   shopping
## [1,]     -0.03541458 -0.1627911 -0.2913313    -0.1990541 -0.2391828
## [2,]      0.49459042  0.5115920  0.5935085     0.6738794  0.9567233
##      health_nutrition college_uni sports_playing    cooking         eco
## [1,]       -0.2928855  -0.2643501     -0.1703976 -0.2775138 -0.05313591
## [2,]        1.2503864   0.7836167      0.4729521  1.0463679  0.47140649
##       computers    business   outdoors    crafts automotive        art
## [1,] -0.1139432 -0.07553669 -0.2177372 0.2710297 -0.1164419 -0.1280010
## [2,]  0.5146832  0.37094281  0.5193199 0.7372488  0.6707883  0.5162287
##      religion    beauty parenting     dating    school personal_fitness
## [1,] 1.242455 0.0102046  1.044323 -0.1811745 0.7600107       -0.2910670
## [2,] 3.474498 0.7187017  2.503864  0.3879444 1.6707883        0.7619784
##         fashion small_business
## [1,] -0.1891690     -0.1090395
## [2,]  0.6506955      0.2689335
```

Cluster4

```r
rbind(cluster_var$center[4,],(cluster_var$center[4,]*sigma + mu))
```

```
##         chatter current_events     travel photo_sharing uncategorized
## [1,] -0.1629943   -0.009377078 -0.1577987    -0.1248761     0.1515886
## [2,]  3.8235294    1.514363885  1.2243502     2.3556772     0.9548564
##         tv_film sports_fandom   politics      food      family
## [1,] -0.1652519    -0.2008207 -0.2001541 0.4703144 -0.09842558
## [2,]  0.7961696     1.1600547  1.1819425 2.2325581  0.75239398
##      home_and_garden       music        news online_gaming    shopping
## [1,]       0.1418382 -0.01799053 -0.06592795    -0.1082468 -0.06173351
## [2,]       0.6251710  0.66073871  1.06703146     0.9179207  1.27770178
##      health_nutrition college_uni sports_playing   cooking       eco
```

```
## [1,]         2.260136  -0.2184651    -0.01716306 0.4312624 0.5856090
## [2,]        12.729138   0.9165527     0.62243502 3.4774282 0.9630643
##          computers   business outdoors     crafts automotive         art
## [1,] -0.09449644 0.03479249 1.786915 0.07520677 -0.1768693 -0.07793736
## [2,]  0.53761970 0.44733242 2.943912 0.57729138  0.5882353  0.59781122
##          religion      beauty   parenting      dating      school
## [1,] -0.1734198 -0.2044566 -0.09975324 0.05860884 -0.1924756
## [2,]  0.7633379  0.4336525  0.77017784 0.81532148  0.5389877
##        personal_fitness    fashion small_business
## [1,]         2.185847 -0.1238214     -0.1413288
## [2,]         6.719562  0.7701778      0.2489740
```

Cluster5

```r
rbind(cluster_var$center[5,],(cluster_var$center[5,]*sigma + mu))
```

```
##           chatter current_events    travel photo_sharing uncategorized
## [1,] -0.06483084      0.3484281 0.3493262    -0.02067035     0.4701301
## [2,]  4.16996047      1.9683794 2.3833992     2.64031621     1.2529644
##         tv_film sports_fandom  politics      food     family home_and_garden
## [1,] 2.743135    -0.1596543 0.0462607 0.2280318 -0.1101367       0.3823739
## [2,] 5.620553     1.2490119 1.9288538 1.8023715  0.7391304       0.8023715
##            music       news online_gaming  shopping health_nutrition
## [1,] 0.09265337 0.09407479    -0.1953647 0.1190676       -0.1164917
## [2,] 0.77470356 1.40316206     0.6837945 1.6047431        2.0434783
##       college_uni sports_playing    cooking       eco   computers  business
## [1,]  -0.1023471     -0.1163344 -0.0974338 0.1919810 -0.1347703 0.2849748
## [2,]   1.2529644      0.5256917  1.6640316 0.6600791  0.4901186 0.6205534
##         outdoors    crafts automotive       art     religion      beauty
## [1,] -0.1830572 1.057220 -0.1821452 4.197658 -0.07253153 -0.01310634
## [2,]  0.5612648 1.379447  0.5810277 7.565217  0.95652174  0.68774704
##        parenting     dating     school personal_fitness     fashion
## [1,] -0.1984919 -0.1171960 0.0890540       -0.1247317 -0.01325875
## [2,]  0.6205534  0.5019763 0.8735178        1.1620553  0.97233202
##       small_business
## [1,]      0.6516162
## [2,]      0.7391304
```

Cluster6

```r
rbind(cluster_var$center[6,],(cluster_var$center[6,]*sigma + mu))
```

```
##        chatter current_events      travel photo_sharing uncategorized
## [1,] 1.037103     0.09941288 -0.04538134   -0.01624415     0.7540978
## [2,] 8.058824     1.65240642  1.48128342    2.65240642     1.5187166
##          tv_film sports_fandom   politics       food     family
## [1,] -0.1165200     -0.153629 -0.1490323 -0.1576074 -0.1064415
## [2,]  0.8770053      1.262032  1.3368984  1.1176471  0.7433155
##       home_and_garden       music       news online_gaming  shopping
## [1,]       0.5925674 -0.1247235 -0.1207456   -0.06974598 -0.108829
## [2,]       0.9572193  0.5508021  0.9518717    1.02139037  1.192513
```

```
##      health_nutrition college_uni sports_playing    cooking       eco
## [1,]      -0.07977629 -0.06968432      0.3095780 -0.1335641 0.1194863
## [2,]       2.20855615  1.34759358      0.9411765  1.5401070 0.6042781
##       computers  business   outdoors    crafts automotive         art
## [1,] 0.01642741 0.4159386 0.06915417 0.4224823 -0.1768693 -0.01490037
## [2,] 0.66844920 0.7112299 0.86631016 0.8609626  0.5882353  0.70053476
##          religion    beauty  parenting   dating   school personal_fitness
## [1,] 0.0004436047 0.2743934 0.08366903 4.884712 1.298085      -0.04536932
## [2,] 1.0962566845 1.0695187 1.04812834 9.417112 2.310160       1.35294118
##        fashion small_business
## [1,] 0.8354186      0.3815543
## [2,] 2.5240642      0.5721925
```

Cluster7

```
rbind(cluster_var$center[7,],(cluster_var$center[7,]*sigma + mu))
```

```
##          chatter current_events     travel photo_sharing uncategorized
## [1,] -0.06636963      0.1460143 -0.0428016      1.217125     0.4738121
## [2,]  4.16452991      1.7115385  1.4871795      6.021368     1.2564103
##         tv_film sports_fandom   politics       food     family
## [1,] -0.1441359    -0.2373136 -0.1382245 -0.2322906 0.01454653
## [2,]  0.8311966     1.0811966  1.3696581  0.9850427 0.88034188
##      home_and_garden     music        news online_gaming  shopping
## [1,]      0.09954925 0.5312803 -0.09376735    -0.0315906 0.1863772
## [2,]      0.59401709 1.2264957  1.00854701     1.1239316 1.7264957
##      health_nutrition college_uni sports_playing   cooking        eco
## [1,]       -0.0677072 -0.03109193      0.1727437  2.840641 0.00621975
## [2,]        2.2628205  1.45940171      0.8076923 11.741453 0.51709402
##       computers  business   outdoors     crafts   automotive        art
## [1,] 0.05296066 0.1972709 0.02775562 0.07214003 -0.003717885 0.01938801
## [2,] 0.71153846 0.5598291 0.81623932 0.57478632  0.824786325 0.75641026
##        religion   beauty   parenting      dating    school
## [1,] -0.1580674 2.630891 -0.08345746 -0.04637428 0.1163756
## [2,]  0.7927350 4.198718  0.79487179  0.62820513 0.9059829
##      personal_fitness  fashion small_business
## [1,]      -0.04374935 2.724800      0.1645212
## [2,]       1.35683761 5.978632      0.4380342
```

Cluster8

```
rbind(cluster_var$center[8,],(cluster_var$center[8,]*sigma + mu))
```

```
##          chatter current_events   travel photo_sharing uncategorized
## [1,] -0.08473768      0.1029475 3.305912     -0.109079   -0.08846081
## [2,]  4.09970674      1.6568915 9.140762      2.398827    0.73020528
##          tv_film sports_fandom  politics      food      family
## [1,] -0.07242664    -0.2056772  3.144394 0.1675681 -0.06623103
## [2,]  0.95014663     1.1495601 11.319648 1.6950147  0.78885630
##      home_and_garden       music      news online_gaming    shopping
## [1,]      0.03362944 -0.07012627 1.166881    -0.1595450 -0.06610483
```

```
## [2,]      0.54545455  0.60703812 3.656891     0.7800587  1.26979472
##       health_nutrition college_uni sports_playing    cooking       eco
## [1,]       -0.1581215 -0.03782923      0.0361951 -0.1850141 0.1383115
## [2,]        1.8563050  1.43988270      0.6744868  1.3636364 0.6187683
##       computers  business    outdoors    crafts automotive        art
## [1,]   2.965403 0.5661732 -0.03852803 0.2013754 -0.1394936 -0.1820496
## [2,]   4.146628 0.8152493  0.73607038 0.6803519  0.6392962  0.4281525
##        religion     beauty  parenting     dating     school personal_fitness
## [1,] 0.1232334 -0.2108069 0.01126906 0.2428454 -0.1080592       -0.1445574
## [2,] 1.3313783  0.4252199 0.93841642 1.1436950  0.6392962        1.1143695
##         fashion small_business
## [1,] -0.1777609      0.3809961
## [2,]  0.6715543      0.5718475
```

Cluster9

```
rbind(cluster_var$center[9,],(cluster_var$center[9,]*sigma + mu))
```

```
##          chatter current_events     travel photo_sharing uncategorized
## [1,] -0.08316324     0.06791589 -0.1805949    -0.2218071    -0.1248266
## [2,]  4.10526316     1.61244019  1.1722488     2.0909091     0.6961722
##          tv_film sports_fandom politics       food    family
## [1,] -0.02650795     0.6694648 1.238631 -0.1672768 0.2173663
## [2,]  1.02631579     3.0406699 5.543062  1.1004785 1.1100478
##       home_and_garden     music     news online_gaming  shopping
## [1,]        0.1440417 -0.106689 2.692196    -0.1231048 -0.186161
## [2,]        0.6267943  0.569378 6.861244     0.8779904  1.052632
##       health_nutrition college_uni sports_playing    cooking        eco
## [1,]       -0.2410929  -0.2136088     -0.1132423 -0.2324433 -0.1154459
## [2,]        1.4832536   0.9306220      0.5287081  1.2009569  0.4234450
##        computers   business  outdoors     crafts automotive        art
## [1,] -0.1973846 -0.1310077 0.2983682 -0.1512087   2.600671 -0.1599810
## [2,]  0.4162679  0.3325359 1.1435407  0.3923445   4.382775  0.4641148
##         religion     beauty  parenting     dating      school
## [1,] -0.1947532 -0.1797148 0.03296389 -0.1008556 -0.00382127
## [2,]  0.7224880  0.4665072 0.97129187  0.5311005  0.76315789
##       personal_fitness    fashion small_business
## [1,]        -0.2259252 -0.2336434     -0.1609550
## [2,]         0.9186603  0.5693780      0.2368421
```

Cluster10

```
rbind(cluster_var$center[10,],(cluster_var$center[10,]*sigma + mu))
```

```
##        chatter current_events     travel photo_sharing uncategorized
## [1,] 1.536559      0.3880221 -0.2094747      1.208100   -0.03190574
## [2,] 9.821468      2.0186199  1.1062432      5.996714    0.78313253
##         tv_film sports_fandom   politics       food      family
## [1,] -0.1810349    -0.2302846 -0.1279253 -0.3312022 -0.04323922
## [2,]  0.7699890     1.0963855  1.4008762  0.8094195  0.81489595
##       home_and_garden      music        news online_gaming shopping
```

```
## [1,]      0.04106329 0.07212564 -0.2678035    -0.1730732 1.542563
## [2,]      0.55093100 0.75355969  0.6429354     0.7437021 4.179628
##      health_nutrition college_uni sports_playing    cooking       eco
## [1,]        -0.212155  -0.1526123    -0.09046166 -0.2255693 0.3077359
## [2,]         1.613363   1.1073384     0.55093100  1.2245345 0.7491785
##        computers  business   outdoors      crafts automotive        art
## [1,] -0.03213512 0.3030637 -0.2712927 0.008073148  0.1237339 -0.2041649
## [2,]  0.61117196 0.6330778  0.4545455 0.522453450  0.9989047  0.3921139
##        religion     beauty  parenting     dating      school
## [1,] -0.3243874 -0.2431614 -0.2357629 -0.1530253 -0.06628266
## [2,]  0.4742607  0.3822563  0.5640745  0.4381161  0.68893757
##      personal_fitness    fashion small_business
## [1,]       -0.1579543 -0.1616641      0.1504785
## [2,]        1.0821468  0.7009858      0.4293538
```

Cluster11

```
rbind(cluster_var$center[11,],(cluster_var$center[11,]*sigma + mu))
```

```
##         chatter current_events      travel photo_sharing uncategorized
## [1,] -0.1610876      0.1814146 0.002363609    -0.1362083     0.9095602
## [2,]  3.8302583      1.7564576 1.590405904     2.3247232     1.6642066
##       tv_film sports_fandom   politics        food     family
## [1,] 2.135456   -0.04777451 -0.2480052 -0.06799847 -0.1600003
## [2,] 4.612546    1.49077491  1.0369004  1.27675277  0.6826568
##      home_and_garden    music       news online_gaming   shopping
## [1,]       0.2399064 2.593443 -0.1558006   -0.09006279 -0.1010181
## [2,]       0.6974170 3.350554  0.8782288    0.96678967  1.2066421
##      health_nutrition college_uni sports_playing    cooking         eco
## [1,]        -0.219723    1.159161       0.411487 -0.2404683 -0.05194297
## [2,]         1.579336    4.907749       1.040590  1.1734317  0.47232472
##       computers  business  outdoors      crafts automotive        art
## [1,] -0.1029278 0.4652721 0.0820594 0.009948394 -0.2211983 -0.2138178
## [2,]  0.5276753 0.7453875 0.8819188 0.523985240  0.5276753  0.3763838
##        religion      beauty  parenting     dating     school
## [1,] 0.09855999 -0.01416076 -0.2305620 -0.1296916 -0.2920527
## [2,] 1.28413284  0.68634686  0.5719557  0.4797048  0.4206642
##      personal_fitness    fashion small_business
## [1,]       -0.1921075 -0.1212345      0.7214336
## [2,]        1.0000000  0.7749077      0.7822878
```

Cluster12

```
rbind(cluster_var$center[12,],(cluster_var$center[12,]*sigma + mu))
```

```
##          chatter current_events      travel photo_sharing uncategorized
## [1,] -0.08806151    -0.08656388 -0.02500277  -0.009234166   -0.05399167
## [2,]  4.08797654     1.41642229  1.52785924   2.671554252    0.76246334
##         tv_film sports_fandom   politics       food    family
## [1,] 0.08138005    -0.1527509 -0.1585933 -0.1032981 0.1901105
## [2,] 1.20527859     1.2639296  1.3079179  1.2140762 1.0791789
```

```
##      home_and_garden       music       news online_gaming   shopping
## [1,]      0.04557156 -0.08436176 -0.1829878      3.636726 -0.1309539
## [2,]      0.55425220  0.59237537  0.8211144     10.982405  1.1524927
##      health_nutrition college_uni sports_playing   cooking         eco
## [1,]       -0.1842109    3.340954       2.197618 -0.127730 -0.06742145
## [2,]        1.7390029   11.228739       2.782991  1.560117  0.46041056
##        computers   business   outdoors     crafts automotive       art
## [1,] -0.07044911 -0.1030313 -0.1476365 0.02546127 0.07945748 0.2912399
## [2,]  0.56598240  0.3519062  0.6041056 0.53665689 0.93841642 1.1994135
##        religion     beauty  parenting      dating     school
## [1,] -0.2106324 -0.2306825 -0.1396781 -0.03521556 -0.2265203
## [2,]  0.6920821  0.3988270  0.7096774  0.64809384  0.4985337
##      personal_fitness     fashion small_business
## [1,]       -0.1823537 -0.07671655     0.08211774
## [2,]        1.0234604  0.85630499     0.38709677
```

The clusters have been profiled. Each cluster's characteritics have been profiled below:

1) Cluster 1 : No Segmentation

2) Cluster 2 : School, Parenting,Religion, Sports Fandom, Cooking, Family, Food, Health & nutrition

Cluster Description : Parents

3) Cluster 3 : No segmentation

4) Cluster 4 : Outdoor,Personal fitness, Cooking, Health & nutrition, Food

Cluster Description: Fitness and training enthusiasts

5) Cluster 5 : Art ,Travel ,TV, Film

Cluster Description : Travellers and art lovers

6) Cluster 6 : Fashion, Dating

Cluster Description : Outgoing young population

7) Cluster 7 : Fashion, Beauty,Cooking, Photo sharing

Cluster Description : Trendy home makers

8) Cluster 8 : Travel, Politics, News

Cluster Description : Executives & Travellers

9) Cluster 9 : Automotive,News, Politics, Sports fandom

Cluster Description : Young population with interests in cars & current affairs

10) Cluster 10 : Photo sharing, Shopping

Cluster Description : Shoppers

11) Cluster 11 : Music ,Tv & film ,College / University

Cluster Description : College going music and film lovers

12) Cluster 12 : Sports Playing, College / University, Online gaming

Cluster Description : College going students/ Gamers