STA380_Exercise1_Mayur

Mayur Srinivasan

7 August 2015

Q1 - Exploratory analysis

Data preparation from the given CSV

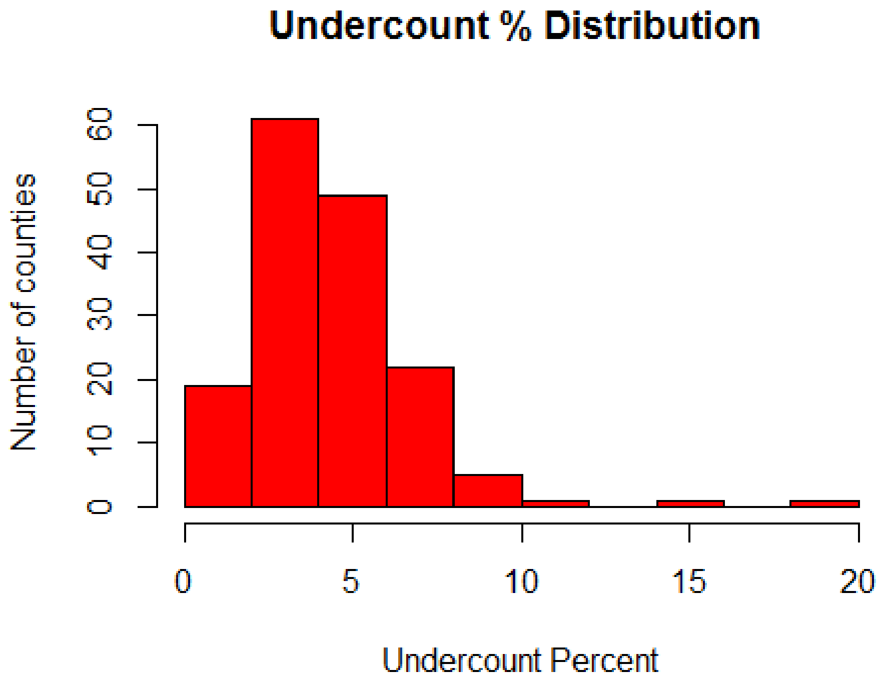
```
georgia = read.csv('../data/georgia2000.csv')

#Calculate the undercounts and the fraction of undercounts

georgia$underCount<-georgia$ballots-georgia$votes
georgia$underCountPerCent<-round(100*(georgia$underCount/georgia$ballots),2)
```

Preliminary summary of data

- There are a total of 159 counties
- Out of 2,691,314 ballots, 2,596,633 were counted leading to an undercount of 0% in Georgia
- The county of FULTON has the highest undercounts with 17,764 which constitutes to 6.32 % of the total ballots casted in the county

```
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          underCount      underCountPerCent
##  Min.   :   249   Min.   :   271   Min.   :    0.0   Min.   : 0.000
##  1st Qu.:  1386   1st Qu.:  1804   1st Qu.:  152.5   1st Qu.: 2.780  
##  Median :  2326   Median :  3597   Median :  296.0   Median : 3.980  
##  Mean   :  7020   Mean   :  8929   Mean   :  595.5   Mean   : 4.379  
##  3rd Qu.:  4430   3rd Qu.:  7468   3rd Qu.:  523.5   3rd Qu.: 5.650  
##  Max.   :154509   Max.   :140494   Max.   :17764.0   Max.   :18.810  
## 
```

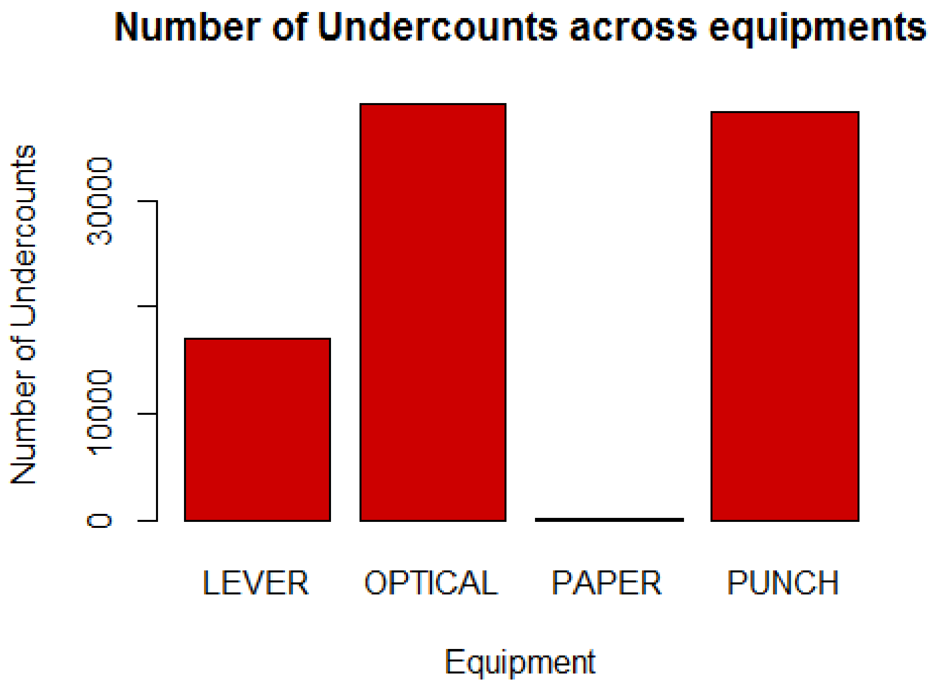
```
hist(georgia$underCountPerCent, main = "Undercount % Distribution ",
ylab="Number of counties",xlab = "Undercount Percent",col = "red")
```



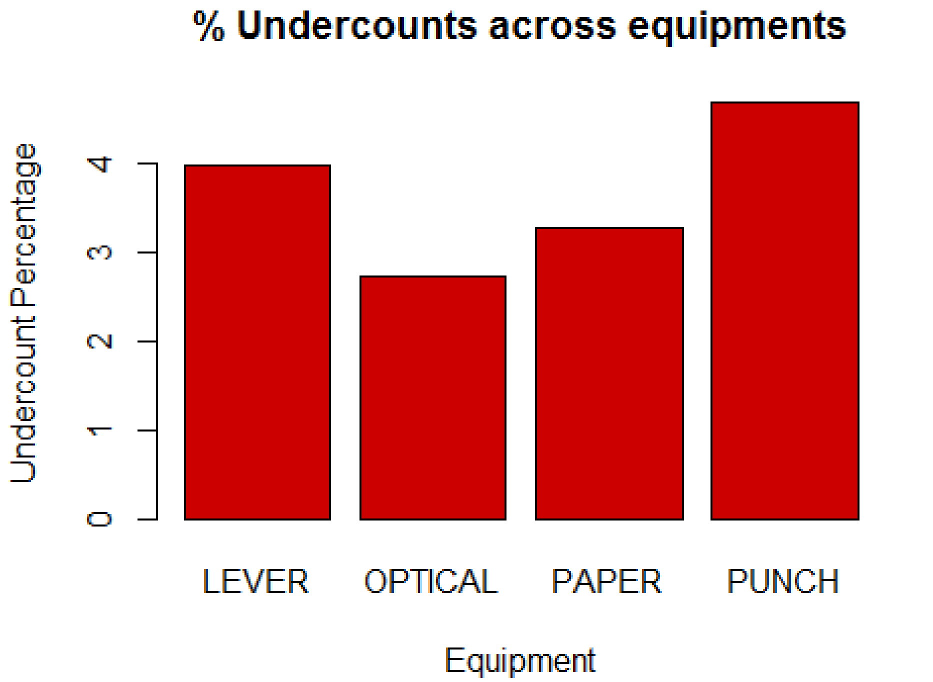
Is Undercount dependent on the type of equipment?

- An aggregate on Undercounts on the type of equipment will help us answer this

```
agg= aggregate(cbind(ballots,votes)~equip,data=georgia,sum)
agg$UndercountPerc<-100*(agg$ballots-agg$votes)/(agg$ballots)


barplot((agg$ballots-agg$votes),col="red3",main="Number of Undercounts across
equipments",names.arg = agg$equip,xlab = "Equipment",ylab = "Number of
Undercounts")
```



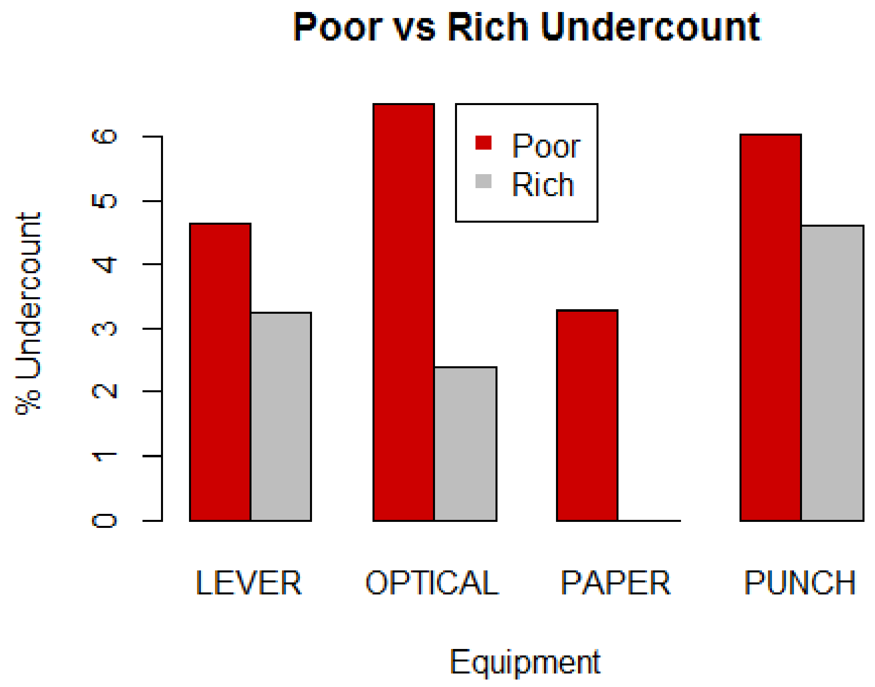
```r
barplot(agg$UndercountPerc,col="red3",main="% Undercounts across
equipments",names.arg = agg$equip,xlab = "Equipment",ylab = "Undercount
Percentage")
```



- We can observe the following from above:
  - 'Optical' has the highest number of vote Undercounts
  - 'Paper' has the lowest number of vote Undercounts
  - Undercounts as a percentage of total ballots gives us a more accurate view
  - 'Punch' and 'Lever' have the highest Undercount percentages

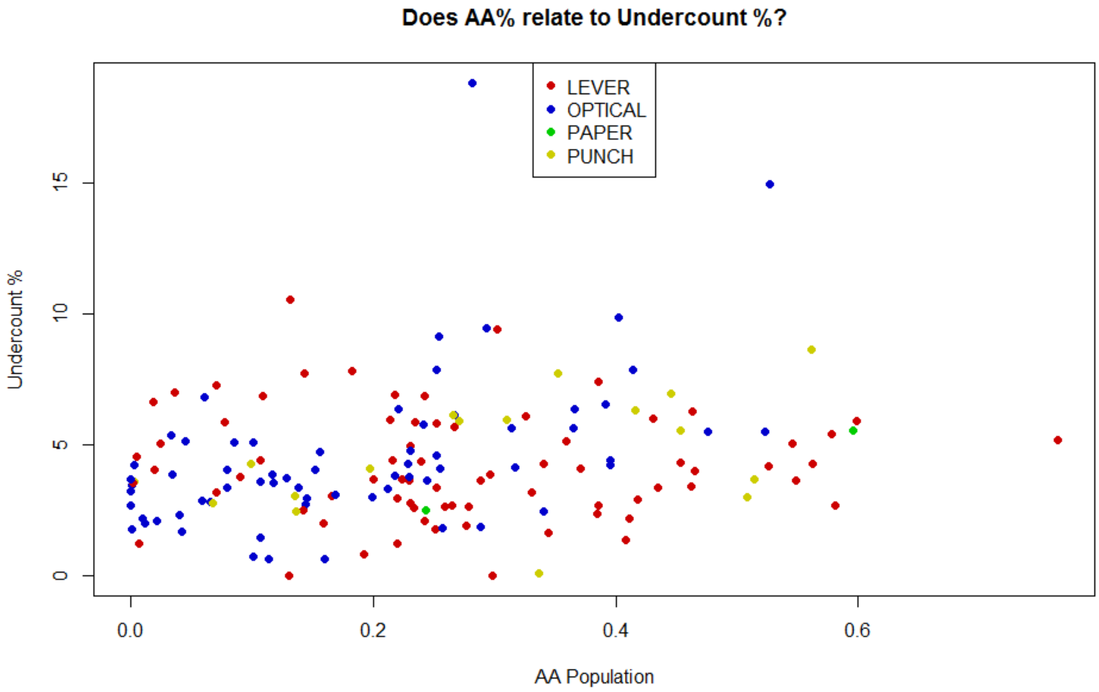
Is there a relation between Undercount % and the economic status of the counties?

```
poorg<-georgia[georgia$poor==1,]

econpoor=aggregate(cbind(ballots,votes)~equip,data=poorg,sum)
econpoor$UndercountPerc<-100*(econpoor$ballots-
econpoor$votes)/(econpoor$ballots)

richg<-georgia[georgia$poor==0,]

econrich=aggregate(cbind(ballots,votes)~equip,data=richg,sum)
econrich$UndercountPerc<-100*(econrich$ballots-
econrich$votes)/(econrich$ballots)

econrich=rbind(econrich,c("PAPER",0,0,0))
econrich=rbind(econrich[1:2,],econrich[4,],econrich[3,])

barplot(matrix(c(as.numeric(econpoor$UndercountPerc),as.numeric(econrich$Unde
rcountPerc)),nr=2, byrow =  TRUE), beside=T,
col=c("red3","grey"),names.arg=econpoor$equip,xlab="Equipment",ylab="%
Undercount",main="Poor vs Rich Undercount")

legend("top", c("Poor","Rich"), pch=15,
       col=c("red3","grey"))
```



- The Undercount percentage is higher for Poor counties than the Rich counties across all types of equipment. The effect is particularly pronounced for 'Optical' and 'Paper' equipment

```
attach(georgia)

plot(x=perAA,y=underCountPerCent,main="Does AA% relate to Undercount
%?",pch=19,col=c("red3","blue3","green3","yellow3")[equip],xlab="AA
Population",ylab="Undercount %")

legend(x="top", legend = levels(georgia$equip),
col=c("red3","blue3","green3","yellow3"), pch=19)
```



```
detach(georgia)
```

- We observe a very weak (but non-zero) correlation between the percentage of American Popuulations in counties and the corresponding Undercount % in those counties
- Further, we don't see a significant effect of the type of equipment on the Undercount % as a function of the AA Population %
- Counties with a higher proportion of AA population have more 'Lever' equipments than any other type of equipment

Question 2 - Bootstrapping

Data Preparation for the five given asset classes

The risk and return for every asset class can be gauged from the following metrics: * Average return on investment for that asset class alone * Value at risk

For example, bootstrapping the returns for **SPY** alone can be done as follows :

```
sim_SPY = foreach(i=1:500, .combine='rbind') %do% {
  totalwealth = 100000
  n_days = 20
  weights_even = c(1.0, 0.0, 0.0, 0.0, 0.0)
  holdings = weights_even * 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
    holdings = weights_even * totalwealth
  }
  wealthtracker
}
```

The Average return for SPY is 1.011651610^{5} and the (loss) value at risk is -5735.3024449.

When this exercise is repeated for all the asset classes individually, we see the following relative pattern:

Asset Class	Risk	Return
EEM	Very High	High
VNQ	High	High
SPY	Medium	Medium
TLT	Low	Medium
LQD	Very Low	Low

With the above learning, various portfolios can be created with varying proportions of the risky and safe assets

Even Split Portfolio

As the name suggests, the even splot portfolio will have an equitable distribution of the the everyday starting wealth across all five asset classes. The code will be similar to the one above for **SPY** alone, with changes only in the proportions of each asset class

```
sim_even = foreach(i=1:500, .combine='rbind') %do% {
  totalwealth = 100000
  n_days = 20
  weights_even = c(0.2, 0.2, 0.2, 0.2, 0.2)
  holdings = weights_even * 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
    holdings = weights_even * totalwealth
  }
  wealthtracker
}
```

The Average return for the even split portfolio is 1.006040410^{5} and the (loss) value at risk is -3795.9241169.

Safe Portfolio

A safe portfolio will feature the safe asset classes dominantly. To take an extreme example, and given the constraints to use at least three asset classes, we can take the top three safest classes with a heavy bias towards the safest of the three. Hence, assigning 10% each to SPY and TLT, and 80% to the safest class, LQD, we get the following

```
sim_safe = foreach(i=1:500, .combine='rbind') %do% {
  totalwealth = 100000
  n_days = 20
  weights_even = c(0.1, 0.1, 0.8, 0.0, 0.0)
  holdings = weights_even * 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
    holdings = weights_even * totalwealth
  }
  wealthtracker
}
```

The Average return for the even split portfolio is 1.006054510^{5} and the (loss) value at risk is -1960.8387685.

Risky Portfolio

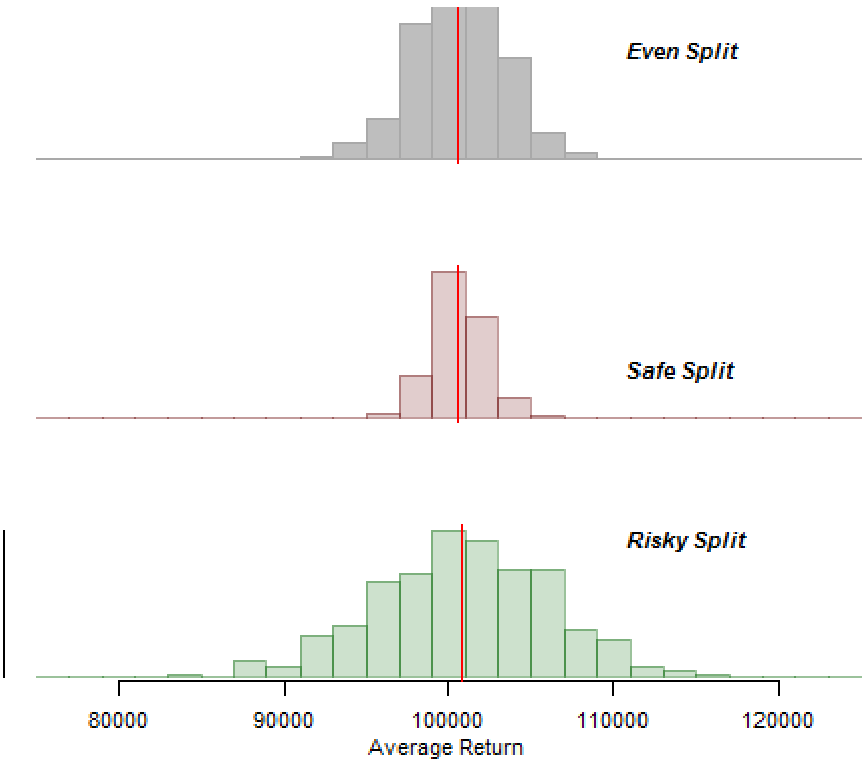
A risky portfolio will feature the high-risk asset classes dominantly. To take an extreme example, and given the constraints to use at least two asset classes, we can take the top two riskiest classes with a heavy bias towards the riskiest of the two. Hence, assigning 30% each to VNQ, and 70% to the riskiest class, EEM, we get the following

```
sim_risk = foreach(i=1:500, .combine='rbind') %do% {
  totalwealth = 100000
  n_days = 20
  weights_even = c(0.0, 0.0, 0.0, 0.7, 0.3)
  holdings = weights_even * 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
    holdings = weights_even * totalwealth
  }
  wealthtracker
}
```

The Average return for the even split portfolio is 1.008362110^{5} and the (loss) value at risk is -8214.7783554.

From the average return and value at risk values of each of the portfolios we can see that an aggressive/risky portfolio offers a marginally higher average return in the long run, but it also comes at the cost of a higher risk involved, as quanitfied by the high (loss) value at risk. On the other hand, a safe portfolio offers a conservative average return, but a lower value at risk.

The spread of returns as a result of the bootstrap corroborates the idea above. Below, we can see that the returns are more/less volatile around the average for the agressive/safe portoflio respectively.



The much larger tail of the risky split, characterises the volatility and high value at risk of such a portfolio. The even split presents the neutral spread of returns over time

Q3 - Clustering and PCA

Data preparation from the given CSV

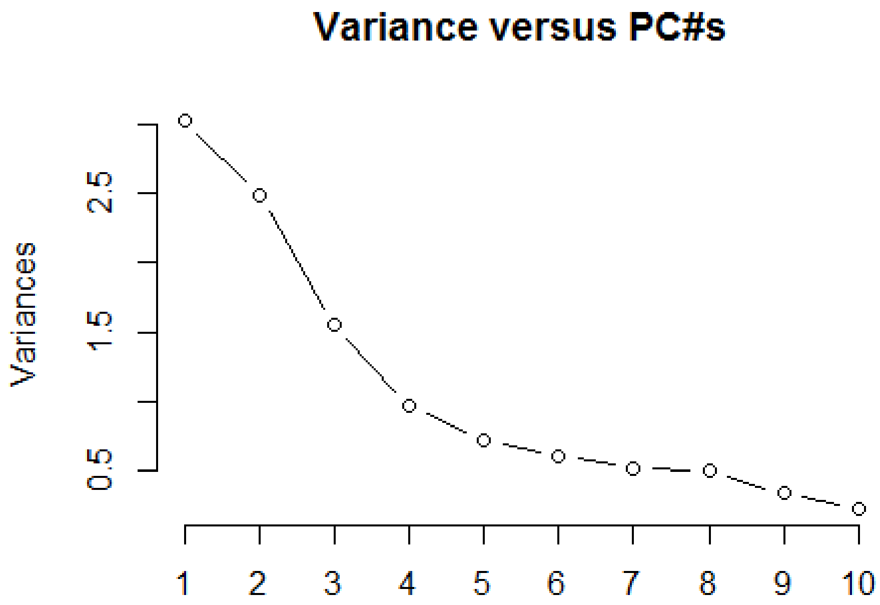
```
wine<- read.csv("../data/wine.csv")
Z = wine[,1:11]
```

Principal Component Analysis - Does it distinguish Red and White wines?

We will now run a Principal Component Analysis on the features of the dataset

We will now look at the summary of PCA

```
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

par( mfrow = c( 1,1  ) )
plot(pc1,type="line", main = "Variance versus PC#s")
```

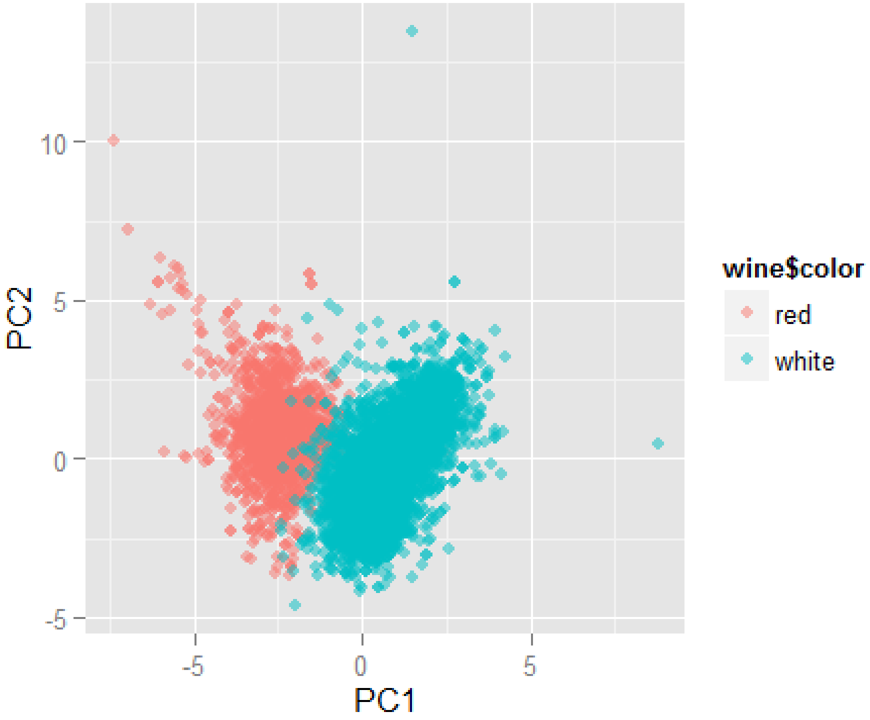


From the cumulative proportion of variance, we can infer that the first 4 principal components can explain 75% of the total variance of all variables. We can choose PC1 through PC4 to proceed further

We now obtain the loadings for each of the PCs

```
##                              PC1         PC2         PC3         PC4
## fixed.acidity        -0.23879890  0.33635454 -0.43430130  0.16434621
## volatile.acidity     -0.38075750  0.11754972  0.30725942  0.21278489
## citric.acid           0.15238844  0.18329940 -0.59056967 -0.26430031
## residual.sugar        0.34591993  0.32991418  0.16468843  0.16744301
## chlorides            -0.29011259  0.31525799  0.01667910 -0.24474386
## free.sulfur.dioxide   0.43091401  0.07193260  0.13422395 -0.35727894
## total.sulfur.dioxide  0.48741806  0.08726628  0.10746230 -0.20842014
## density              -0.04493664  0.58403734  0.17560555  0.07272496
## pH                   -0.21868644 -0.15586900  0.45532412 -0.41455110
## sulphates            -0.29413517  0.19171577 -0.07004248 -0.64053571
## alcohol              -0.10643712 -0.46505769 -0.26110053 -0.10680270
##                             PC5         PC6         PC7          PC8
## fixed.acidity        -0.1474804 -0.20455371 -0.28307944  0.401235645
## volatile.acidity      0.1514560 -0.49214307 -0.38915976 -0.087435088
## citric.acid          -0.1553487  0.22763380 -0.38128504 -0.293412336
## residual.sugar       -0.3533619 -0.23347775  0.21797554 -0.524872935
## chlorides             0.6143911  0.16097639 -0.04606816 -0.471516850
## free.sulfur.dioxide   0.2235323 -0.34005140 -0.29936325  0.207807585
## total.sulfur.dioxide  0.1581336 -0.15127722 -0.13891032  0.128621319
## density              -0.3065613  0.01874307 -0.04675897  0.004831136
## pH                   -0.4533764  0.29657890 -0.41890702 -0.028643277
## sulphates            -0.1365769 -0.29692579  0.52534311  0.165818022
## alcohol              -0.1888920 -0.51837780 -0.10410343 -0.399233887
##                             PC9         PC10          PC11
## fixed.acidity         0.3440567 -0.281267685 -0.3346792663
## volatile.acidity     -0.4969327  0.152176731 -0.0847718098
## citric.acid          -0.4026887  0.234463340  0.0011089514
## residual.sugar        0.1080032 -0.001372773 -0.4497650778
## chlorides             0.2964437 -0.196630217 -0.0434375867
## free.sulfur.dioxide   0.3666563  0.480243340  0.0002125351
## total.sulfur.dioxide -0.3206955 -0.713663486  0.0626848131
## density               0.1128800 -0.003908289  0.7151620723
## pH                    0.1278367 -0.141310977 -0.2063605036
## sulphates            -0.2077642  0.045959499 -0.0772024671
## alcohol               0.2518903 -0.205053085  0.3357018784
```

The alphas for both PC1 and PC2 are plotted against each other to observe any obvious clusters/differentiators



Similar to the 'Congressmen' dataset that we discussed in class, we see that PC1 is an excellent classifier/discriminator to identify Red versus White wine

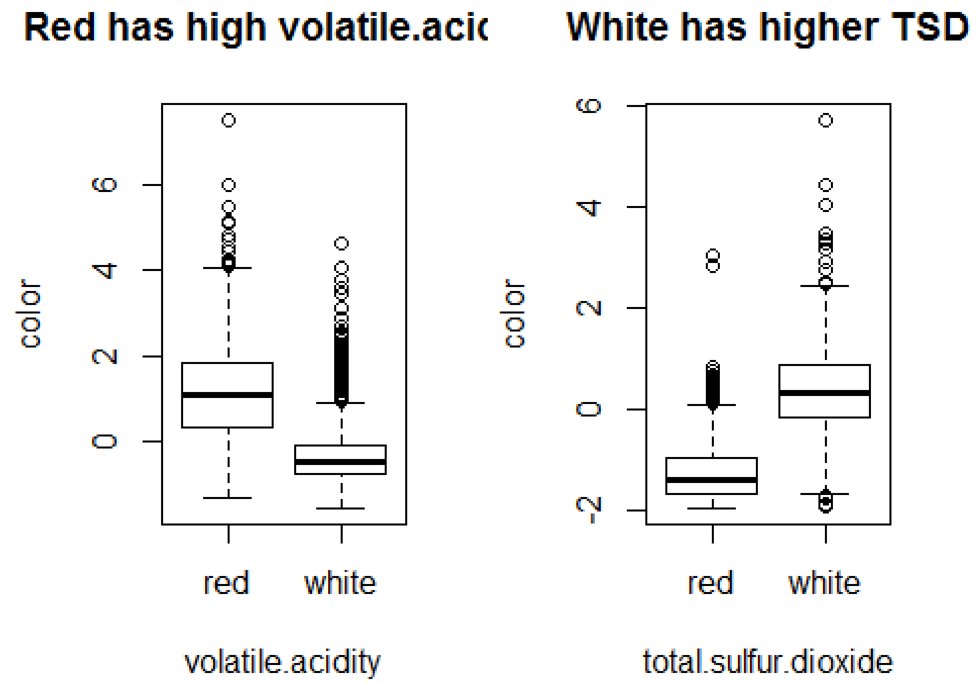
To check the validity of features that go into PC1, we can check if the varibales with the highest and lowest loadings in PC1, are 'dominant' variables to determine if a wine is red or white (from the data)

First, we find the variables with the highest and lowest loadings as below:

```
o1 = order(loadings[,1])
colnames(Z)[head(o1,3)]

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

colnames(Z)[tail(o1,3)]

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

From above we can conclude that: * "volatile.acidity", "sulphates", and "chlorides" are the most important variables for one group (red wine) * "residual.sugar", "free.sulfur.dioxide", and "total.sulfur.dioxide" are the most important variables for another group (white wine)

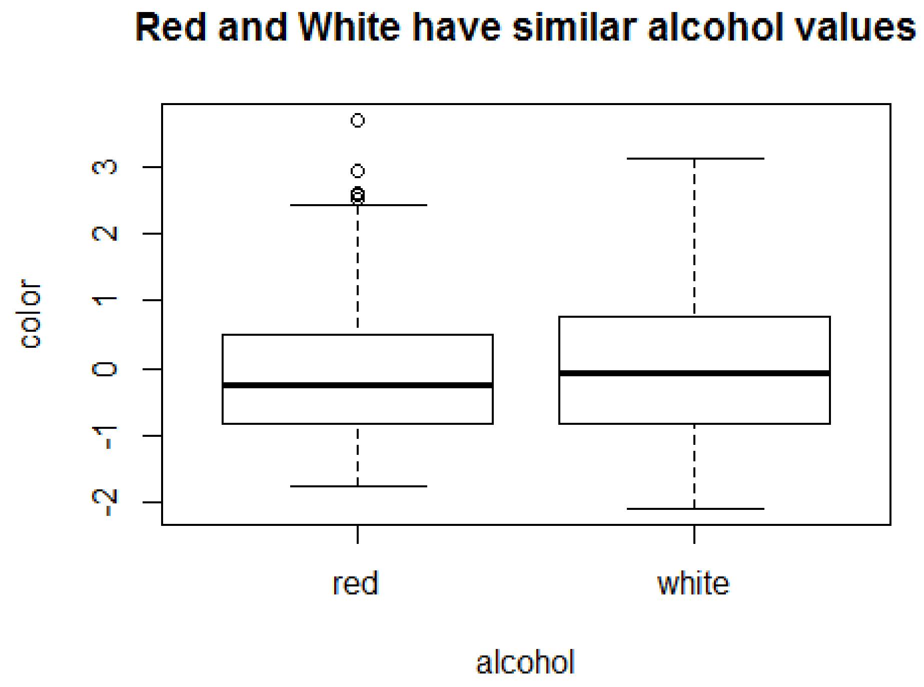
We can test if some of these variables indeed discriminate well between red and white wine using basic boxplots as below:



The 'loss' that we incur from using PC1, by way of dropping/ignoring some variables can also be validated using the same view. We can pick up the most important varibales from the second principal component, and check if they have any discriminatory power on the wine color

```
o2 = order(loadings[,2])
colnames(Z)[head(o2,3)]

## [1] "alcohol"             "pH"                  "free.sulfur.dioxide"

colnames(Z)[tail(o2,3)]

## [1] "residual.sugar" "fixed.acidity"  "density"
```

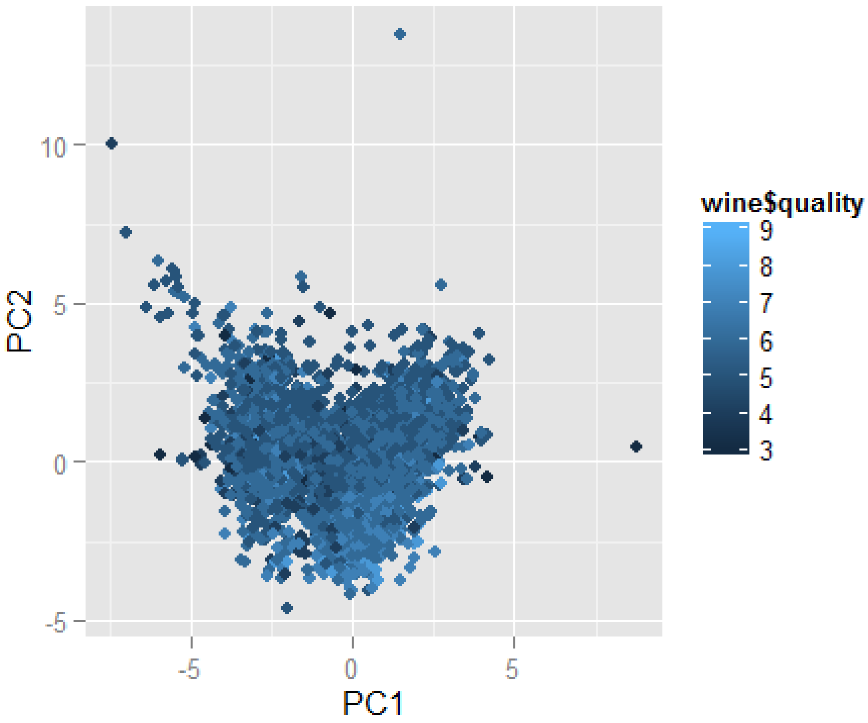
'alcohol' is a feature that is important in PC2, and is not a dominant part of PC1. We can test the same in the boxplot below:



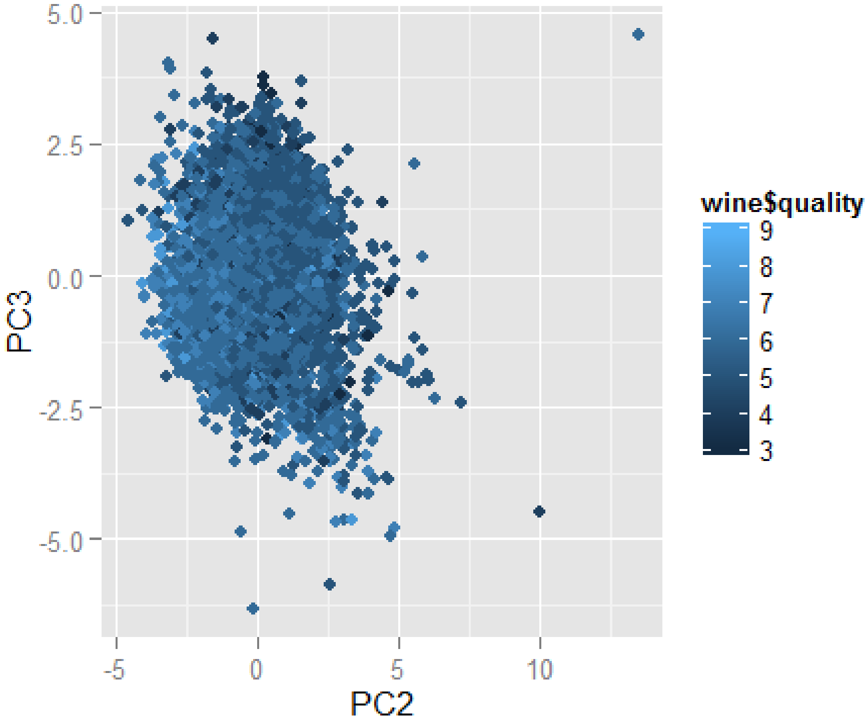
Can PCA tell us about the quality of wine as well?

We can plot the alphas for PC1 versus PC2 again and color code it with the qine quality data to see if PC1 or PC2 are able to distinguish the data points to the 7-point quality scale

```
par(mfrow = c(1,2))
qplot(scores[,1], scores[,2],color=wine$quality, xlab='PC1', ylab='PC2')
```



```
qplot(scores[,2], scores[,3],color=wine$quality, xlab='PC2', ylab='PC3')
```



As can be seen above, PC1 versus PC2, and even PC2 versus PC3 are not able to identify the various quality scales properly.

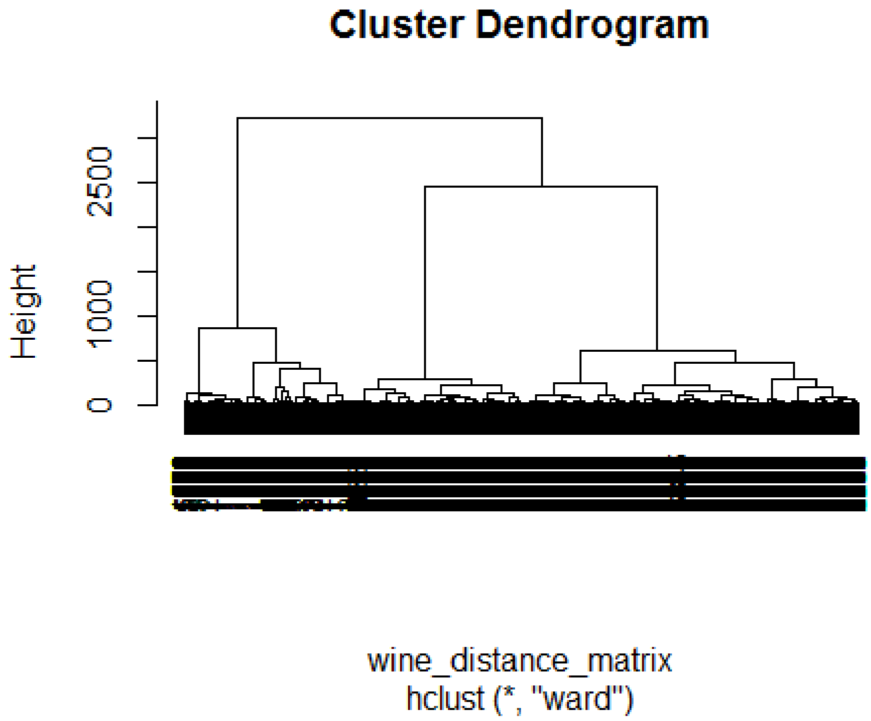
This indicates that a different clustering methid might be useful to identify the quality scales

Hierarchical clustering - Does it distinguish Red and White wines?

We first scale the data from the wine dataset

The pairwise distance matrix for the scaled dataset is calculated and input into the hclust fucntion to compute the dendrogram *NOTE : if the 'ward' method doesn't work, please use 'ward.D'. It's a R version issue*

```
par(mfrow =c(1,1))
wine_distance_matrix = dist(wine_scaled, method='euclidean')
wine_dend = hclust(wine_distance_matrix, method='ward')
plot(wine_dend, cex=0.8)
```



Reviewing the dendrogram, we can say that k=4 is a reasonable height to cut the dendrogram at

```
cluster1 = cutree(wine_dend, k=4)
summary(factor(cluster1))

##    1    2    3    4 
##  580 1106 1598 3213
```

We can check the homogenity of the clusters by looking at the proportions of each color wine in each of the clusters as below:

```
"Cluster 1"

## [1] "Cluster 1"

table(wine[which(cluster1 == 1),13])

##
##   red white
##   572     8

"Cluster 2"

## [1] "Cluster 2"

table(wine[which(cluster1 == 2),13])

##
##   red white
##   982   124

"Cluster 3"

## [1] "Cluster 3"

table(wine[which(cluster1 == 3),13])

##
##   red white
##     8  1590

"Cluster 4"

## [1] "Cluster 4"

table(wine[which(cluster1 == 4),13])

##
##   red white
##    37  3176
```

We notice that this Hierarchical Cluster is a good discriminator of Red and White wine

Can Hierarchical Clustering tell us about the quality of wine as well?

Similar to the exercise above, we can look at the distribution of wine quality scales across the different clusters to see if we find any obvious patterns

```
"Cluster 1"

## [1] "Cluster 1"

table(wine[which(cluster1 == 1),12])
```

```
## 
##   3   4   5   6   7   8 
##   6  37 264 235  35   3

"Cluster 2"

## [1] "Cluster 2"

table(wine[which(cluster1 == 2),12])

## 
##   3   4   5   6   7   8 
##   6  30 458 438 158  16

"Cluster 3"

## [1] "Cluster 3"

table(wine[which(cluster1 == 3),12])

## 
##   3   4   5   6   7   8   9 
##   3  29 709 712 123  21   1

"Cluster 4"

## [1] "Cluster 4"

table(wine[which(cluster1 == 4),12])

## 
##    3    4    5    6    7    8    9 
##   15  120  707 1451  763  153    4
```

We see that there are no obvious proportions and trends that stand out in the clusters that indicate identification of wine quality

In conclusion, we can say that PCA helped us in identifying Red versus White wine using just the first principal component. Hierachical Clustering also gave promising results by way of accurately identifying the White and Red wines. Both PCA and Hierarchical Clustering failed to identify the wine quality scales though

Question 4 - Market segmentation

Data Preparation for before analysis

We first red in the data from the input CSV

We will now convert the tweet counts to proportions across tweet categories to normalize the data. This helps to negate any effects of people who tweet often (high tweet counts) across one or many topics

```
Z = twit/rowSums(twit)
```

We now look to treat the adult and spam variables to identify any bots that might have creeped into the data despite the filters mentioned in the question. As a general rule, I will supress the records that have greater than average representation of adult and spam related tweets i.e., accounts that tweet on adult and spam themes more than the sample average number of times, will be excluded from further analysis

```
avg_adult = mean(Z[,'adult'])
avg_spam = mean(Z[,'spam'])
Z_New = Z[which(Z$adult < avg_adult & Z$spam < avg_spam),]
Z_New$spam <- NULL
Z_New$adult <- NULL
```

This may seem harsh for genuine users who tweet adult stuff occassinally, but they cannot be easily distinguished from, say, 'new' spam/adult bots i.e., bots that haven't tweeted

How can we make user clusters - K-means clustering

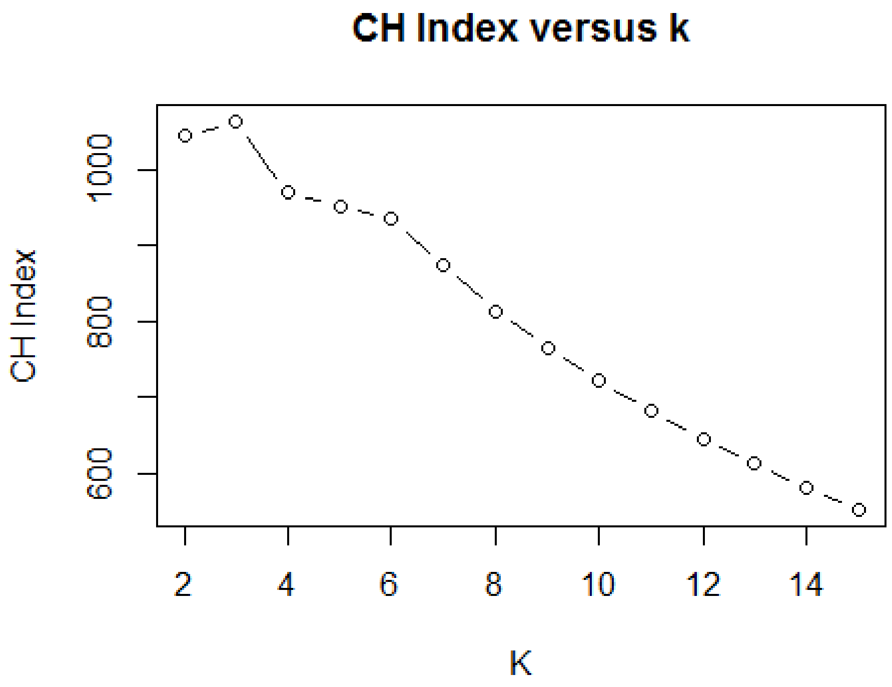
As the objective of this exercise is to achieve targeted marketing-viable segments of customers, we should ideally look for clustering methods to identify disparate segments first and then characterise them using latent factor methods such as PCA

As we do not have a particular number of segments in mind, we can use the CH Index method below to set the benchmark

```
kmax= 15

n = nrow(Z_New)
ch = numeric(length=kmax-1)

for (k in (2:kmax))
{
  km =kmeans (Z_New, k, nstart =50)
  with = km$tot.withinss
  betw = km$betweenss
  ch[k-1] = (betw/(k-1))/(with/(n-k))

}
```

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

## Warning: did not converge in 10 iterations

## Warning: did not converge in 10 iterations

## Warning: did not converge in 10 iterations

## Warning: did not converge in 10 iterations

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

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

```
plot(2:kmax,ch, xlab='K', ylab='CH Index', type='b',main='CH Index versus k'
)
```



The plot shows that k=6 should be ideal for this particular case, as the CH Index decreases after that drastically (3 clusters will be too small for a dataset of this magnitude)

Next, we run k-means clustering on the cleaned data to obtain the 6 clusters. We then add the cluster labels generated to our input cleaned data for further analysis

```
cluster_all <- kmeans(Z_New, centers=6, nstart=25)
Z_New$cluster <- cluster_all$cluster
```

- To characterise these clusters we can:
- Pick up the variables that have the highest means in each of the clusters
- Perform a PCA on each of the clusters to glean any latent factors that exist in them

Principal Components of each cluster using PCA

- After we've obtained the 6 clusters, we can analyse the 'most important' factors to characterise the cluster. This can be done in two ways:
- Pick up the variables with the highest column means to select the most well-represented variables in that cluster

- Perform a PCA on the cluster to find the variables with the highest loadings

The two methods above are related and will likely give the same result, hence we'll use PCA to gain additional insights

For each cluster, we observe the variables with the highest absolute loadings

Cluster 1

```
cluster1 = subset(Z_New, cluster == 1)

pc1 = prcomp(cluster1)

summary(pc1)

## Importance of components:
##                            PC1     PC2     PC3     PC4     PC5     PC6
## Standard deviation     0.09664 0.06173 0.05395 0.05142 0.04991 0.04815
## Proportion of Variance 0.19465 0.07942 0.06067 0.05511 0.05191 0.04833
## Cumulative Proportion  0.19465 0.27407 0.33475 0.38986 0.44178 0.49010
##                            PC7     PC8     PC9    PC10    PC11    PC12
## Standard deviation     0.04686 0.04319 0.03923 0.03825 0.03741 0.03628
## Proportion of Variance 0.04578 0.03889 0.03208 0.03050 0.02917 0.02744
## Cumulative Proportion  0.53588 0.57477 0.60684 0.63734 0.66651 0.69396
##                           PC13    PC14    PC15    PC16    PC17    PC18
## Standard deviation     0.03493 0.03436 0.03302 0.03197 0.03109 0.02997
## Proportion of Variance 0.02544 0.02461 0.02273 0.02131 0.02015 0.01872
## Cumulative Proportion  0.71939 0.74400 0.76673 0.78803 0.80818 0.82691
##                           PC19    PC20    PC21    PC22    PC23    PC24
## Standard deviation     0.02853 0.02788 0.02648 0.02538 0.02442 0.02438
## Proportion of Variance 0.01697 0.01620 0.01462 0.01343 0.01243 0.01239
## Cumulative Proportion  0.84387 0.86007 0.87469 0.88811 0.90055 0.91293
##                           PC25    PC26    PC27   PC28    PC29    PC30
## Standard deviation     0.02404 0.02353 0.02266 0.0219 0.02091 0.02079
## Proportion of Variance 0.01205 0.01154 0.01070 0.0100 0.00911 0.00901
## Cumulative Proportion  0.92498 0.93652 0.94722 0.9572 0.96633 0.97534
##                           PC31    PC32    PC33     PC34      PC35
## Standard deviation     0.02051 0.02009 0.01895 4.28e-17 5.798e-18
## Proportion of Variance 0.00876 0.00841 0.00748 0.00e+00 0.000e+00
## Cumulative Proportion  0.98410 0.99252 1.00000 1.00e+00 1.000e+00

loadings1 = pc1$rotation
o1 = order(loadings1[,1])

colnames(cluster1)[head(o1,5)]

## [1] "tv_film"        "art"              "current_events" "college_uni"
## [5] "travel"

colnames(cluster1)[tail(o1,5)]
```

```
## [1] "family"        "food"          "parenting"     "religion"
## [5] "sports_fandom"

#The key attributes for this cluster are :
#"chatter" "current_events" "travel" "photo_sharing" "shopping" (Ignoring
#"uncategorized")
```

This cluster is opinionated and is out-going. It is also seems very receptive and hence present a low-hanging fruit opportunity for social/physical engagement, say, outside the domain on Twitter -- malls, outdoor events etc.

"chatter" and "photo_sharing" are attributes that are shared by almost all clusters by virtue of how peopel generally tweet

**Hence, we can treat "chatter" and "photo_sharing" as engagement metrics instead of classification metrics i.e., we can use it to classify a cluster as "being receptive" or "being passive/blind to marketing"

Cluster 2

```
cluster2 = subset(Z_New, cluster == 2)

pc2= prcomp(cluster2)

loadings2 = pc2$rotation
o2 = order(loadings2[,1])

colnames(cluster2)[head(o2,5)]

## [1] "travel"      "politics"    "computers"   "college_uni" "food"

colnames(cluster2)[tail(o2,5)]

## [1] "family"        "chatter"       "sports_fandom" "automotive"
## [5] "news"

#The key attributes for this cluster are :
#"news", "automotive", "travel", "politics"
```

This cluster represents the user that consumes and has opinions on news items across the range of topics. This user base should ideally be receptive to associations with news sites/longforms/op-eds

Cluster 3

```
cluster3 = subset(Z_New, cluster == 3)

pc3= prcomp(cluster3)

loadings3 = pc3$rotation
o3 = order(loadings3[,1])
```

```
colnames(cluster3)[head(o3,5)]

## [1] "health_nutrition" "personal_fitness" "cooking"
## [4] "outdoors"         "food"

colnames(cluster3)[tail(o3,5)]

## [1] "dating"        "sports_fandom" "shopping"      "photo_sharing"
## [5] "chatter"

#The key attributes for this cluster are :
#"sports_fandom", "religion", "parenting", "tv_film"
```

This cluster represents the head of the family who also values nutrition (as a product of parenting). Our client should ideally involve this segment in community-related online activities and focus on family well-being through nutrition in its marketing materials

Cluster 4

```
cluster4 = subset(Z_New, cluster == 4)

pc4= prcomp(cluster4)

loadings4 = pc4$rotation
o4 = order(loadings4[,1])

colnames(cluster4)[head(o4,5)]

## [1] "tv_film"       "chatter"       "music"         "religion"
## [5] "uncategorized"

colnames(cluster4)[tail(o4,5)]

## [1] "parenting"      "art"            "sports_playing" "college_uni"
## [5] "online_gaming"

#The key attributes for this cluster are :
#"online_gaming", "fcollege_uni", "sports_playing", "tv_film", "music"
```

This cluster represents the student population that is heavily immersed in popular culture. This insight can be used to flavour the marketing materials with a focus on college and should ideally respond to campaigns that reference popular media

Cluster 5

```
cluster5 = subset(Z_New, cluster == 5)

pc5= prcomp(cluster5)

loadings5 = pc5$rotation
```

```
o5 = order(loadings5[,1])

colnames(cluster5)[head(o5,5)]

## [1] "chatter"        "travel"         "shopping"       "current_events"
## [5] "tv_film"

colnames(cluster5)[tail(o5,5)]

## [1] "music"         "uncategorized" "fashion"       "beauty"
## [5] "cooking"

#The key attributes for this cluster are :
#"beauty", "cooking", "fashion", "travel"
```

This cluster represents the presumably female segment of our client's twitter followers and hence the marketing messaging can be targeted towards female health issues and can be dominated by any offerings

Cluster 6

```
cluster6 = subset(Z_New, cluster == 6)

pc6= prcomp(cluster6)

loadings6 = pc6$rotation
o6 = order(loadings6[,1])

colnames(cluster6)[head(o6,5)]

## [1] "photo_sharing" "shopping"      "automotive"    "politics"
## [5] "eco"

colnames(cluster6)[tail(o6,5)]

## [1] "dating"         "travel"         "uncategorized"  "current_events"
## [5] "chatter"

#The key attributes for this cluster are :
#"health_nutrition", "personal_fitness", "shopping", "food", "cooking"
```

This cluster represents the segment that is serious about health and nutrition and also shops online (probably for healthy cooking related items). The verbage for any marketing for this cluster can include more technical specifications than other cluster