Homework1-STA380

Question 1 - Exploratory Data Analysis

Objective:

1. Whether voting certain kinds of voting equipment lead to higher rates of undercount
2. If so, whether we should worry that this effect has a disparate impact on poor and minority communities.

```
library(ggplot2)

#Reading the dataset
georgia <- read.csv("C:/Users/Dhwani/Documents/Coursework/Summer - Predictive
Analytics/STA380/STA380/data/georgia2000.csv")
```

Exploring the dataset:

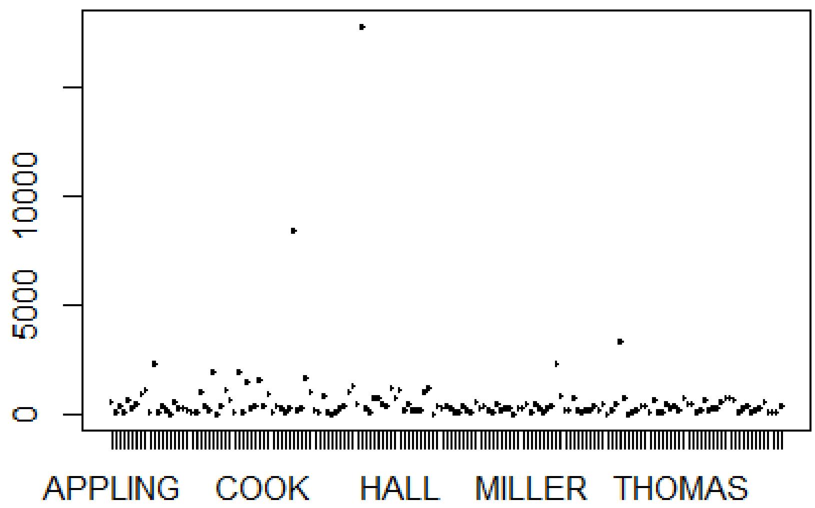
```
#Difference b/n ballots cast and votes recorded
attach(georgia)
votes.diff <- sum(ballots) - sum(votes)
votes.diff

## [1] 94681
```

Out of 2.691M ballots cast, only 2.596M votes were recorded.

94,681 votes were not counted.

Plotting undercounts by counties:

```
par(mfrow =c(1,1))
plot(county,ballots - votes)
```



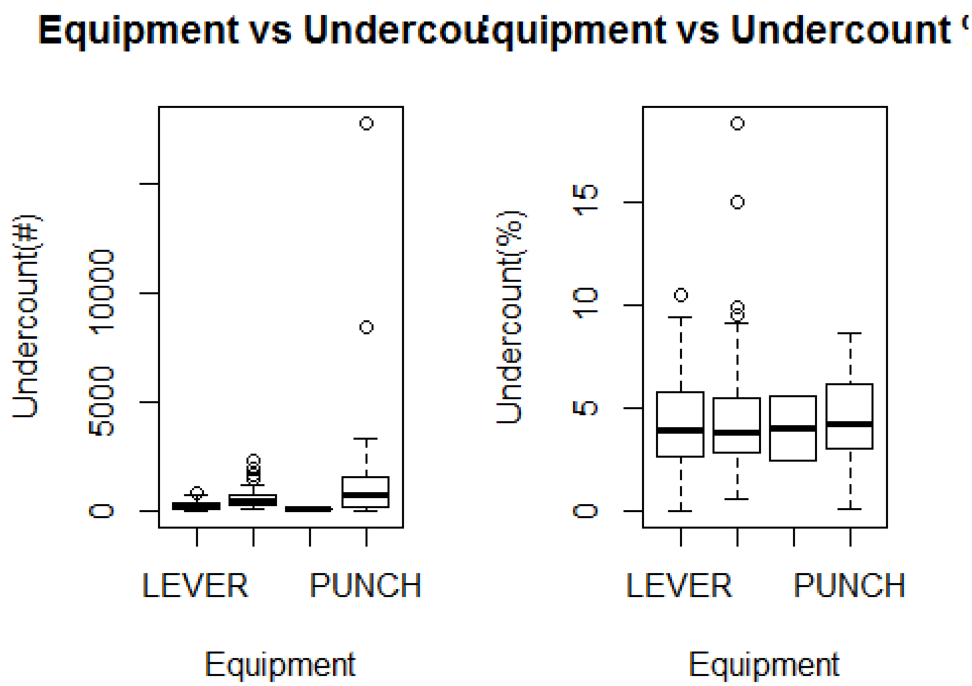

```
#Adding variables to the dataset
georgia$county.undercount <- ballots - votes
georgia$county.undercount.per <- (georgia$county.undercount/ballots) *100

max.issue <- which.max(georgia$county.undercount.scaled)
county.max.issue <- georgia[max.issue,1]
```

Fulton county had maximum undercount.

```
#Order data by maximum issue
georgia.sorted <- georgia[order(-georgia$county.undercount,-
georgia$county.undercount.per),]
head(georgia.sorted)

##       county ballots  votes   equip poor urban atlanta perAA   gore   bush
## 60    FULTON  280975 263211   PUNCH    0     1       1 0.416 152039 104870
## 44    DEKALB  228352 219980   PUNCH    0     1       1 0.514 154509  58807
## 121 RICHMOND   60904  57538   PUNCH    0     1       0 0.454  31413  25485
## 106 MUSCOGEE   54471  52163 OPTICAL    0     1       0 0.396  28193  23479
## 11      BIBB   52075  49776 OPTICAL    0     1       0 0.396  24996  24071
## 31   CLAYTON   63309  61398   PUNCH    0     1       1 0.509  40042  19966
##     county.undercount county.undercount.per
## 60              17764              6.322271
## 44               8372              3.666270
## 121              3366              5.526731
## 106              2308              4.237117
```

```
## 11                 2299                 4.414786
## 31                 1911                 3.018528
```

```r
par(mfrow =c(1,2))

#plot(county,georgia$county.undercount.scaled)
plot(equip,georgia$county.undercount, main = "Equipment vs Undercount", xlab
= "Equipment", ylab = "Undercount(#)")
plot(equip,georgia$county.undercount.per,main = "Equipment vs Undercount
%age", xlab = "Equipment", ylab = "Undercount(%)")
```



By looking at the plots of absolute numbers, we see that "Punch" and "Optical" seem to face more problem than other equipments. On plotting the percentage of undercounts by equipment in a county, we don't see a large variation among equipments.

Hence, trying a simple linear regression to test the relationship between undercounts and equipment:

```r
#Simple linear regression
lm.undercount <- lm(county.undercount ~ equip, data = georgia)
summary(lm.undercount)

##
## Call:
## lm(formula = county.undercount ~ equip, data = georgia)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max 
## -2260.5  -246.8  -110.9   116.1 15501.5 
## 
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)      229.9      171.9   1.338    0.183    
## equipOPTICAL     362.3      250.3   1.447    0.150    
## equipPAPER      -173.4     1059.5  -0.164    0.870    
## equipPUNCH      2032.5      397.7   5.111 9.32e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 1479 on 155 degrees of freedom
## Multiple R-squared:  0.1455, Adjusted R-squared:  0.129 
## F-statistic: 8.799 on 3 and 155 DF,  p-value: 2.009e-05
```

By regressing undercount with respect to equipment, we observe "punch"" to be statistically significant. Hence, voters who voted via "punch" were unable to record their votes significantly.

To determine whether this issue affected poor and the minorities, we shall compare counties via voting method

```
georgia.punch = subset(georgia,georgia$equip == "PUNCH")

round(sum(georgia.punch$poor)/nrow(georgia.punch) * 100,1)

## [1] 41.2
```

41% of counties which voted via Punch have 25% of residents living 1.5 times below federal poverty line

```
#Sorting data by %age undercounts per county
georgia.punch.sorted <- georgia.punch[order(-
georgia.punch$county.undercount.per,-georgia.punch$county.undercount),]
head(georgia.punch.sorted)

##      county ballots  votes equip poor urban atlanta perAA   gore   bush
## 19  CALHOUN    2065   1887 PUNCH    1     0       0 0.562   1107    768
## 142  TURNER    2661   2456 PUNCH    1     0       0 0.352   1169   1258
## 143  TWIGGS    3884   3615 PUNCH    1     1       0 0.446   1977   1570
## 60   FULTON  280975 263211 PUNCH    0     1       1 0.416 152039 104870
## 159   WORTH    6458   6061 PUNCH    1     0       0 0.266   2214   3792
## 90  LINCOLN    3300   3103 PUNCH    1     0       0 0.310   1275   1807
##     county.undercount county.undercount.per
## 19                178              8.619855
## 142               205              7.703871
## 143               269              6.925850
## 60              17764              6.322271
## 159               397              6.147414
## 90                197              5.969697
```

Out of top 6 counties, by percentage of undercount as compared to ballots, we see 5 out of 6 counties to be labeled as "poor".

```
# %age of poor counties?
aggregate(georgia$poor, by=list(equip), FUN=mean, na.rm=TRUE)
```

```
##   Group.1         x
## 1   LEVER 0.6081081
## 2 OPTICAL 0.2727273
## 3   PAPER 1.0000000
## 4   PUNCH 0.4117647
```

On comparison of "Punch" with counties which opted for other equiment method, we cannot strongly state that the poor were affected

```
#Average and median AA population in communities which voted via Punch?
aggregate(georgia$perAA, by=list(equip), FUN=mean, na.rm=TRUE)
```

```
##   Group.1         x
## 1   LEVER 0.2762432
## 2 OPTICAL 0.1860455
## 3   PAPER 0.4195000
## 4   PUNCH 0.2984706
```

```
#Average and median AA population in communities which voted via Punch?
aggregate(georgia$perAA, by=list(equip), FUN=median, na.rm=TRUE)
```

```
##   Group.1      x
## 1   LEVER 0.2515
## 2 OPTICAL 0.1580
## 3   PAPER 0.4195
## 4   PUNCH 0.3100
```

```
#median(head(georgia.punch.sorted$perAA))

#Plots
#ggplot(data=georgia, aes(x=equip, y=county.undercount.per, fill=poor)) +
geom_bar(stat="identity", position=position_dodge(), colour="black")
```

We see around 30% minorty population on an average in counties which voted via "punch". Top 6 counties affected have 38% African-American population on an average.

On comparison with counties which opted for a different voting methods, we cannot justify african americans being affected by "punch" equipment.

Question 2 - Portflio Analysis

Objective:

Considering the below five asset classes: . US domestic equities (SPY: the S&P 500 stock index) . US Treasury bonds (TLT) . Investment-grade corporate bonds (LQD) . Emerging-market equities (EEM) . Real estate (VNQ

Suppose there is a notional $100,000 to invest in one of the mentioned portfolios. Write a brief report that:

1. marshals appropriate evidence to characterize the risk/return properties of the five major asset classes listed above.
2. outlines your choice of the "safe" and "aggressive" portfolios.
3. uses bootstrap resampling to estimate the 4-week (20 trading day) value at risk of each of your three portfolios at the 5% level.

```r
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
```

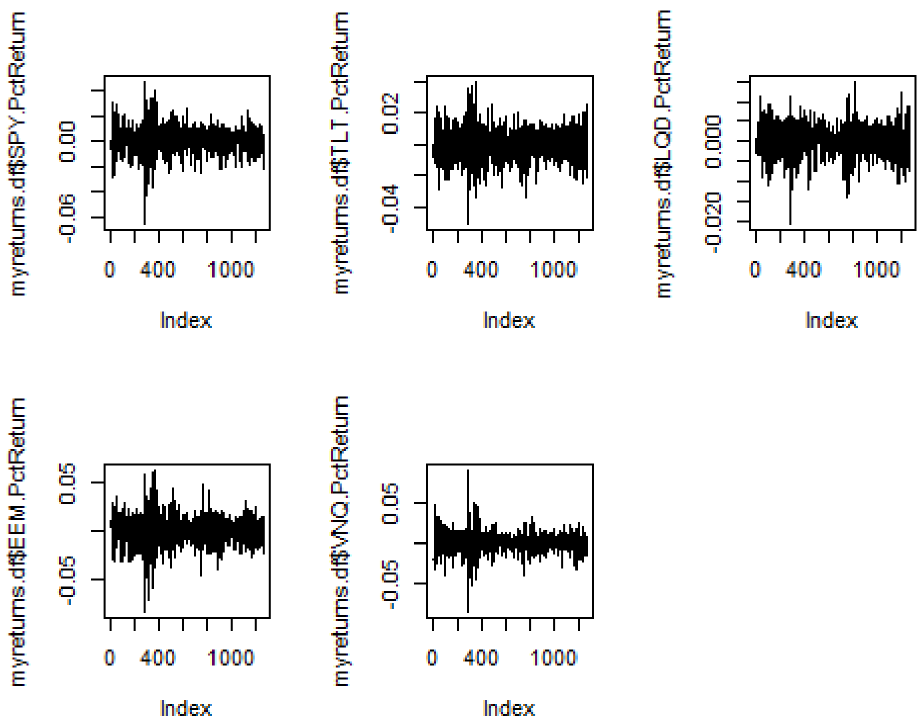
```
##
## The following objects are masked from 'package:base':
##
##     max, mean, min, prod, range, sample, sum
```

```
library(fImport)
```

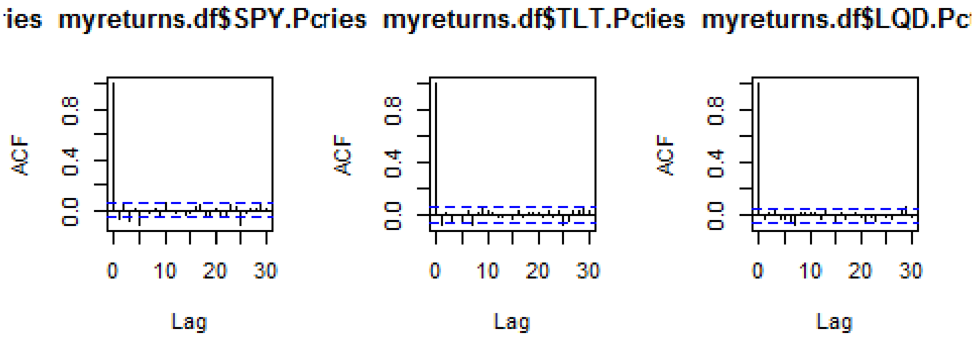
```
## Loading required package: timeDate
## Loading required package: timeSeries
```

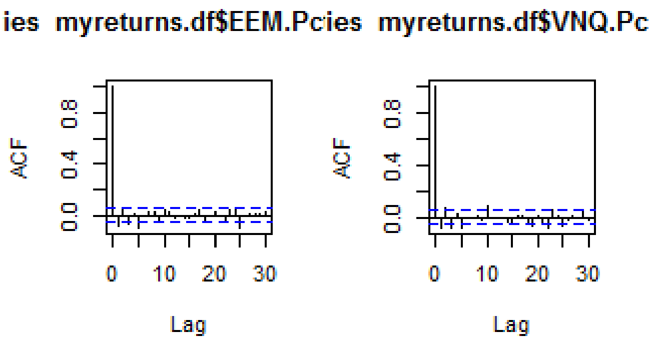
```
library(foreach)

# Import a few stocks
mystocks = c("SPY", "TLT", "LQD","EEM","VNQ")
myprices = yahooSeries(mystocks, from='2010-07-01', to='2015-06-30')
#head(myprices)
#tail(myprices)
#nrow(myprices)

#Returns on each day : %age of closing price on day x as compared to day (x-
1)

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))
}

#Creating a vector/list of returns for all stocks
myreturns = YahooPricesToReturns(myprices)

#First 6 days of returns for each stock
#head(myreturns)

#Converting the list to a dataframe
myreturns.df <- as.data.frame(myreturns)
#summary(myreturns.df)

#Plotting correlation between different stocks
#plot(myreturns.df)

#Plotting the day on day returns trend for all stocks to notice trends
plot.new()
par(mfrow = c(2,3))
```

```r
plot(myreturns.df$SPY.PctReturn, type = 'l')
plot(myreturns.df$TLT.PctReturn, type = 'l')
plot(myreturns.df$LQD.PctReturn, type = 'l')
plot(myreturns.df$EEM.PctReturn, type = 'l')
plot(myreturns.df$VNQ.PctReturn, type = 'l')

#Plotting the ACF to pull trends by lags
plot.new()
```




```r
par(mfrow = c(2,3))
acf(myreturns.df$SPY.PctReturn)
acf(myreturns.df$TLT.PctReturn)
acf(myreturns.df$LQD.PctReturn)
acf(myreturns.df$EEM.PctReturn)
acf(myreturns.df$VNQ.PctReturn)
```



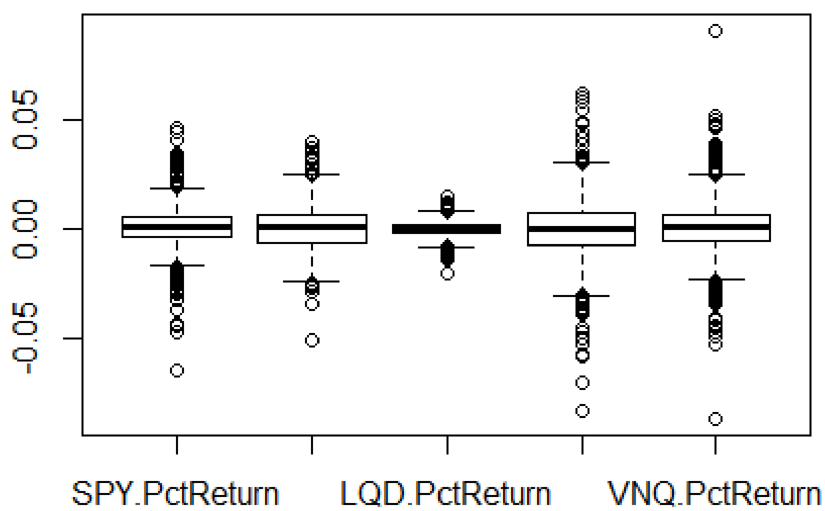


Plots show autocorrelation at the lag of 5, 10, 25 days etc since similar time in a week or month tend to behave similarly.

To estimate risk n each stock, we can estimate the variancealong with Beta (on the baseline of SPY) in each stock:

```
# Using CAPM to estimate portfolio variability
#Assuming SPY to be the market index

# First fit the market model to each stock
lm_TLT = lm(myreturns.df$TLT.PctReturn ~ myreturns.df$SPY.PctReturn)
lm_LQD = lm(myreturns.df$LQD.PctReturn ~ myreturns.df$SPY.PctReturn)
lm_EEM = lm(myreturns.df$EEM.PctReturn ~ myreturns.df$SPY.PctReturn)
lm_VNQ = lm(myreturns.df$VNQ.PctReturn ~ myreturns.df$SPY.PctReturn)

# The estimated beta for each stock based on daily returns
coef(lm_TLT)

##                (Intercept) myreturns.df$SPY.PctReturn
##               0.0006655208              -0.5595933444

coef(lm_LQD)

##                (Intercept) myreturns.df$SPY.PctReturn
##               0.0002415003              -0.0446051405

coef(lm_EEM)
```

```
##                (Intercept) myreturns.df$SPY.PctReturn
##              -0.0006155898               1.2217557452

coef(lm_VNQ)

##                (Intercept) myreturns.df$SPY.PctReturn
##              -6.243223e-05               9.758012e-01

sapply(myreturns.df, var)

## SPY.PctReturn TLT.PctReturn LQD.PctReturn EEM.PctReturn VNQ.PctReturn
##  8.951549e-05  9.533799e-05  1.277847e-05  1.883966e-04  1.395546e-04

#order(sapply(myreturns.df, sd))

plot.new()
par(mfrow = c(1,1))
boxplot(myreturns.df)
```



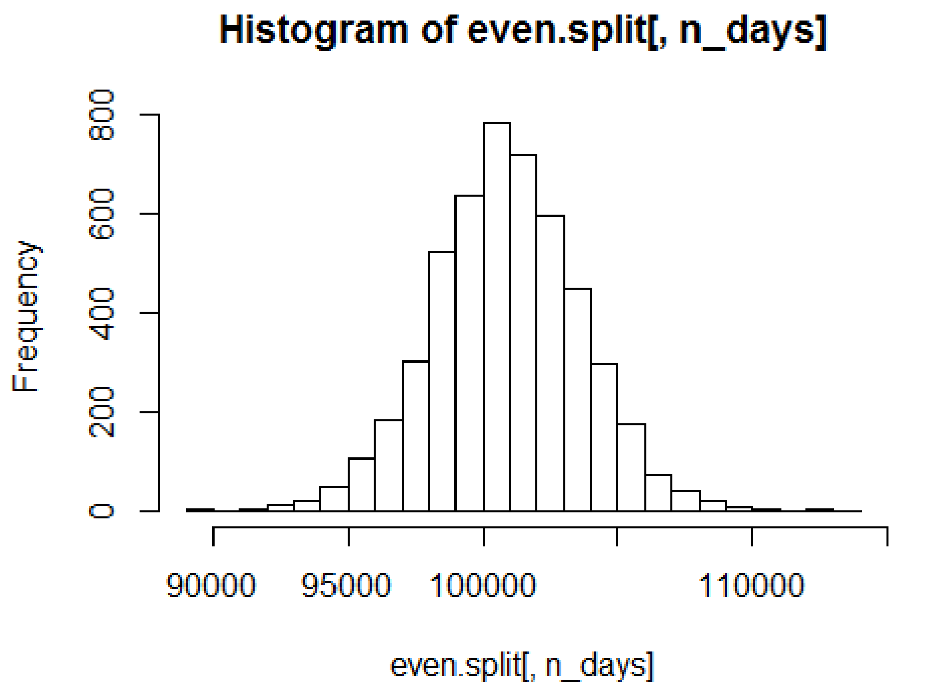
We observe that Emerging maret equities(EEM) has maximum volatility, followed by Real estate(VNQ).

Treasury bonds (TLT), Domestic equities(SPY) and Investment grade corporate bonds (LQD) follow in order of volatility in last 5 years.

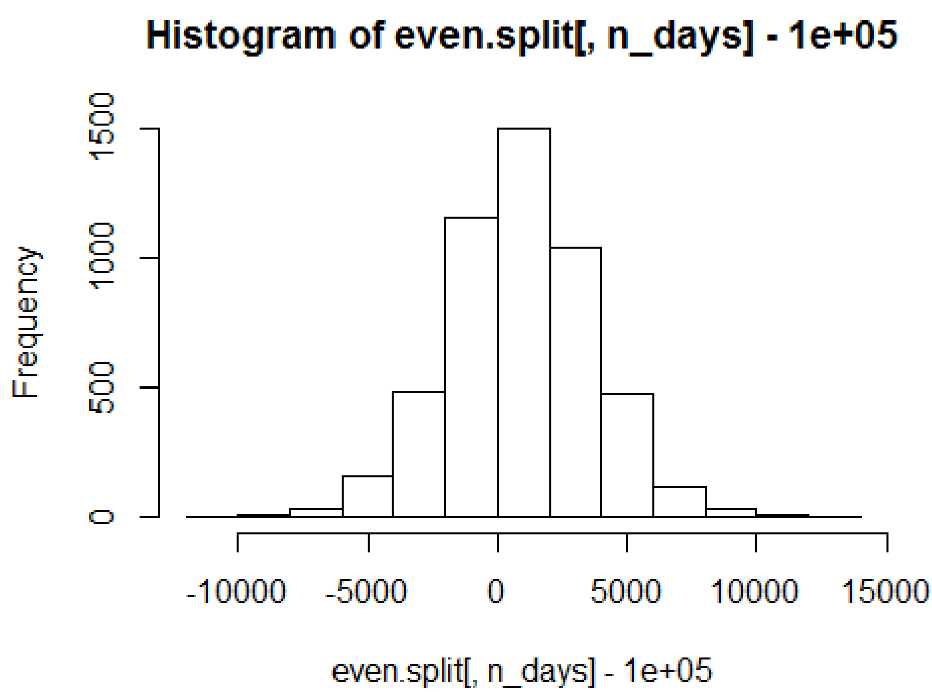
Volatlity is a measure of risk, hence variance in stocks is a good statistic to help estimate the risk pertaining to stocks. Also, via CAPM on the base line of SPY, we observe that EEM and VNQ have negative betas i.e. they have a tendency of decreasing when the market increases, hence more risky.

On splitting the porfolio across assets equally:

```
#Computing returns with equal split across portfolio

set.seed(12345)

even.split = foreach(i=1:5000, .combine='rbind') %do% {
    totalwealth = 100000
    n_days = 20
    weights = c(0.2, 0.2, 0.2, 0.2, 0.2)
    holdings = weights * totalwealth
    wealthtracker = rep(0, n_days) # Set up a placeholder to track total
wealth
    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
        #Redistributing the wealth at end of the day
        holdings = weights * totalwealth
    }
    wealthtracker
}# Now simulate many different possible trading years!


hist(even.split[,n_days], 25)
```




```r
# Profit/loss
hist(even.split[,n_days]- 100000)
```



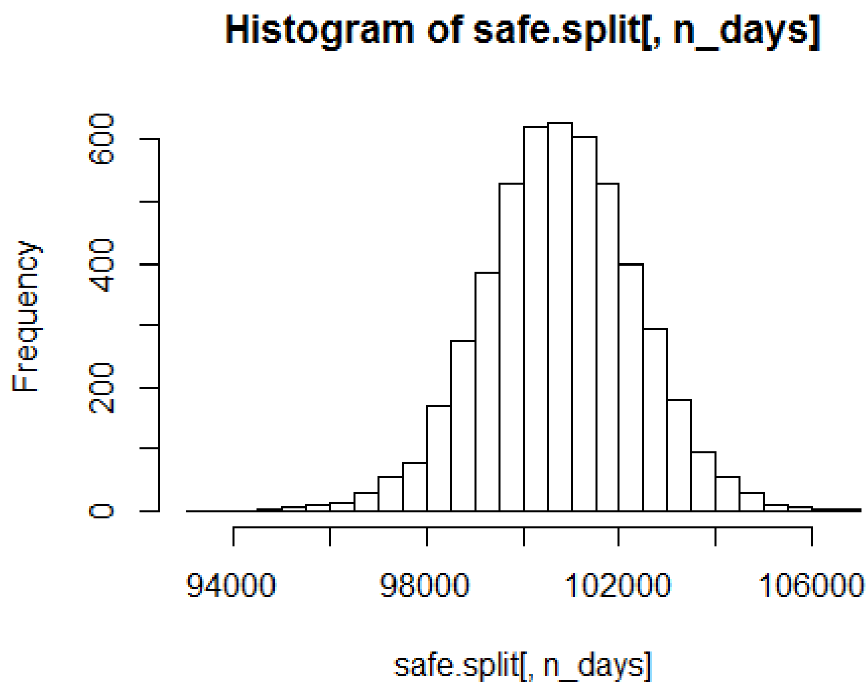
```
# Calculate 5% value at risk
quantile(even.split[,n_days], 0.05) - 100000

##        5%
## -3551.717

cbind(quantile(even.split[,n_days], 0.025),quantile(even.split[,n_days],
0.975))

##          [,1]     [,2]
## 2.5% 95425.24 106254.2
```

There is a possibility of $3,551 loss at 5% value of risk i.e. with 95% confidence, one can argue that the maximum loss expected is $3551. Expected returns of this portfolio is in range of $95,425 to $106,254 on an investment of $100,000

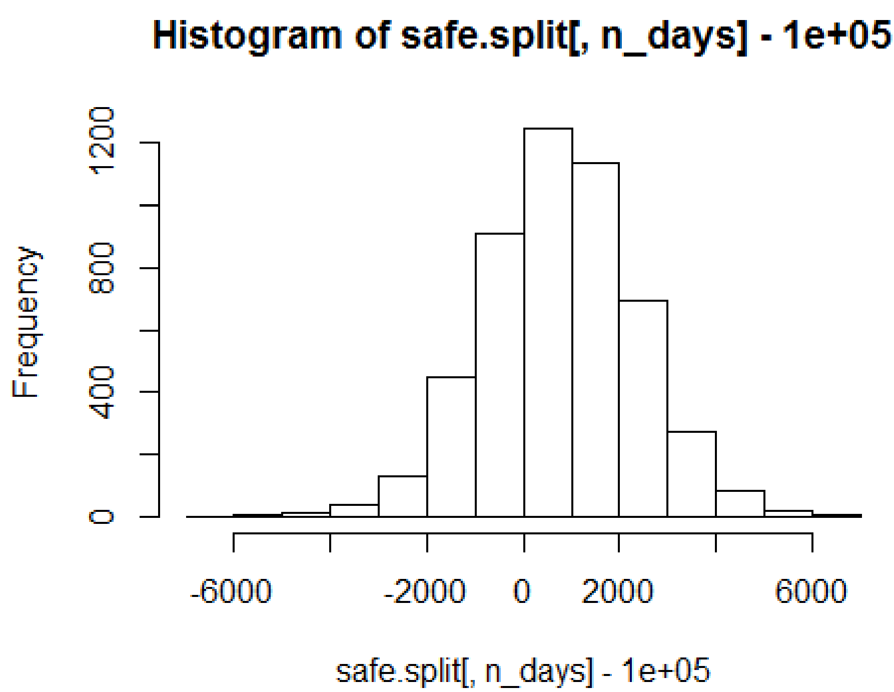
Splitting my investment in safe assets:

Assets with least variance are low-risk/safe assets. We have chosen to invest 60% of money in corporate bonds, since they are least risk and 30% and 10% in domestic equities and Treasury bonds respectively.

```
#Computing returns with safe split across portfolio

set.seed(12345)

# Now simulate many different possible trading years!
safe.split = foreach(i=1:5000, .combine='rbind') %do% {
    totalwealth = 100000
    n_days = 20
    weights = c(0.3, 0.1, 0.6, 0, 0)
    holdings = weights * totalwealth
    wealthtracker = rep(0, n_days) # Set up a placeholder to track total
wealth
    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
        #Redistributing the wealth at end of the day
        holdings = weights * totalwealth
    }
    wealthtracker
}

hist(safe.split[,n_days], 25)
```




```r
# Profit/loss
hist(safe.split[,n_days]- 100000)
```



```r
# Calculate 5% value at risk
quantile(safe.split[,n_days], 0.05) - 100000
```

```
##        5%
## -1784.931
```

```r
cbind(quantile(safe.split[,n_days], 0.05),quantile(safe.split[,n_days],
0.95))
```

```
##         [,1]     [,2]
## 5% 98215.07 103313.3
```

Loss at 5% value of risk drops to $1,784.

My returns also drops to the max of $103,313.

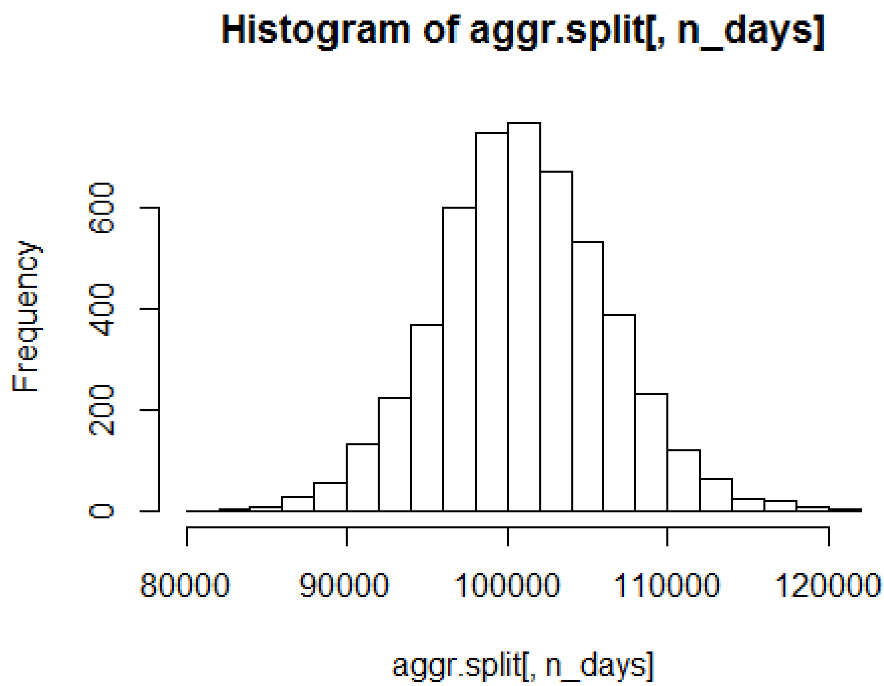
The range of returns in a safe portflio at 95% confidence level is $98,215 to $103,313

Computing returns with aggressive portfolios:

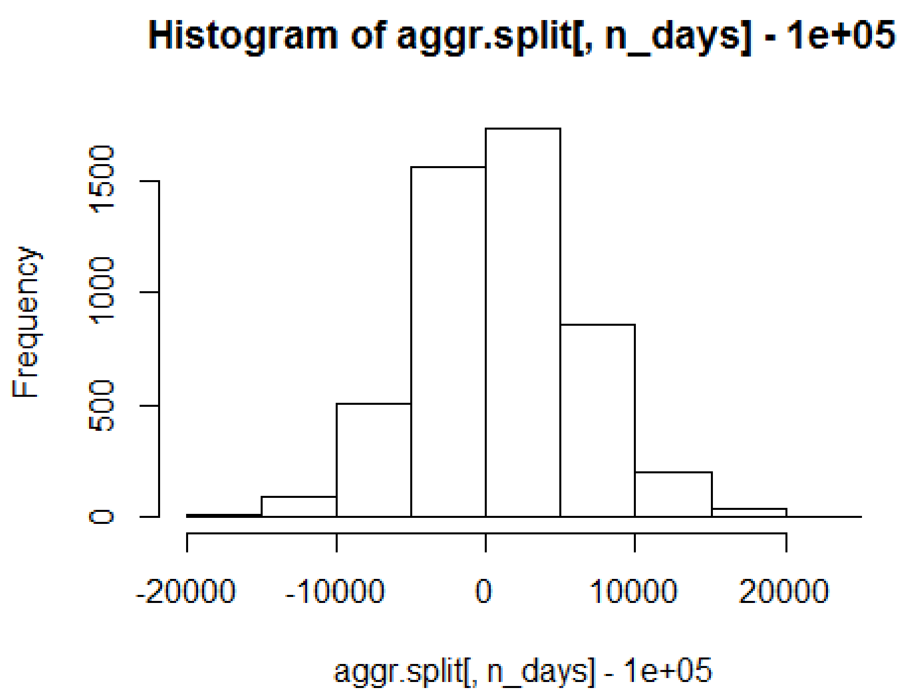
Agressive portfolios have higher variance as compared to other portfolios and indicate higher risk and may in turn give better returns.

We have chosen to invest 50%-50% in both real estate and equities

```r
#Computing returns with agressive split across portfolio

set.seed(12345)

# Now simulate many different possible trading years!
aggr.split = foreach(i=1:5000, .combine='rbind') %do% {
    totalwealth = 100000
    n_days = 20
    weights = c(0, 0, 0, 0.5, 0.5)
    holdings = weights * totalwealth
    wealthtracker = rep(0, n_days) # Set up a placeholder to track total
wealth
    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
        #Redistributing the wealth at end of the day
        holdings = weights * totalwealth
    }
    wealthtracker
}

hist(aggr.split[,n_days], 25)
```




```
# Profit/loss
hist(aggr.split[,n_days]- 100000)
```



```
# Calculate 5% value at risk
quantile(aggr.split[,n_days], 0.05) - 100000

##        5%
## -7623.076

cbind(quantile(aggr.split[,n_days], 0.025),quantile(aggr.split[,n_days],
0.975))

##           [,1]     [,2]
## 2.5% 90631.96 111746.2
```

Loss at 5% value of risk increases to $7623.

My returns also changes to the range of $90,504 to $111,404.

Comparising the returns from each investment and estimating return/risk, one can decide on assets to invest in.

Question 3 - Clustering and PCA

Objective:

1. Which dimensionality reduction technique makes more sense to you for this data?
2. Convince yourself (and me) that your chosen method is easily capable of distinguishing the reds from the whites, using only the "unsupervised" information contained in the data on chemical properties
3. Does this technique also seem capable of sorting the higher from the lower quality wines?

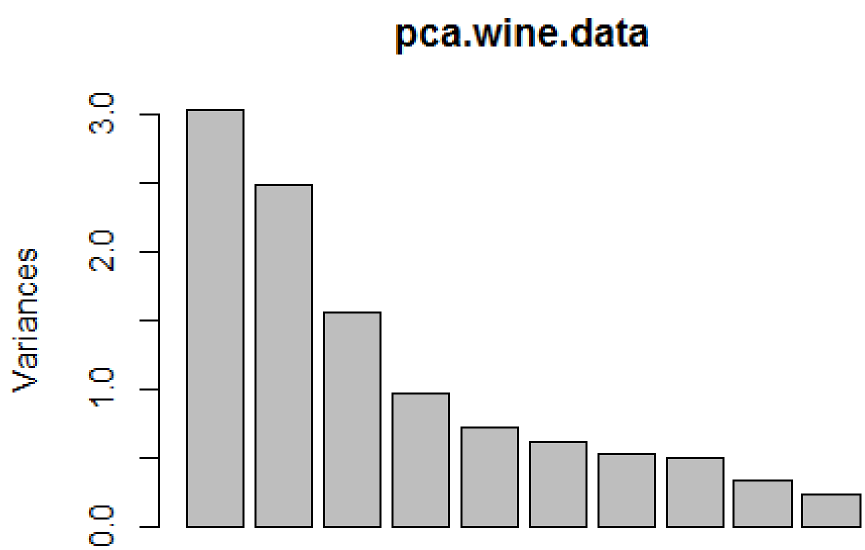
Understanding the data!

```
set.seed(111)
library(ggplot2)

#Reading data
wine.data <- read.csv("C:/Users/Dhwani/Documents/Coursework/Summer -
Predictive Analytics/STA380/STA380//data/wine.csv")

#head(wine.data)

#Checking unique quality scores
unique(wine.data$quality)

## [1] 5 6 7 4 8 3 9

# Picking out the relevant columns from the data set
cluster.wine.data = wine.data[,1:11]
```

```
# Run PCA
pca.wine.data = prcomp(cluster.wine.data, scale.=TRUE)
```

Summarizing PCA objective function to understand the variance captured by each principle component

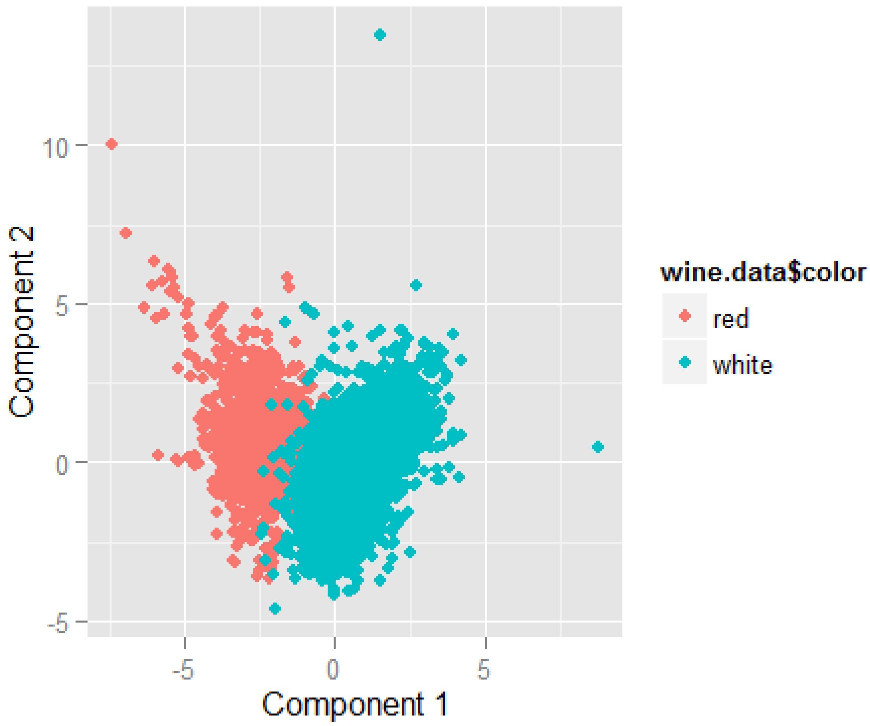
```
summary(pca.wine.data)

## 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

#sum((pca.wine.data$sdev)^2)

plot(pca.wine.data)
```



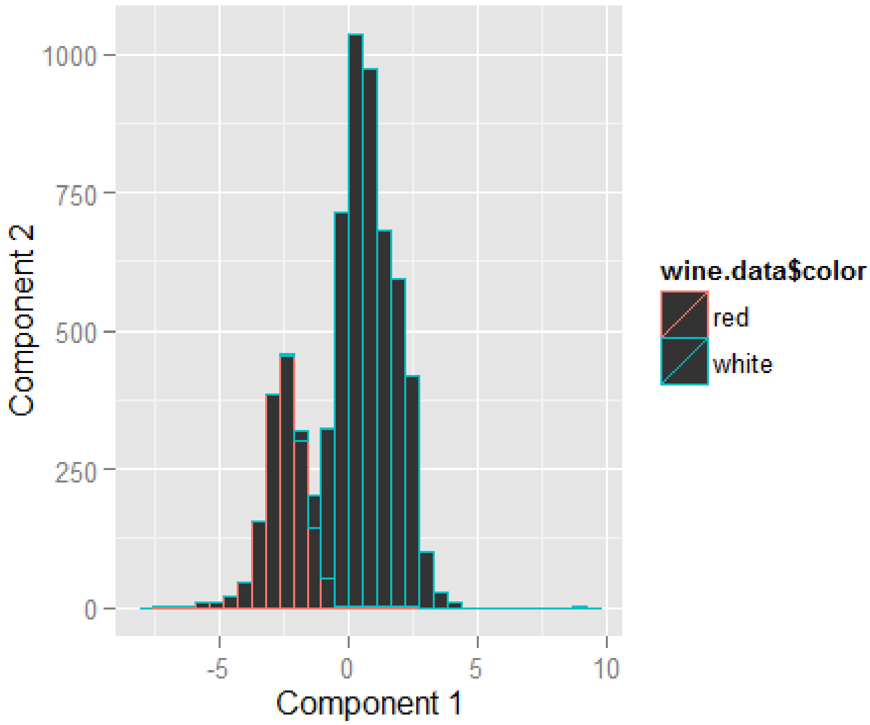

```
# An informative biplot
loadings = pca.wine.data$rotation
scores = pca.wine.data$x
#head(scores)
#nrow(scores)
```

Plotting clusters around principal components

```
#Plot PC to determine wine-type capture
qplot(scores[,1], scores[,2], color=wine.data$color, xlab='Component 1',
ylab='Component 2')
```

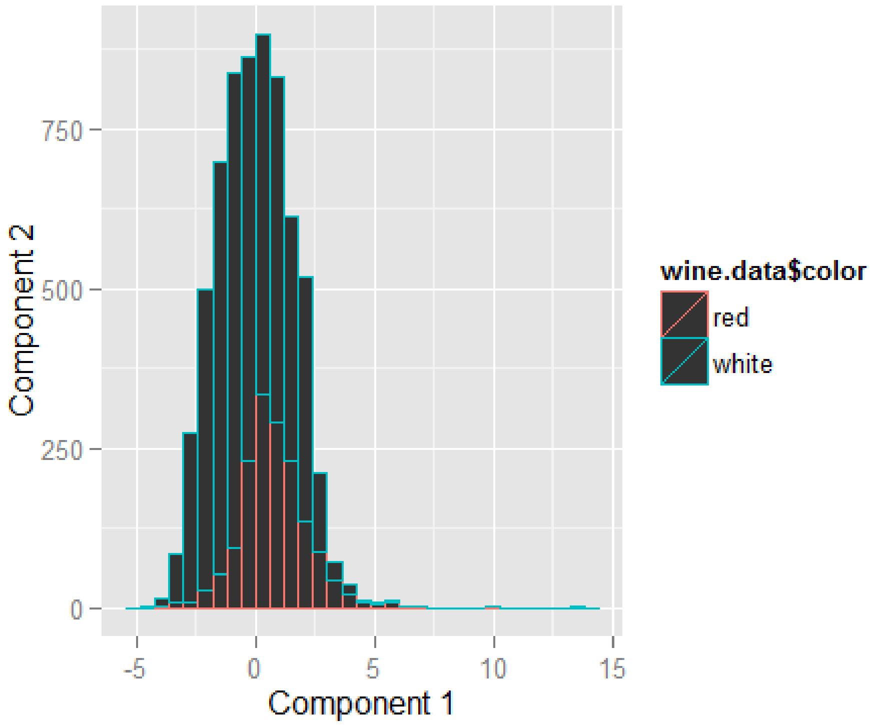



```
#Plotting PC1 vs wine type
qplot(scores[,1], color=wine.data$color, xlab='Component 1', ylab='Component
2')

## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust
this.
```




```
#Plotting PC2 vs wine type
qplot(scores[,2], color=wine.data$color, xlab='Component 1', ylab='Component
2')
```

```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust
this.
```



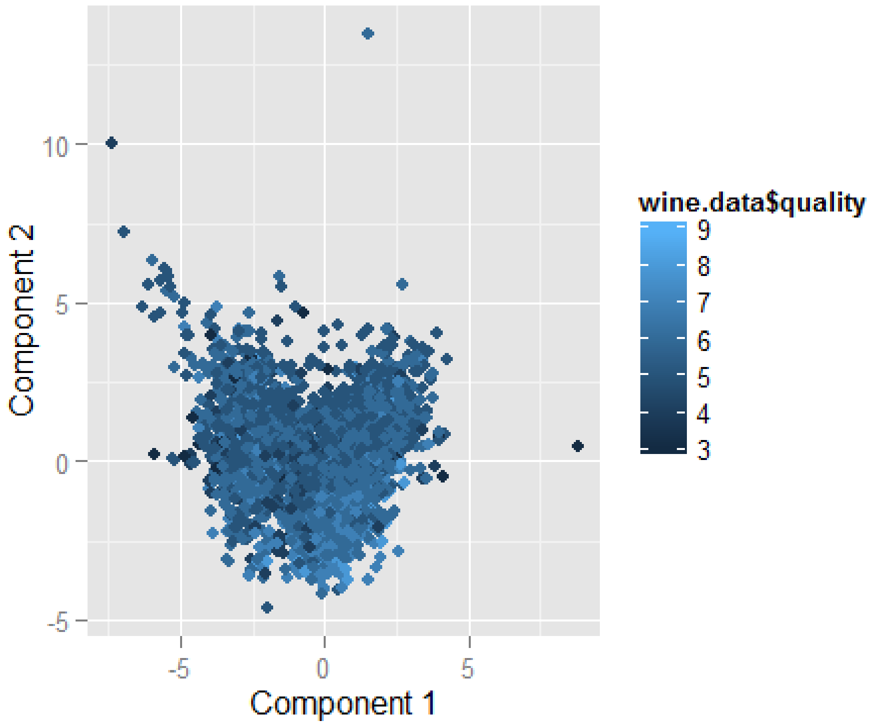
We notice that PC1 is segmenting red and white around a point, but PC2 alone is unable to do the same.

Following are the best and worst features of PC1:

```
o1 = order(loadings[,1])

#Best features
colnames(cluster.wine.data)[head(o1,3)]

## [1] "volatile.acidity" "sulphates"        "chlorides"

#Worst features
colnames(cluster.wine.data)[tail(o1,3)]

## [1] "residual.sugar"       "free.sulfur.dioxide"  "total.sulfur.dioxide"
```

What aboout wine quality?

```
#Plot PC to determine quality capture
qplot(scores[,1], scores[,2], color=wine.data$quality, xlab='Component 1',
ylab='Component 2')
```

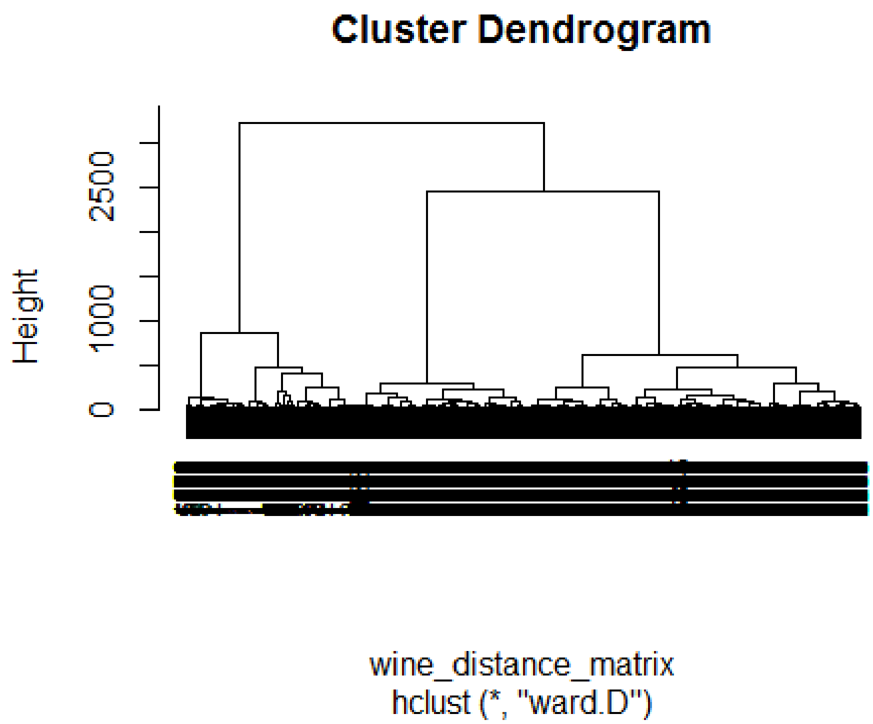


We notice that Principle component is unable to differentiate between different qualities of wine.

I have tried both heirarchical and K-means clustering technique on the same dataset to determine which technique would segregate the data better.

Heirarchical clustering

```
#Scaling the dataset
cluster.wine.data.scaled <- scale(cluster.wine.data, center=TRUE, scale=TRUE)
mu = attr(cluster.wine.data.scaled,"scaled:center")
sigma = attr(cluster.wine.data.scaled,"scaled:scale")

# Form a pairwise distance matrix using the dist function
wine_distance_matrix = dist(cluster.wine.data.scaled, method='euclidean')

# Now run hierarchical clustering
hier_wine = hclust(wine_distance_matrix, method='ward.D')

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




```
# Cut the tree into 4 clusters
cluster4 = cutree(hier_wine, k=4)
#summary(factor(cluster4))

# Cut the tree into 2 clusters
cluster2 = cutree(hier_wine, k=2)
#summary(factor(cluster2))

confusion_maxtrix.color = table(wine.data$color,cluster2)
round(prop.table(confusion_maxtrix.color, margin = 1),2)

##         cluster2
##             1    2
##   red    0.97 0.03
##   white 0.03 0.97

confusion_maxtrix.quality = table(wine.data$quality,cluster4)
round(prop.table(confusion_maxtrix.quality, margin = 1),2)

##    cluster4
##         1    2    3    4
##   3 0.20 0.20 0.10 0.50
##   4 0.17 0.14 0.13 0.56
##   5 0.12 0.21 0.33 0.33
##   6 0.08 0.15 0.25 0.51
##   7 0.03 0.15 0.11 0.71
```

```
##    8 0.02 0.08 0.11 0.79
##    9 0.00 0.00 0.20 0.80
```

K-means clustering

```
#Creating 2 clusters for white and red
cluster.k.wine <- kmeans(cluster.wine.data.scaled, centers=2, nstart=50)

#Creating 3 clusters to maybe help classify quality in low,medium and high
cluster.k.wine3 <- kmeans(cluster.wine.data.scaled, centers=3, nstart=50)

# A few plots with cluster membership shown
qplot(volatile.acidity,sulphates, data=wine.data,
color=factor(cluster.k.wine$cluster))
```


```
#qplot(volatile.acidity, chlorides, data=wine.data,
color=factor(cluster.k.wine$cluster))
#qplot(color, quality, data=wine.data, color=factor(cluster.k.wine$cluster))

#qplot(wine.data$color, cluster.k.wine$cluster, data=wine.data,
color=factor(cluster.k.wine$cluster))

confusion_maxtrix.color.k = table(wine.data$color,cluster.k.wine$cluster)
round(prop.table(confusion_maxtrix.color.k, margin = 1),2)

##
##              1     2
```

```
##   red   0.02 0.98
##   white 0.99 0.01
```

```
confusion_maxtrix.quality.k =
table(wine.data$quality,cluster.k.wine3$cluster)
round(prop.table(confusion_maxtrix.quality.k, margin = 1),2)
```

```
##
##        1    2    3
##   3 0.27 0.33 0.40
##   4 0.46 0.31 0.22
##   5 0.30 0.33 0.38
##   6 0.48 0.22 0.30
##   7 0.69 0.17 0.15
##   8 0.77 0.08 0.16
##   9 0.80 0.00 0.20
```

```
#qplot(color, quality, data=wine.data,
color=factor(cluster.k.wine3$cluster),cex= 1.2)
```

Although PCA segregates wine type well i.e. wine color to red and white , K-means is classifying around 99% of data points correctly.

K-means is computationally faster, starts by clustering around a centriod rather than using a bottom-up approach like heirarchical and shows excellent classification with just 2 clusters. Hence in this partcular case, K-means is a better classification technique to opt for over both PCA and heirarchical clustering. However, we can use PCA before clustering.

Though the K-means as well as other techniques classify wine-type (color) well, neither of the techniques are capable to sort out the better quality wines from lower quality wines

Question 4 - Market Segmentation

Objective:

1.Your task to is analyze this data as you see fit, and to prepare a report for NutrientH20 that identifies any interesting market segments that appear to stand out in their social-media audience. You have complete freedom in deciding how to pre-process the data and how to define "market segment." (Is it a group of correlated interests? A cluster? A latent factor? Etc.) Just use the data to come up with some interesting, well-supported insights about the audience.

```
social.data <- read.csv("C:/Users/Dhwani/Documents/Coursework/Summer -
Predictive Analytics/STA380/STA380/data/social_marketing.csv")
#names(social.data)
#Removing chatter, photo sharing, uncategorized,spam and adult
social.data.clean <- social.data[,-c(2,5,6,36,37)]
names(social.data.clean)
```

```
##  [1] "X"                "current_events"   "travel"
##  [4] "tv_film"          "sports_fandom"    "politics"
##  [7] "food"             "family"           "home_and_garden"
## [10] "music"            "news"             "online_gaming"
## [13] "shopping"         "health_nutrition" "college_uni"
## [16] "sports_playing"   "cooking"          "eco"
## [19] "computers"        "business"         "outdoors"
## [22] "crafts"           "automotive"       "art"
## [25] "religion"         "beauty"           "parenting"
## [28] "dating"           "school"           "personal_fitness"
## [31] "fashion"          "small_business"
```

```
#Scaling the entire data
social.data.clean.scaled <- scale(social.data.clean[,-c(1)])
```

Exploring the data

```
#Summing tweets per person
social.data.clean$tweets.tot <- rowSums (social.data.clean[,-c(1)], na.rm =
FALSE, dims = 1)

#Ordering the data set in the order of highest tweets to the lowest
social.data.ordered <- social.data.clean[order(-
social.data.clean$tweets.tot),]

#Deciling the dataset
library(dplyr)
social.data.ordered$decile <- ntile(social.data.ordered$tweets.tot, 10)

#Taking a cumulative sum of total tweets to pull of the majority of tweeters
social.data.ordered <- within(social.data.ordered, acc_sum <-
cumsum(tweets.tot))
```

We observe that people who fall in Top 5 deciles have tweeted around 70% of the total tweets.

Now we just tried exploring user profiles:

```
#Converting the dataset in relevant format
aggdata <-as.data.frame(aggregate(social.data.ordered[,-
c(1,34:36)],by=list(social.data.ordered$decile),FUN=sum,na.rm=TRUE))

aggdata2<- t(aggdata)

colnames(aggdata2) = aggdata2[1, ] # the first row will be the header
aggdata2 = aggdata2[-1, ]

aggdata.rowsum <- as.data.frame(round(aggdata2/rowSums(aggdata2),3)*100)
aggdata.colsum <-
as.data.frame(round(t(t(aggdata2)/colSums(aggdata2)),3)*100)
```

```r
rownames(aggdata.colsum)[which.max(aggdata.colsum$`10`)]

## [1] "tweets.tot"
```

People who tweet the most tweet about health & nutrition

```r
rownames(aggdata.colsum)[which.min(aggdata.colsum$`10`)]

## [1] "small_business"
```

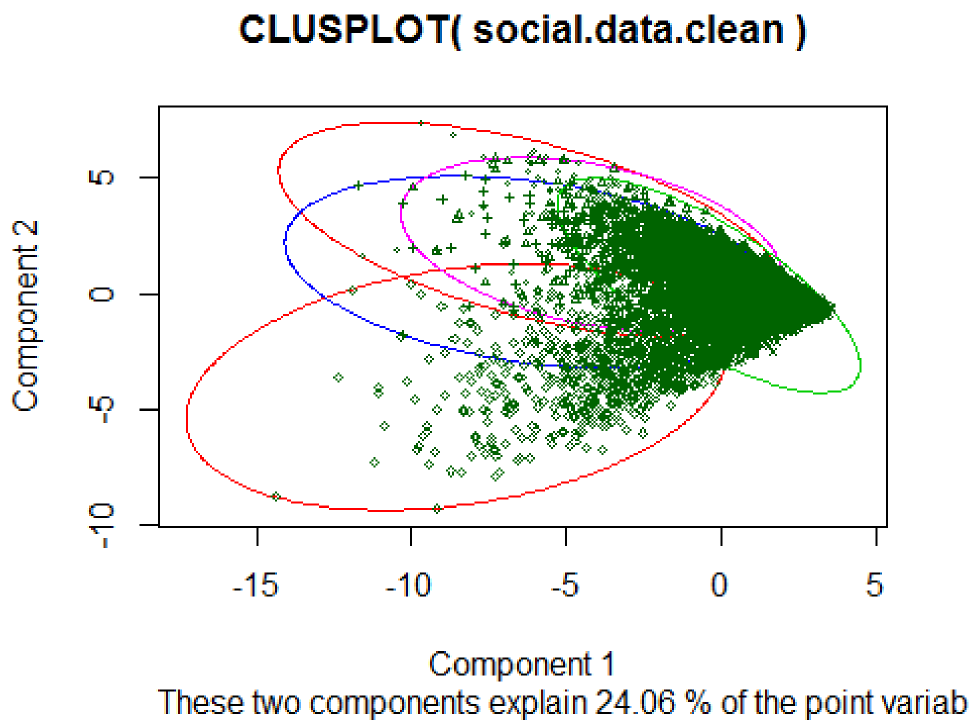
People who tweet the most tweet least about small businesses

```r
rownames(aggdata.colsum)[which.max(aggdata.colsum$`1`)]

## [1] "tweets.tot"
```

People who tweet the least mostly tweet on current events

After trying multiple clusters at random, I observed that adding clusters was segregating the smaller clusters more than the main cluster. Hence, 5 clusters seemed optimum:

```r
set.seed(1335)
k.social <- kmeans(social.data.clean.scaled, centers= 5, nstart=50)

k.social$size

## [1]  606  940  718 4834  784

#One cluster is predominantly dominant over others

library(ggplot2)
library(cluster)
clusplot(social.data.clean,k.social$cluster,color =
TRUE,shade=FALSE,labels=0,lines=0, cex = 0.4)
```




```
rbind(k.social$center[1,],(k.social$center[1,]*sigma + mu))

## Warning in k.social$center[1, ] * sigma: longer object length is not a
## multiple of shorter object length

## Warning in k.social$center[1, ] * sigma + mu: longer object length is not
a
## multiple of shorter object length

##      current_events     travel    tv_film sports_fandom    politics
## [1,]      0.1613699 -0.0537983 0.04616514    -0.2115064 -0.14313118
## [2,]      7.4245125  0.3308088 0.32534183     4.4369292  0.05101946
##            food      family home_and_garden    music        news
## [1,] -0.1876208   0.0138367       0.1444061 0.582136 -0.09076633
## [2,] 27.1951625 116.5266502       0.9951297 3.312101  0.51776171
##      online_gaming  shopping health_nutrition college_uni sports_playing
## [1,]    0.09851928 0.2864935      -0.09313024   0.1418344       0.276841
## [2,]   10.60930594 7.5867269       0.32433337   0.3392443       6.760390
##        cooking          eco    computers  business    outdoors    crafts
## [1,] 2.4912035  0.009737842   0.03730158 0.2681596 0.02551667 0.1118958
## [2,] 0.1433097 30.698160226 117.85292875 0.9955008 3.22260360 0.5479190
##      automotive       art   religion    beauty   parenting     dating
## [1,]   0.018241 0.1465905 -0.1403125 2.3532461 -0.09401199 0.17796345
## [2,]  10.513557 7.4053520  0.3165654 0.6606019  4.99594473 0.06226856
##          school personal_fitness  fashion small_business
## [1,]  0.1635566      -0.06861497 2.438657      0.2620961
## [2,] 33.4283511     111.86632893 1.002009      3.2606425
```

```r
#sports, personal fitness, religion

rbind(k.social$center[2,],(k.social$center[2,]*sigma + mu))

## Warning in k.social$center[2, ] * sigma: longer object length is not a
## multiple of shorter object length

## Warning in k.social$center[2, ] * sigma + mu: longer object length is not
a
## multiple of shorter object length

##      current_events     travel     tv_film sports_fandom   politics
## [1,]     0.01283864 -0.1484379 -0.03211116    -0.2074429 -0.1819128
## [2,]     7.23195150  0.3152277  0.31396689     4.4562627  0.0496608
##           food       family home_and_garden      music       news
## [1,]  0.406442  -0.05827028       0.1726535 0.07796514 -0.0451705
## [2,] 37.739420 112.45102993       0.9952144 3.23103664  0.5245466
##      online_gaming   shopping health_nutrition college_uni sports_playing
## [1,]   -0.01634873 0.05410403        2.0638945 -0.08207284     0.05035279
## [2,]   10.47230151 7.28544936        0.6794583  0.30670657     5.68280403
##         cooking        eco    computers   business outdoors    crafts
## [1,] 0.37643154  0.5271735  -0.08039193 0.06783848 1.572934 0.1108212
## [2,] 0.06922161 39.8823334 111.20067372 0.99490006 3.471409 0.5477591
##      automotive       art   religion     beauty  parenting     dating
## [1,] -0.1153058 0.0226354 -0.1776075 -0.2025606 -0.1151749 0.18072718
## [2,] 10.3542743 7.2446524  0.3104253  0.2891975  4.8952558 0.06236539
##         school personal_fitness    fashion small_business
## [1,] -0.151873         2.027776 -0.1046019    -0.05534008
## [2,] 27.829664       230.358230  0.9943830     3.20960287

#school, cooking

rbind(k.social$center[3,],(k.social$center[3,]*sigma + mu))

## Warning in k.social$center[3, ] * sigma: longer object length is not a
## multiple of shorter object length

## Warning in k.social$center[3, ] * sigma + mu: longer object length is not
a
## multiple of shorter object length

##      current_events    travel    tv_film sports_fandom  politics
## [1,]       0.105529 1.7409808 0.08692995     0.1827216 2.3193817
## [2,]       7.352118 0.6262949 0.33126569     6.3125888 0.1372902
##            food       family home_and_garden       music      news
## [1,]  0.02400569   0.05379313       0.1420787 -0.04288456 1.9030125
## [2,] 30.95140592 118.78506201       0.9951227  3.21160556 0.8144477
##      online_gaming   shopping health_nutrition college_uni sports_playing
## [1,]   -0.01033381 0.01650327       -0.2036036  0.04060919     0.06720283
## [2,]   10.47947557 7.23670246        0.3061454  0.32453446     5.76297324
##         cooking        eco  computers   business outdoors    crafts
## [1,] -0.2016994  0.1269562   1.551596 0.3562696 0.112900 0.1647295
```

```
## [2,]  0.0489676 32.7787155 203.443648 0.9957650 3.236654 0.5557810
##       automotive        art    religion     beauty   parenting     dating
## [1,]   1.097115 0.03382982 -0.04255192 -0.1660297 0.006873036 0.20129509
## [2,]  11.800343 7.25916519  0.33266040  0.2945061 5.475935894 0.06308595
##           school personal_fitness    fashion small_business
## [1,] -0.03306206       -0.1938447 -0.1558047      0.2535001
## [2,] 29.93848765      104.7881148  0.9942294      3.2592604
```

```
#travel, sports fandom,eco and craft

rbind(k.social$center[4,],(k.social$center[4,]*sigma + mu))
```

```
## Warning in k.social$center[4, ] * sigma: longer object length is not a
## multiple of shorter object length

## Warning in k.social$center[4, ] * sigma + mu: longer object length is not
a
## multiple of shorter object length

##      current_events     travel     tv_film sports_fandom    politics
## [1,]    -0.05737901 -0.2067213 -0.01680677    -0.2824514 -0.25785864
## [2,]     7.14091898  0.3056321  0.31619089     4.0993869  0.04700014
##            food     family home_and_garden       music       news
## [1,] -0.3472494  -0.231229      -0.1027661 -0.09652111 -0.2499751
## [2,] 24.3618513 102.675084       0.9943885  3.20298149  0.4940705
##      online_gaming    shopping health_nutrition  college_uni
## [1,]   -0.01212246 -0.05697713       -0.3344496 -0.007225035
## [2,]   10.47734223  7.14143999        0.2846034  0.317583289
##      sports_playing     cooking         eco   computers   business
## [1,]    -0.08223383 -0.33807606 -0.1535912  -0.2325052 -0.1180044
## [2,]     5.05198294  0.04418984 27.7991682 102.6029508  0.9943428
##        outdoors     crafts automotive         art   religion     beauty
## [1,] -0.3159804 -0.1723788 -0.1690755 -0.04516238 -0.2950642 -0.2797440
## [2,]  3.1676952  0.5056173 10.2901425  7.15675703  0.2910877  0.2779814
##       parenting      dating     school personal_fitness    fashion
## [1,] -0.2990694 -0.09393115 -0.2494853       -0.3396096 -0.2651388
## [2,]  4.0203216  0.05274312 26.0971049       96.5492076  0.9939016
##      small_business
## [1,]    -0.07892851
## [2,]     3.20581015
```

```
#No distinct interests. Talk about everything - Generic cluster

rbind(k.social$center[5,],(k.social$center[5,]*sigma + mu))
```

```
## Warning in k.social$center[5, ] * sigma: longer object length is not a
## multiple of shorter object length

## Warning in k.social$center[5, ] * sigma + mu: longer object length is not
a
## multiple of shorter object length
```

```
##      current_events     travel    tv_film sports_fandom    politics
## [1,]      0.1170176 -0.1002553 0.02683243       1.98641 -0.20547433
## [2,]      7.3670126  0.3231603 0.32253245      14.89418  0.04883536
##           food    family home_and_garden      music        news
## [1,]  1.776799   1.43562       0.1848912 0.09095982 -0.07719239
## [2,] 62.062430 196.88850       0.9952511 3.23312602  0.51978160
##      online_gaming   shopping health_nutrition  college_uni sports_playing
## [1,]    0.02765916 0.04987913       -0.1539634 -0.003870875      0.1711345
## [2,]   10.52479024 7.27997205        0.3143180  0.318070708      6.2574596
##          cooking        eco    computers  business    outdoors    crafts
## [1,] -0.10769786  0.1911492   0.08016308 0.1127023 -0.06075795 0.6926308
## [2,]  0.05226082 33.9181021 120.27554047 0.9950346  3.20873175 0.6343358
##      automotive       art  religion    beauty parenting     dating
## [1,]  0.1618817 0.1070332 2.1796851 0.3008058  2.048473 0.04056622
## [2,] 10.6848791 7.3540685 0.6985217 0.3623457 15.189466 0.05745504
##         school personal_fitness    fashion small_business
## [1,]  1.624229       -0.1067286 0.01791842      0.1182613
## [2,] 59.354416      109.7120765 0.99475037      3.2375157

#food and dating
```

On observing the data, we come across 5 distinct segments. Though we have a huge over-powering generic segment, we have 4 distinctly classfied segments who NutrientH20 can target.

We see sports players and people who focus on personal fitness in one segment. NutrientH20 can direct sports and fitness related campaigns to this cluster.

The second cluster mainly comprises of people who tweet about school and cooking. They seem to be families. NutrientH20 can direct family, parenting related or cooking related campigns to members in this cluster.

The third distinct cluster mainly comprises of travellers and sports fans. They could be sports fans who traveled to watch sports or just travellers. NutrientH20 has a good base to direct all their travel, sports mascots, camping, hiking campaigns.

The fourth cluster mostly tweets about food and dating. It would be safe to assume that this cluster comprises of unmarried people who would be in their twenties or thirties. This cluster seems to be more open to trying out new restaurents in the town. NutrientH20 can target this cluster with gifting ideas or suggestions to new places and restaurents in town. They can also target this segment with new products which appeal to the youth.

The fifth and the final cluster is generic and people comrising this segment do not talk about anything specific.