

# Assignment\_PM

Sanchit Shaleen

August 7, 2015

## Q1 Exploratory analysis

```
library(ggplot2)
dataGeorgia<-read.csv("F:/Predictive
Modelling(JScott)/PredictiveModelling/STA380/data/georgia2000.csv")
dataGeorgia[, "VoteUnderCount"]<-(dataGeorgia$ballots - dataGeorgia$votes)
dataGeorgia$poor<-as.factor(dataGeorgia$poor)
```

### Part (a)

Whether voting certain kinds of voting equipment lead to higher rates of undercount?

Applying aggregate function to calculate the total ballot count and vote under-count over different equipments

```
nuc<-aggregate(dataGeorgia$VoteUnderCount ~ dataGeorgia$equip, data =
dataGeorgia, sum)
nbc<-aggregate(dataGeorgia$ballots ~ dataGeorgia$equip, data = dataGeorgia,
sum)
```

Merging the aggregated data

```
ndf<-merge(nbc,nuc,by=intersect(names(nbc), names(nuc)))
ndf$VoteUnderCountPercent<-ndf[,3]/ndf[,2] *100
```

Distribution of voting across equipments

As per the analysis, there seems to be a higher under-count percentage when the voting is done using Punch Device.

### Part(b) Whether we should worry that this effect has a disparate impact on poor and minority communities?

**\*\* Impact on Poor \*\***

```
uc<-aggregate(dataGeorgia$VoteUnderCount ~
dataGeorgia$equip+dataGeorgia$poor, data = dataGeorgia, sum)
bc<-aggregate(dataGeorgia$ballots ~ dataGeorgia$equip+dataGeorgia$poor, data
= dataGeorgia, sum)
```

Merging the aggregated data into a new data frame and adding a new column named UnderCountPercent to it.

```
df = data.frame()
df<-merge(bc,uc,by=intersect(names(bc), names(uc)))
df$VoteUnderCountPercent<-df[,4]/df[,3] *100
```

Analysis: There seems to be a higher under-count percentage amongst poor people using Optical device for casting their vote

**\*\* Impact on Minority \*\*** There's a weak co-relation between Minority community and Vote undercount

```
qplot(dataGeorgia$VoteUnderCount/dataGeorgia$ballots,dataGeorgia$perAA,geom=c
("point","smooth"),method="lm",
xlab="Vote Under Count Fraction",ylab="Minority Fraction")
```

## Ques 2 Bootstrapping

Part(a)

```
library(mosaic)
library(fImport)
library(foreach)
set.seed(10)
```

Import a few stocks

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

A helper function for calculating percent 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)
```

View the returns as draws from the joint distribution

```
pairs(myreturns)
```

```
plot(myreturns[,1], type='l')
```

```
mu_SPY = mean(myreturns[,4]) sigma_SPY = sd(myreturns[,4])
```

Calculate the standard deviation across different different stock returns

```
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

Plot for variability in returns for each stock over the period from='2011-01-01' to='2015-08-05'

## Part(b)

After analysing the Standard Deviation of the risk/return properties of the 5 mentioned ETF's LQD and and SPY safe stocks to purchase since they have smaller standard deviations.

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

Below are the values of standard deviations for the 5 stocks mentioned. The standard deviation values helps in characterizing the risk/return properties for these stocks

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

## SPY.PctReturn TLT.PctReturn LQD.PctReturn EEM.PctReturn VNQ.PctReturn
## 0.009405747 0.009593108 0.003490087 0.013843982 0.011444111
```

## Part(c)

Bootstrap resampling to estimate the 4-week (20 trading day) value at risk of each of your three portfolios.

### (i) Equal Split Portfolio

```
totalwealth = 100000
weights = c(0.20,0.20,0.20,0.20,0.20)
holdings = weights * totalwealth
n_days = 20
```

Using 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 from the empirical joint distribution This simulates a random day

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

Update the value of your holdings and compute your new total 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

```
set.seed(40)
n_days = 20
# Now simulate many different possible trading years!
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) # 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
  }
  wealthtracker
}
hist(sim1[,n_days], 20)
```

```
# Profit/Loss
hist(sim1[,n_days]- 100000,xlab="Equal Split Portfolio",main="Histogram for
return for Equal Split Portfolio")
```

```
# Calculate 5% value at risk
quantile(sim1[,n_days], 0.05) - 100000
```

```
##          5%
## -4771.615
```

**Analysis: The 5% risk associated with the portfolio (even split) is -3560.43**

**(ii) Safer Portfolio** Bootstrapping for safer portfolio. Now loop over two trading weeks. Considering the portfolio of SPY, TLT and LQD as a safe portfolio. Going by the sd values of the above stocks, which is in the ratio of 2:2:1, we would invest in these stocks in the inverse proportion (lower the sd value, higher the investment in that stock, considering it's a safe portfolio). So the investment in this portfolio would be in the ratio of 1:1:2 for SPY, TLT and LQD resp.

```
set.seed(40)
n_days = 20

## Now simulate many different possible trading years!
sim1 = foreach(i=1:5000, .combine='rbind') %do% {
  totalwealth = 100000
  weights = c(0.25, 0.25, 0.50, 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
  }
  wealthtracker
}

hist(sim1[,n_days], 20)
```

```
## Profit/loss
hist(sim1[,n_days] - 100000, xlab="Safer Portfolio", main="Histogram for return
for Safe Portfolio")
```

```
## Calculate 5% value at risk
quantile(sim1[,n_days], 0.05) - 100000

##          5%
## -5716.463
```

**Analysis: The 5% risk associated with the portfolio (SPY, TLT and LQD) is -2217.123**

**(iii) Riskier Portfolio**

Bootstrapping for riskier portfolio. Now loop over two trading weeks. Considering the portfolio of EEM and VNQ as a risky portfolio. Going by the sd values of the above stocks, which is in the ratio of 6:5, we would invest in these stocks in the direct proportion (higher the sd value, higher the investment in that stock, considering it's a risky portfolio)

```
## Now simulate many different possible trading years!
set.seed(20)
n_days = 20
sim1 = foreach(i=1:5000, .combine='rbind') %do% {
  totalwealth = 100000
  weights = c(0.54, 0.46, 0,0,0)
  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
  }
  wealthtracker
}

hist(sim1[,n_days], 20)
```

```
##Profit/loss
hist(sim1[,n_days]- 100000,xlab="Riskier Portfolio",main="Histogram for
return for Riskier Portfolio")
```

```
## Calculate 5% value at risk
quantile(sim1[,n_days], 0.05) - 100000

##          5%
## -5889.972
```

**Analysis:**The 5% risk associated with the portfolio (EEM and VNQ) -5889.972

## Ques 3 Clustering and PCA

Applying K-Means Clustering Technique for distinguishing between colors of wine

Reading the data

```
wine<-read.csv("F:/Predictive
Modelling(JSScott)/PredictiveModelling/STA380/wine.csv")
wine$quality<-as.factor(wine$quality)
wine_scaled <- scale(wine[,c(12,13)], center=TRUE, scale=TRUE)
```

Transforming and scaling data

```
wine_scaled <- scale(wine[, -c(12,13)], center=TRUE, scale=TRUE)
df_color <- wine$color
```

Finding the optimum value of number of clusters(k) using the value of CH- Index

```
n <- dim(wine_scaled)[2]
ch_index = numeric(length = 15)
for(i in 2:15){
  cluster_all = kmeans(wine_scaled, centers=i, nstart=20)
  ch_index[i-1] = (sum(cluster_all$betweenss)/(i-1))/
    (sum(cluster_all$withinss)/(n-i))
}
plot(2:15, ch_index[1:14], xlab='K', ylab='CH(K)', type='b', main=' CH Index
versus Number of Clusters(K) ' )
```

Since there is a rise in the value at k=2 and a constant decline thereafter, value of k should be 2

Apply K-Means clustering for distinguishing between the 2 colors

```
set.seed(10)
clusterall <- kmeans(wine_scaled, centers=2, nstart=20)
```

Comparing the results of the clustering with the original clustered data on the basis of color

```
table(df_color, cluster_all$cluster)

##
## df_color    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15
##   red       8 291    4    2    0   34 422    8 538   24    4    4 256    1    3
##   white    91   22 690 568 526 501   15 438   12    0 598 605    2 591 239
```

Applying K-Means Clustering Technique for distinguishing amongst the 7 qualities of wine

```
cluster_all <- kmeans(wine_scaled, centers=7, nstart=50)
```

```
## Warning: did not converge in 10 iterations
```

```
df_quality = as.factor(wine$quality)
table(df_quality, cluster_all$cluster)

##
## df_quality    1    2    3    4    5    6    7
##           3    4    7    2    1    5    4    7
##           4   21   24   27    2   64   15   63
##           5   77  655  269   20  446  200  471
##           6  548  640  475    9  549  265  350
##           7  446  122  189    1  137  141   43
##           8   97   22   31    0   27   14    2
##           9    4    0    0    0    1    0    0
```

Analysis of K-Means technique for our problem: The technique was successful in distinguishing between the 2 colors while performed reasonably well while distinguishing amongst the qualities of wine

Using PCA to solve the problem

```
wine<-read.csv("F:/Predictive
Modelling(JScott)/PredictiveModelling/STA380/wine.csv")
wine_scaled <- scale(wine[,c(12,13)], center=TRUE, scale=TRUE)
```

**\*\* Applying PCA\*\***

```
pc1 <-prcomp(wine_scaled)
```

Look at the basic plotting and summary methods

```
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

plot(pc1)
```

```
biplot(pc1)
```

A more informative biplot for color and quality features

```
library(ggplot2)
loadings = pc1$rotation
scores = pc1$x

qplot(scores[,1], scores[,2], color=wine$color, xlab='Component 1',
ylab='Component 2')
```

```
wine$quality<-as.factor(wine$quality)

qplot(scores[,1], scores[,2], color=wine$quality, xlab='Component 1',
ylab='Component 2')
```



## Ques 4 Market segmentation

```
twitter_data<-read.csv("F:/Predictive  
Modelling(JSScott)/PredictiveModelling/STA380/data/social_marketing.csv")  
twitter_data<-twitter_data[,-c(1,6)]  
twitter_data_scaled <- scale(twitter_data, center=TRUE, scale=TRUE)
```

Finding the right number of clusters from CH Index value by analysing the graph

```
ch_index = numeric(length = 14)  
for(i in 2:15){  
  cluster_all = kmeans(twitter_data_scaled, centers=i, nstart=30)  
  ch_index[i-1] = (sum(cluster_all$betweenss)/(i-  
1))/(sum(cluster_all$withinss)/(n-i))  
}  
  
## Warning: did not converge in 10 iterations  
## Warning: did not converge in 10 iterations  
  
plot(2:15, ch_index[1:14], xlab='K', ylab='CH(K)', type='b', main=' CH Index  
versus Number of Clusters(K)' )
```

Run the K-Means clustering algorithm over the optimal number of clusters found above viz.  
6

```
set.seed(10)  
cluster_new <- kmeans(twitter_data, centers=6, nstart=20)
```

Get all the clusters sequentially and analyse each cluster to find which market segment is being represented by each of them. We do this, by first adding the cluster field to our original data frame and then calculating the mean over the values present under each column within each cluster.

```
twitter_data$cluster1<-cluster_new$cluster
```

**Cluster 1:** Studying this group, what we infer is that, this group is health conscious, cares about personal fitness, health nutrition, food, love outdoor activities and interacting with people. So mainly this group comprises of young men and women probably in the age group of 16-25 years

```
clust1= subset(twitter_data, cluster1==1)  
tail(sort(sapply(clust1[, -36], mean)))  
  
## health_nutrition      beauty      chatter      fashion  
##      2.013487      3.845857      3.940270      5.689788  
## photo_sharing      cooking  
##      5.899807      11.932563
```

**Cluster 2:** This group is religious,involved in parenting,cooking food,enjoys crafts,sports and chatting with people. So this group mainly comprises of people who are in the age group of 30-50 years.

```
clust1= subset(twitter_data, cluster1==2)
tail(sort(sapply(clust1[, -36],mean)))
```

##	politics	sports_fandom	current_events	shopping	photo_sharing
##	1.461778	1.623245	1.865835	3.543682	5.617005
##	chatter				
##	9.879875				

**Cluster 3:** This group loves to travel,share photos,watches movies as well.This segment mainly consists of children and the youth population in the age group of 5-22 years.

```
clust1= subset(twitter_data, cluster1==3)
tail(sort(sapply(clust1[, -36],mean)))
```

##	photo_sharing	computers	chatter	news	travel
##	2.267462	2.643952	4.044293	5.146508	6.347530
##	politics				
##	9.870528				

**Cluster 4:** This group loves to do shopping,has a taste in arts and keep themselves updated of all the latest events. The group probably comprises of mid-aged people in the age group of 30-55.

```
clust1= subset(twitter_data, cluster1==4)
tail(sort(sapply(clust1[, -36],mean)))
```

##	photo_sharing	outdoors	cooking	chatter
##	2.438384	2.438384	3.368687	4.022222
##	personal_fitness	health_nutrition		
##	6.107071	12.275758		

**Cluster 5:** This group has an interest in following politics and enjoying travelling as well.This segment of population mainly comprises of young men in the age group of 18-35 years.

```
clust1= subset(twitter_data, cluster1==5)
tail(sort(sapply(clust1[, -36],mean)))
```

##	sports_fandom	sports_playing	photo_sharing	chatter	online_gaming
##	1.623153	2.487685	2.640394	4.071429	10.450739
##	college_uni				
##	10.793103				

**Cluster 6:** This group is active in businesses and chattering with new clients.This is a professional group of people. The working class population dominates this group.

```
clust1= subset(twitter_data, cluster1==6)
tail(sort(sapply(clust1[, -36],mean)))
```

##	travel	food	current_events	photo_sharing	sports_fandom
##	1.115178	1.189605	1.378721	1.507077	1.517082
##	chatter				
##	2.916301				