PSet 3

Yavuz Ramiz Çolak, Ryo Tamaki, David Lieberman, Liana Wang

2/22/2019

Introduction to Our Data set

We're using a dataset on the New York Stock Exchange, which we obtained from Kaggle. There are 79 columns of variables measuring various aspects of stock performance of the S&P 500, with data from the past few years. In every row, we have a company, and each row name is set to the company's stock identifying ticker. Column names include various characteristics about the company ranging from Earnings and Total Costs, to Profit Marging and Price/Earnings ratio. For our cluster analysis, we formed a subset of this data set. For this, we selected 6 columns easily comparable across companies: Cash Ratio, Gross Margin, Operating Margin, Profit Margin, Quick Ratio, and Earnings Per Share.

These 6 ratio measures tell more about the profitability / cost management of the company through Profit Margin, Operating Margin, Gross Margin and Earnings Per Share. Furthermore, they tell about the leverage risk of the company through Quick Ratio and Cash Ratio–liquidity ratios that tell more about how easily a company can service its debt.

When we look at the selected 45 rows, each of which represents a publicly-traded company, we see that they come from a range of industries. They represent approximately 7 different industries. It is known that each industry has its own unique average profit margins and liquidity ratios. For example, AMGN (Amgen) is a biotech firm that has much higher debt levels, which is normal for the biotech industry. While AMZN (Amazon) is a tech/consumers company that has higher profit margins and relatively less debt burden, which is also normal for a consumers company. Given that we have 7 industries in this subsetted data set, and each industry has a unique character, our hypothesis is that we should have between 6-8 clusters. We will test whether this is true through hierarchical and k-means cluster analysis.

```
# Reading the Data
new_data <- read.csv("https://pastebin.com/raw/bNrAceXf")
```

```
# Selecting Relevant Columns
index3 <- c(5, 9, 19, 20, 21, 26)

# Selecting 45 Rows
rowindex <- seq(1, 89, by = 2)

data_1 <- new_data[rowindex,]

# Changing the Row Names to Tickers of the Company
data_1 <- data_1 %>% remove_rownames %>% column_to_rownames(var="Ticker.Symbol")

# Subsetting by 6 Columns
data_2 <- data_1[,index3]
```

PROBLEM 1

1. Think about what metrics are appropriate for your data based on data type. Write a few sentences about this. Also think about whether you should standardize or transform your data (comment as appropriate).

```
# Scaling the data
data_f <- scale(data_2)
```

Before we started our analysis, we standardized our data. For this we used the scale function in R. A standardized dataset is necessary mainly because we are calculating distance in the cluster analysis. Non-scaled columns could distort the distance calculations and offer a final clustering that does not have much explanatory power over our data.

The columns of our subsetted data set include only continuous variables. Therefore, for this data set, Euclidean or Manhattan distance metrics will work well. Below in our analysis, we observe that Euclidean distance offers more descriptive clusters, and thus for most of this analysis, we preferred Euclidean distance.

Furthermore, because each company is unique and our data set includes all sorts of companies, there are likely to be outliers. Because simple linkage is more sensitive to outliers, agglemeration methods that are less sensitive to outliers would work better for our data set. Complete linkage is an example method that is suitable.

PROBLEM 2

2. Try various forms of hierarchical cluster analysis. Try at least two different metrics and two agglomeration methods. Produce dendrograms and comment on what you observe.

Model 1

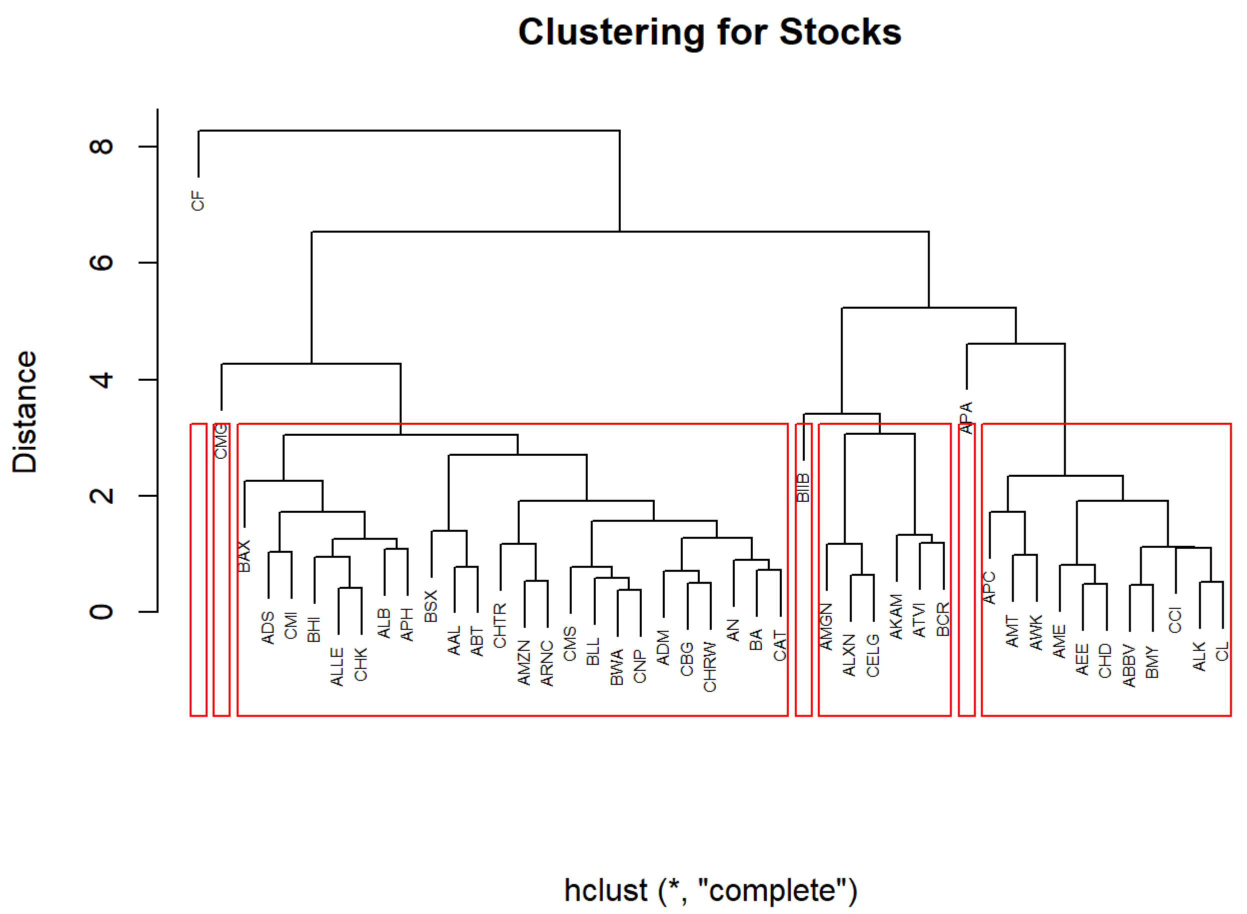
Euclidean Distance + Complete

```
# Get distance matrix
data_dist <- dist(data_f[,-1], method="euclidean")

# Perform cluster analysis
data_clust <- hclust(data_dist, method = "complete")

# Make dendrogram
plot(data_clust, xlab="",ylab="Distance", cex = .5, main="Clustering for Stocks")

# Identify groups
#rect.hclust(data_clust, k=5)
rect.hclust(data_clust,k=7)
```



In this first model distance was calculated using the Euclidean method and a complete agglomeration method. The last command asks R to identify 7 clusters, since our data contains roughly 7 industries, and we think that might influence variables like risk and profit margin. From the dendogram, though, we can see three relatively larger clusters and four small single-stock clusters (CF, CMG, BIIB, APA). The left-most large cluster on the tree contains a variety of stocks, including those from companies in communications and manufacturing. The second cluster is smaller, but more specific to medical and other technology. The last cluster has a large number of stocks related to utilities and household products.

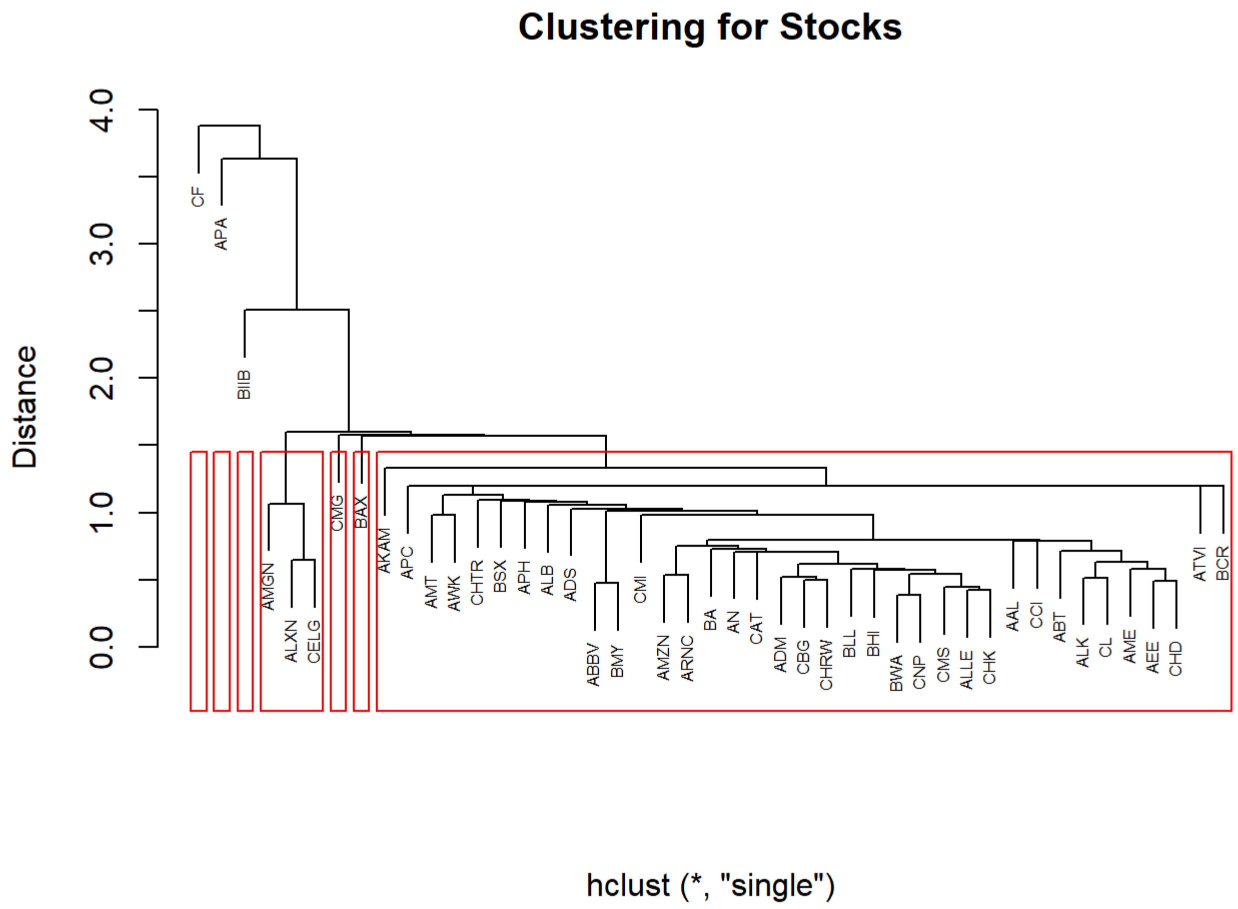
Model 2

```
# Get distance matrix
data_dist_2 <- dist(data_f[,-1], method = "euclidean")

# Perform cluster analysis
data_clust_2 <- hclust(data_dist_2, method = "single")

# Make dendrogram
plot(data_clust_2, xlab="",ylab="Distance",main="Clustering for Stocks", cex = .5)

# Identify groups
rect.hclust(data_clust_2, k=7)
```



Model 2 used a “single” agglomeration method, which ought to produce more distinct groups. In this case, however, we obtained one large cluster that includes most of the stocks, and a few scattered ones toward the left-hand side of the dendogram. After seeing the results of single agglomeration, it does not appear to be particularly useful for cluster analysis, so we decided to proceed without using this agglomeration method again.

Model 3

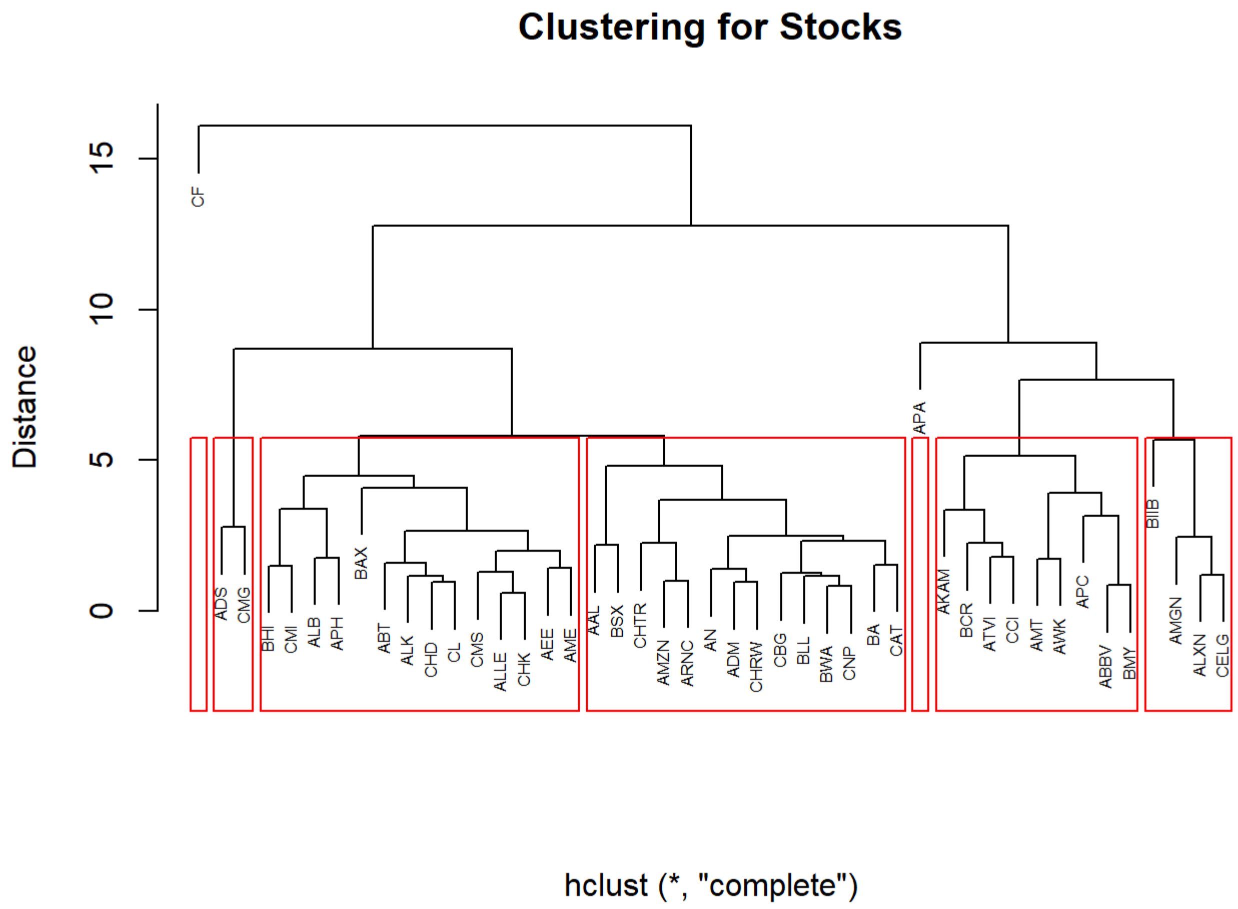
Manhattan + Complete

```
# Get distance matrix
data_dist_3 <- dist(data_f[,-1], method = "manhattan")

# Perform cluster analysis
data_clust_3 <- hclust(data_dist_3, method = "complete")

# Make dendrogram
plot(data_clust_3, xlab="",ylab="Distance",main="Clustering for Stocks", cex = .5)

# Identify groups
rect.hclust(data_clust_3,k=7)
```

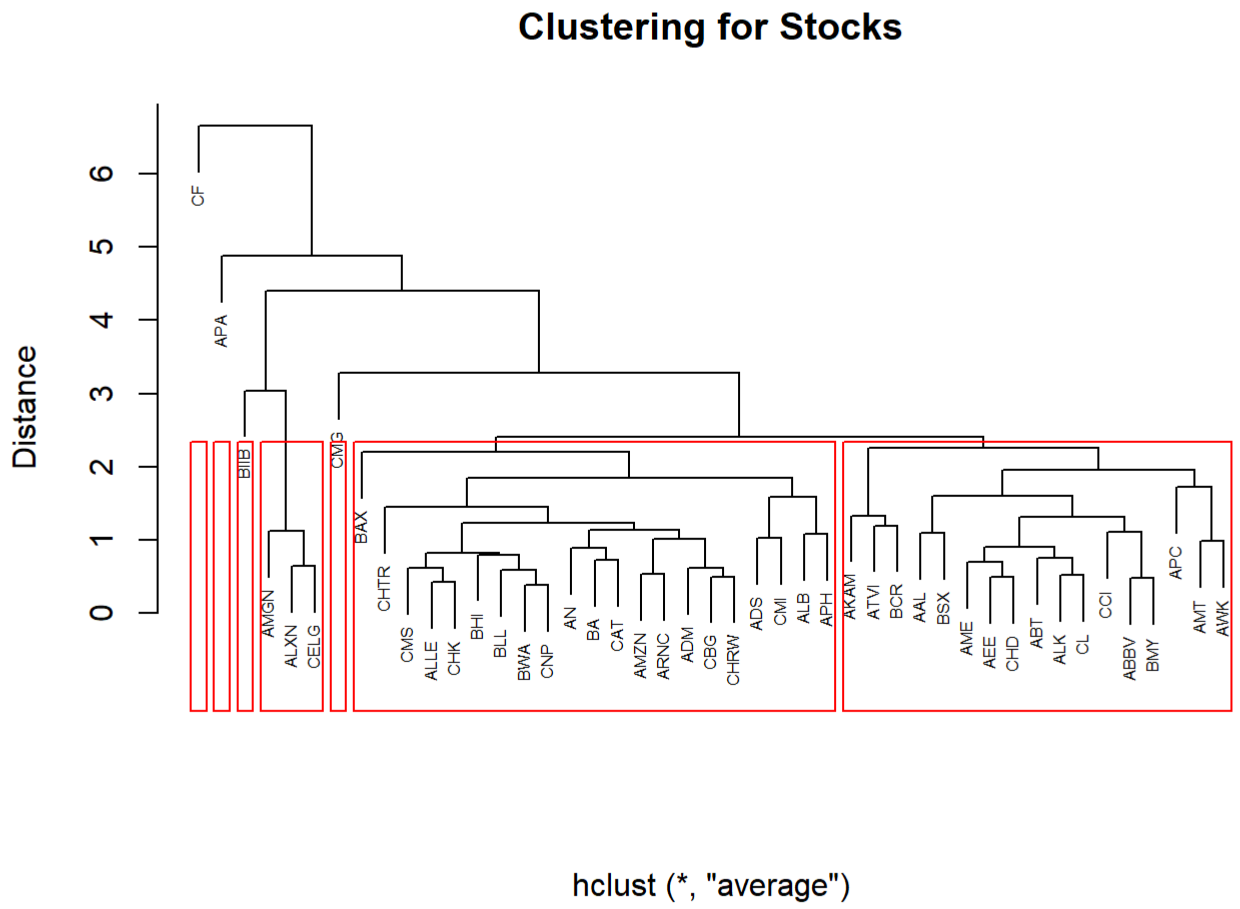


Since the Manhattan distance calculation could work well for high-dimensional datasets, and also places less weight upon outliers, we also decided to see whether a different distance calculation would produce different clusters from the Euclidean distance method. In fact, this method obtains four groups with more stocks in them. While these groups do not directly correspond with different industries, the right-most cluster is composed of large pharmaceutical stocks, while another cluster is composed of mainly tech, media, and communications related stocks. The mix of stocks that do not align with industry might have something to do with market share or company size–we can see that several large companies, like Amazon, CenterPoint Energy and Boeing are all located in one cluster.

Model 4

Manhattan + Average

```
# Get distance matrix
data_dist_6 <- dist(data_f[,-1], method = "euclidean")
# Perform cluster analysis
data_clust_6 <- hclust(data_dist_6, method = "average")
# Make dendrogram
plot(data_clust_6, xlab="",ylab="Distance", main="Clustering for Stocks", cex = .5)
# Identify three groups
rect.hclust(data_clust_6, k=7)
```



Since the Manhattan distance method seemed to produce more distinct clusters in Model 3, we decided to try this technique with “average” agglomeration, which ought to average the effect of both single and complete agglomeration. However, the clusters that are obtained from this method appear to be less clearly useful than in models 2 and 3. Four of the seven “clusters” are single stocks, while the fifth contains only three stocks, with the remaining two groups containing a mix of stocks not united by any common characteristics. This indicates high variation within groups and not between them.

PROBLEM 3

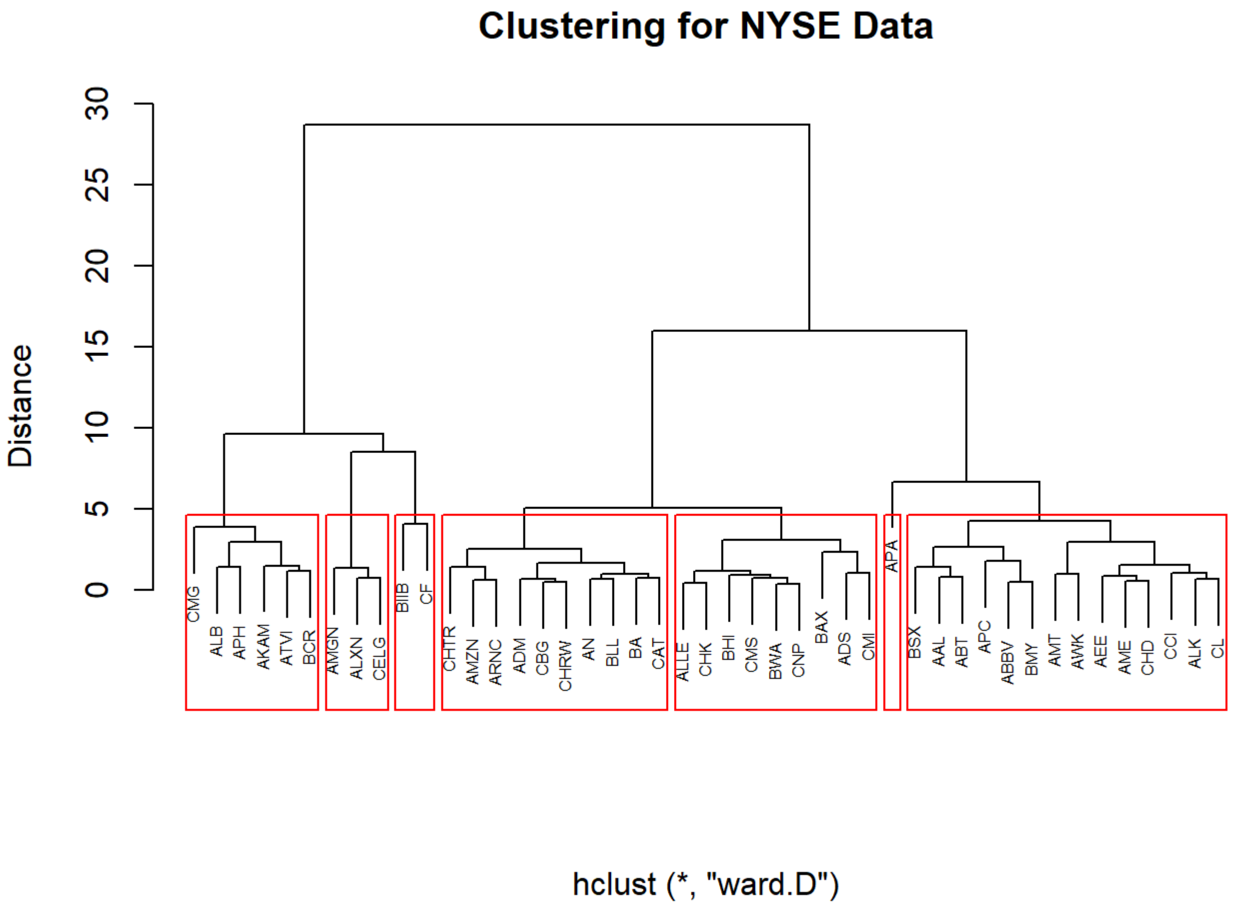
3. If possible, run the SAS macro or the R function to think about how many groups you want to retain (i.e. get the plots of cluster distance, R-squared, etc). If you can’t run this, discuss how many groups you think are present.

We utilise the Euclidian distance method to compute distances and intiate clustering. Euclidean distance is particularly meaningful as our subsetted data set only includes continuous variables. We utilise Ward’s method for hierarchical clustering for two reasons. First, there may be a few outliers, and simple linkage is sensitive to outliers. Second, as explained in the lecture “several studies of clustering suggest that Ward’s method or Average Linkage work the ‘best’” (pg. 324, Cluster Analysis Lecture). We opt for 7 clusters along industry lines to begin when we plot the dendogram as we hypothesise that the stocks could— as mentioned previously— potentially be divided as such. Our observations of the stocks suggest that there are 7 relevant industries. We included these industries and stock breakdown in the end of this analysis, under Question 5. When we perform distance cluster analysis with 7 clusters, the result appears to match our hypothesis–that the clustering of companies supports a similar narrative.

```
# Using the Euclidean distance method to compute distances to begin the clustering process
dist1 <- dist(data_f, method="euclidean")

#We run the hclust() command to create the Hierarchical Ward Cluster (using Euclidean distance).
clust1 <- hclust(dist1, method="ward.D")

# We plot the dendogram of our data with 7 clusters.
plot(clust1, cex=.9, xlab="",ylab="Distance",main="Clustering for NYSE Data", cex = .5)
rect.hclust(clust1,k=7)
```



To build on our hypothesis of retaining 7 groups across industry lines, we evaluate the number of groups that we wish to retain by calculating the ‘guidance statistics,’ namely: R-Squared, Root-mean-square standard deviation, semi-partial R-squared, and cluster distance. Again, we utilise Euclidean distance and the Ward method. In computing these statistics and producing the plot, we opt to retain 7 groups. Starting with the RSQ, we see that it reaches a plateau when we choose 7 clusters. After 7, adding additional clusters is not particularly beneficial— it is marginal— as the difference in the internal sum of squares is essentially unchanged. In RMSSTD, we observe a local minimum at the 7 cluster mark as well. Local minimum can often be seen interpreted as a sign of joining data points that are similar. These can be seen in the plot below.

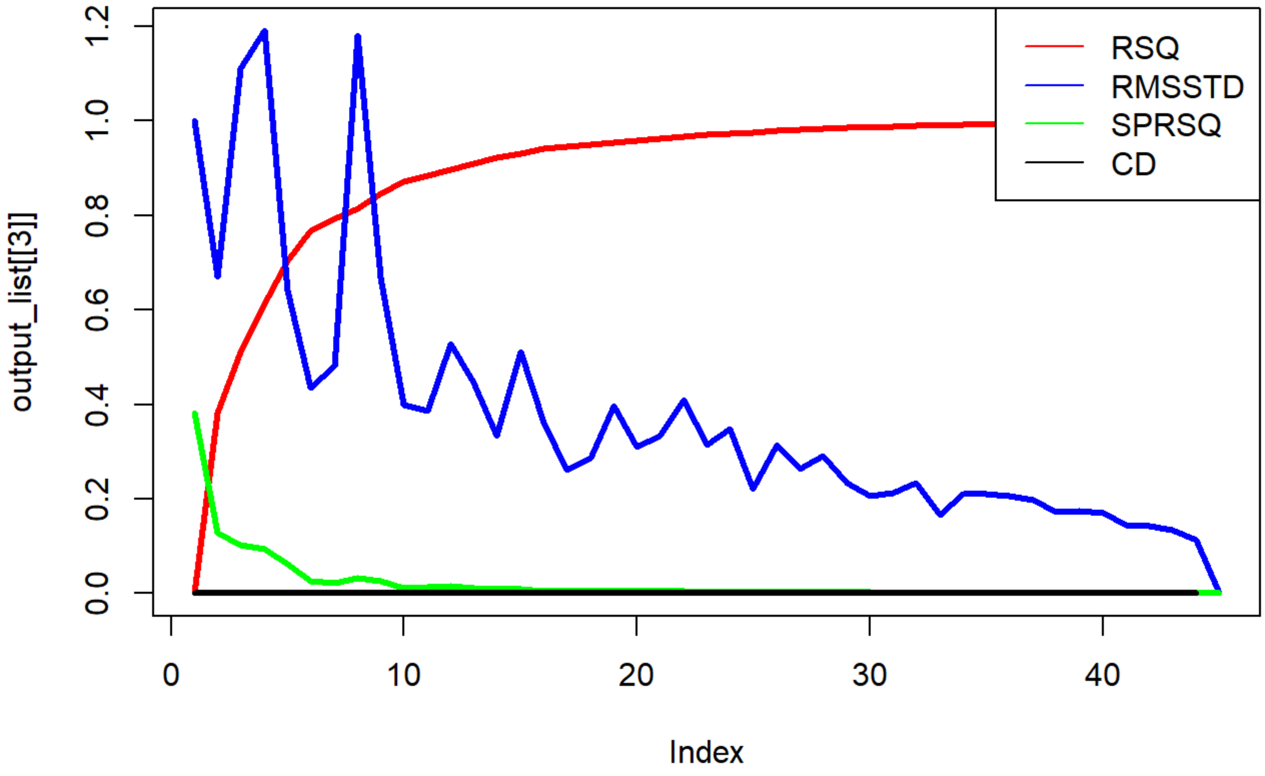
```
# Loading the hclus_eval code
source("http://reuningscherer.net/stat660/R/HClusEval.R.txt")

# Running hclus_eval to produce the aforementioned guidance statistics; we use the Euclidean method for distances and the Ward method for clustering once again.
hclus_eval(data_f, dist_m = 'euclidean', clus_m = 'ward', plot_op = T)
```

```
## [1] "Creating Distance Matrix using euclidean"
## [1] "Clustering using ward"
```

```
## The "ward" method has been renamed to "ward.D"; note new "ward.D2"
```

```
## [1] "Clustering Complete. Access the Cluster object in first element of output"
## [1] "Calculating RMSSTD"
## [1] "RMSSTD Done. Access in Element 2"
## [1] "Calculating RSQ"
## [1] "RSQ Done. Access in Element 3"
## [1] "Calculating SPRSQ"
## [1] "SPRSQ Done. Access in Element 4"
## [1] "Calculating Cluster Dist. "
## [1] "CD Done. Access in Element 5"
```



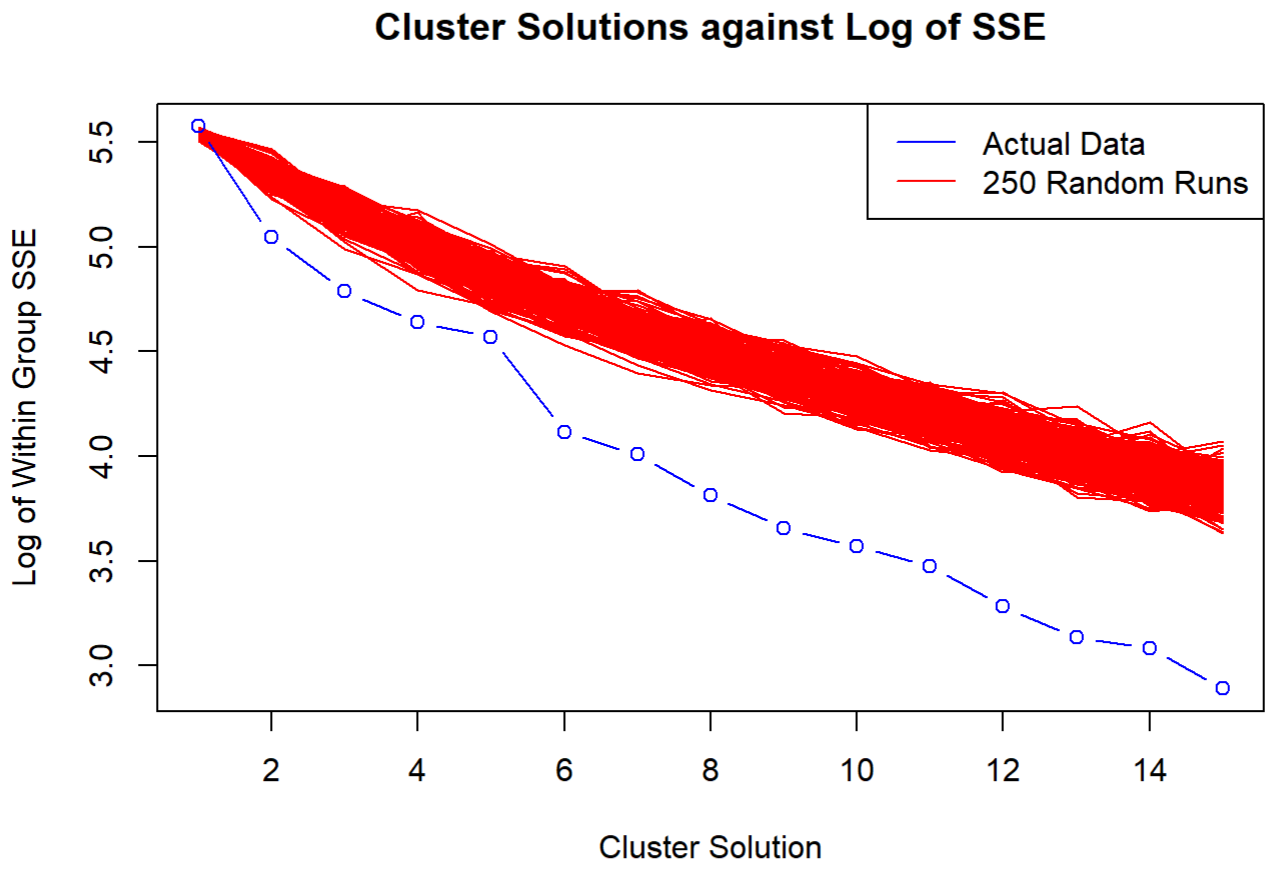
```
## [[1]]
##
## Call:
## hclust(d = dist1, method = clus_m)
##
## Cluster method   : ward.D
## Distance         : euclidean
## Number of objects: 45
##
##
## [[2]]
##  [1] 1.0000000 0.6694337 1.1122066 1.1909486 0.6400930 0.4354933 0.4817678
##  [8] 1.1807627 0.6687721 0.3991059 0.3857770 0.5290575 0.4457276 0.3332113
## [15] 0.5122562 0.3617457 0.2607688 0.2860754 0.3961159 0.3100152 0.3334677
## [22] 0.4094691 0.3132677 0.3483969 0.2211766 0.3142430 0.2634327 0.2911844
## [29] 0.2344589 0.2069490 0.2122070 0.2327383 0.1662922 0.2112981 0.2102045
## [36] 0.2061939 0.1979438 0.1714831 0.1750057 0.1689388 0.1452462 0.1433703
## [43] 0.1331432 0.1132957 0.0000000
##
## [[3]]
##  [1] 0.0000000 0.3827567 0.5109118 0.6122827 0.7050773 0.7668674 0.7927816
##  [8] 0.8151036 0.8467900 0.8721690 0.8836434 0.8960930 0.9105959 0.9221697
## [15] 0.9312389 0.9409222 0.9464032 0.9505913 0.9546899 0.9590634 0.9627360
## [22] 0.9665595 0.9703700 0.9738161 0.9765747 0.9788108 0.9810550 0.9833189
## [29] 0.9852460 0.9870235 0.9886866 0.9900849 0.9913159 0.9922812 0.9932959
## [36] 0.9943001 0.9952664 0.9961569 0.9970140 0.9977101 0.9983588 0.9988382
## [43] 0.9993054 0.9997083 1.0000000
##
## [[4]]
##  [1] 0.3827567197 0.1281550769 0.1013709358 0.0927945558 0.0617900818
##  [6] 0.0259142448 0.0223220333 0.0316863773 0.0253789341 0.0114744627
## [11] 0.0124496214 0.0145028809 0.0115737283 0.0090692467 0.0096832737
## [16] 0.0054810193 0.0041880974 0.0040986047 0.0043735286 0.0036725422
## [21] 0.0038235062 0.0038105674 0.0034460540 0.0027586457 0.0022360141
## [26] 0.0022442883 0.0022639055 0.0019270080 0.0017775231 0.0016631233
## [31] 0.0013982570 0.0012310713 0.0009652341 0.0010147022 0.0010042260
## [36] 0.0009662712 0.0008904940 0.0008571922 0.0006960680 0.0006486433
## [41] 0.0004794650 0.0004671600 0.0004028890 0.0002917253 0.0000000000
##
## [[5]]
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [36] 0 0 0 0 0 0 0 0 0
```

PROBLEM 4

4. Run k-means clustering on your data. Compare results to what you got in 3.) Include a sum of squares vs. k (number of clusters) plot and comment on how many groups exist.

```
set.seed(10)
```

```
# kdata is just normalized input dataset
kdata <- data_f
# Set max value for number of clusters k
n.lev <- 15
# Calculate the within groups sum of squared error (SSE)
#for the number of cluster solutions selected by the user
wss <- rnorm(10)
while (prod(wss==sort(wss,decreasing=T))==0) {
  wss <- (nrow(kdata)-1)*sum(apply(kdata,2,var))
  for (i in 2:n.lev) wss[i] <- sum(kmeans(kdata, centers=i)$withinss)}
# Calculate the within groups SSE for 250
#randomized data sets (based on the original input data)
k.rand <- function(x){
  km.rand <- matrix(sample(x),dim(x)[1],dim(x)[2])
  rand.wss <- as.matrix(dim(x)[1]-1)*sum(apply(km.rand,2,var))
  for (i in 2:n.lev) rand.wss[i] <- sum(kmeans(km.rand, centers=i)$withinss)
  rand.wss <- as.matrix(rand.wss)
  return(rand.wss)
}
rand.mat <- matrix(0,n.lev,250)
k.1 <- function(x) {
  for (i in 1:250) {
    r.mat <- as.matrix(suppressWarnings(k.rand(kdata)))
    rand.mat[,i] <- r.mat}
  return(rand.mat)
}
# Same function as above for data with < 3 column variables
k.2.rand <- function(x){
  rand.mat <- matrix(0,n.lev,250)
  km.rand <- matrix(sample(x),dim(x)[1],dim(x)[2])
  rand.wss <- as.matrix(dim(x)[1]-1)*sum(apply(km.rand,2,var))
  for (i in 2:n.lev) rand.wss[i] <- sum(kmeans(km.rand, centers=i)$withinss)
  rand.wss <- as.matrix(rand.wss)
  return(rand.wss)
}
k.2 <- function(x){
  for (i in 1:250) {
    r.1 <- k.2.rand(kdata)
    rand.mat[,i] <- r.1}
  return(rand.mat)
}
# Determine if the data data table has > or < 3
#variables and call appropriate function above
if (dim(kdata)[2] == 2) { rand.mat <- k.2(kdata) } else { rand.mat <- k.1(kdata) }
# Plot within groups SSE against all tested cluster solutions
#for actual and randomized data - 1st: Log scale, 2nd: Normal scale
xrange <- range(1:n.lev)
yrange <- range(log(rand.mat),log(wss))
plot(xrange,yrange, type='n', xlab='Cluster Solution', ylab='Log of Within Group SSE', main='Clu
ster Solutions against Log of SSE')
for (i in 1:250) lines(log(rand.mat[,i]),type='l',col='red')
lines(log(wss), type="b", col='blue')
legend('topright',c('Actual Data', '250 Random Runs'), col=c('blue', 'red'), lty=1)
```



We calculate the internal sum of squares distances through 250 random partitions with the same number of clusters, each having the same sample size as their respective k-means partition counterparts.

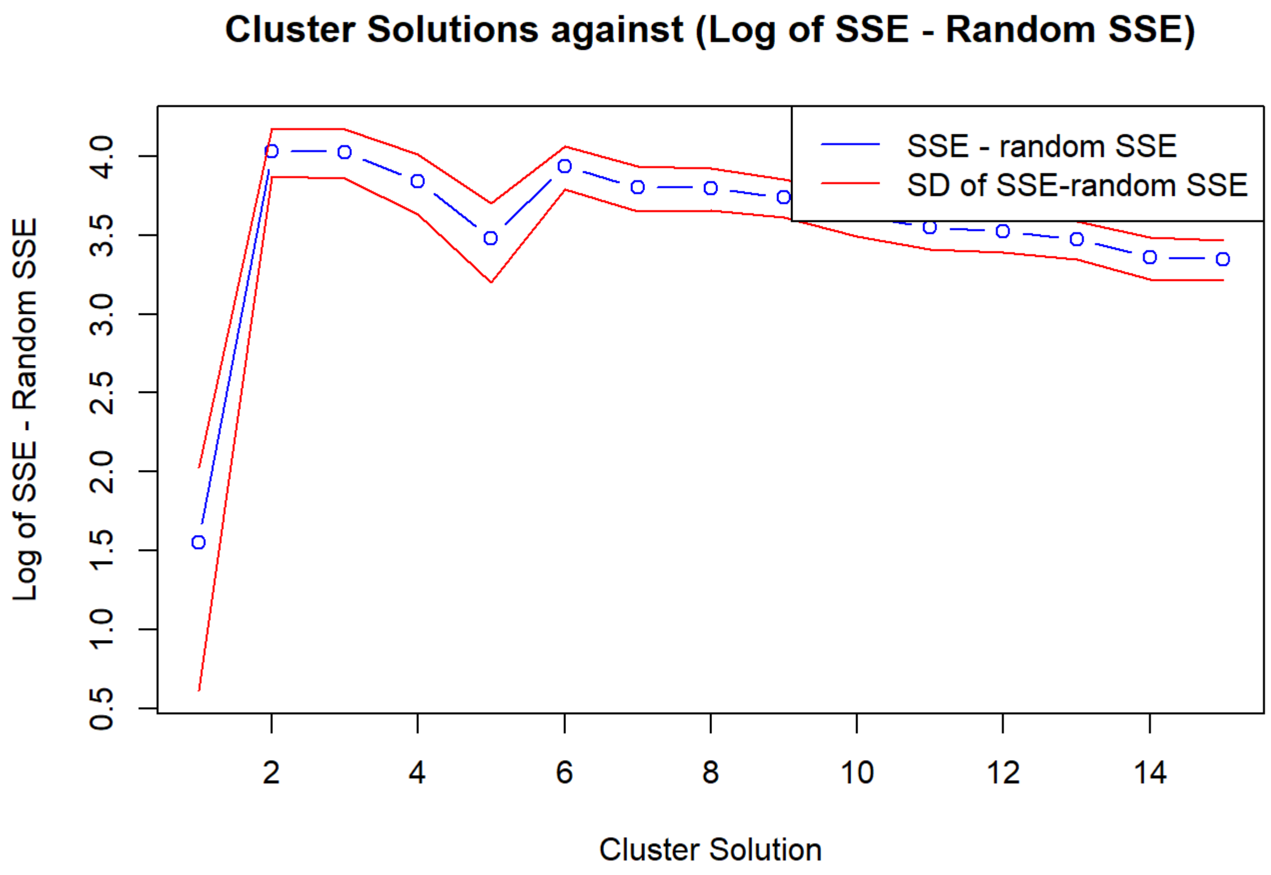
We look at the distance between our internal sum of squares blue line and red lines, and determine at what number of clusters that difference stops increasing (improving). That becomes our cluster number, and for our data, it is 6 clusters. We expected there to be 6 to 8 clusters, mainly through groups formed on an industry level.

In the next step, we will check the difference between SSE of the actual data and the SSE of 250 randomized datasets.

```
# Calculate the mean and standard deviation of
#difference between SSE of actual data and SSE
#of 250 randomized datasets
r.sse <- matrix(0,dim(rand.mat)[1],dim(rand.mat)[2])
wss.1 <- as.matrix(wss)
for (i in 1:dim(r.sse)[2]) {
  r.temp <- abs(rand.mat[,i]-wss.1[,1])
  r.sse[,i] <- r.temp}
r.sse.m <- apply(r.sse,1,mean)
r.sse.sd <- apply(r.sse,1,sd)
r.sse.plus <- r.sse.m + r.sse.sd
r.sse.min <- r.sse.m - r.sse.sd

# Plot difference between actual SSE mean SSE
#from 250 randomized datasets - 1st: Log scale, 2nd: Normal scale

xrange <- range(1:n.lev)
yrange <- range(log(r.sse.plus),log(r.sse.min))
plot(xrange,yrange, type='n',xlab='Cluster Solution', ylab='Log of SSE - Random SSE', main='Cluster Solutions against (Log of SSE - Random SSE)')
lines(log(r.sse.m), type="b", col='blue')
lines(log(r.sse.plus), type='l', col='red')
lines(log(r.sse.min), type='l', col='red')
legend('topright',c('SSE - random SSE', 'SD of SSE-random SSE'), col=c('blue', 'red'), lty=1)
```



This analysis focuses on the difference between the average of the red lines and the blue lines. We see that after 6 clusters, we do not observe much change in the difference between the averages, which is to say adding addition clusters beyond 6 did not really bring any further improvement. This matches the result we obtained in the previous section in our analysis that included 250 simulated data sets.

Therefore, in k-means clustering, we decided to keep 6 clusters, and below we provide the means of each cluster per column variables.

```
clust.level <- 6
# Apply K-means cluster solutions - append clusters to CSV file
fit <- kmeans(na.omit(kdata), clust.level)
aggregate(kdata, by=list(fit$cluster), FUN=mean)
```

```
##   Group.1 Cash.Ratio Gross.Margin Operating.Margin Profit.Margin
## 1       1  1.8722956   1.30080621        1.8623383    1.92665455
## 2       2  1.1855790   0.08214201        0.2342750    0.01779819
## 3       3 -0.6235780  -0.48477250       -1.1802717   -1.07679077
## 4       4 -0.6516892   1.38160162        0.2787576    2.78764221
## 5       5 -0.4910537  -0.89144276       -0.6346185   -0.46275306
## 6       6 -0.4107360   0.57053994        0.3403489   -0.01557342
##     Quick.Ratio Earnings.Per.Share
## 1  1.7375101471          1.2899931
## 2  1.2148649966          0.1548702
## 3 -0.8518076917         -0.5529465
## 4 -0.0002017545         -3.1607495
## 5 -0.3034388366          0.1317366
## 6 -0.5512244579         -0.2638272
```

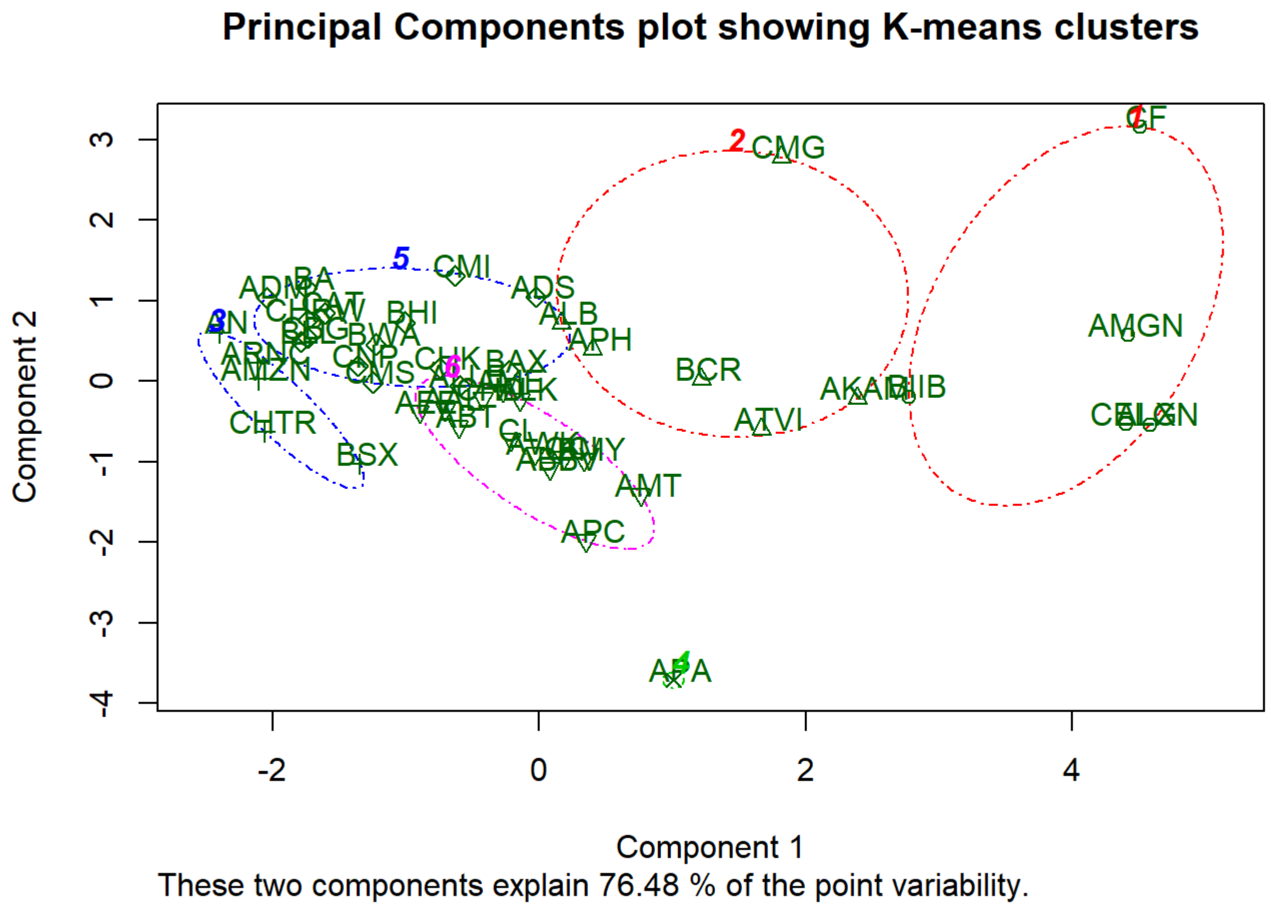
As hypothesised in the beginning, we see that each group has a unique characteristic in terms of profitability and liquidity/risk level. These are close to the average margins / cash ratios of said industries, as we will explain in Question 5.

Finally, it is possible to plot our k-means result in principles component space and discriminant analysis space.

In Principal Component space, we see that the two major principal components account for 76.48% of the variability. This is expected, particularly because different types of margins may show correlation. For example, profit margin and gross margin, while different measures, often tend to be high or low at the same time. This is why although we can cut to two principal components, we are still able to account for more than 75% of the variability.

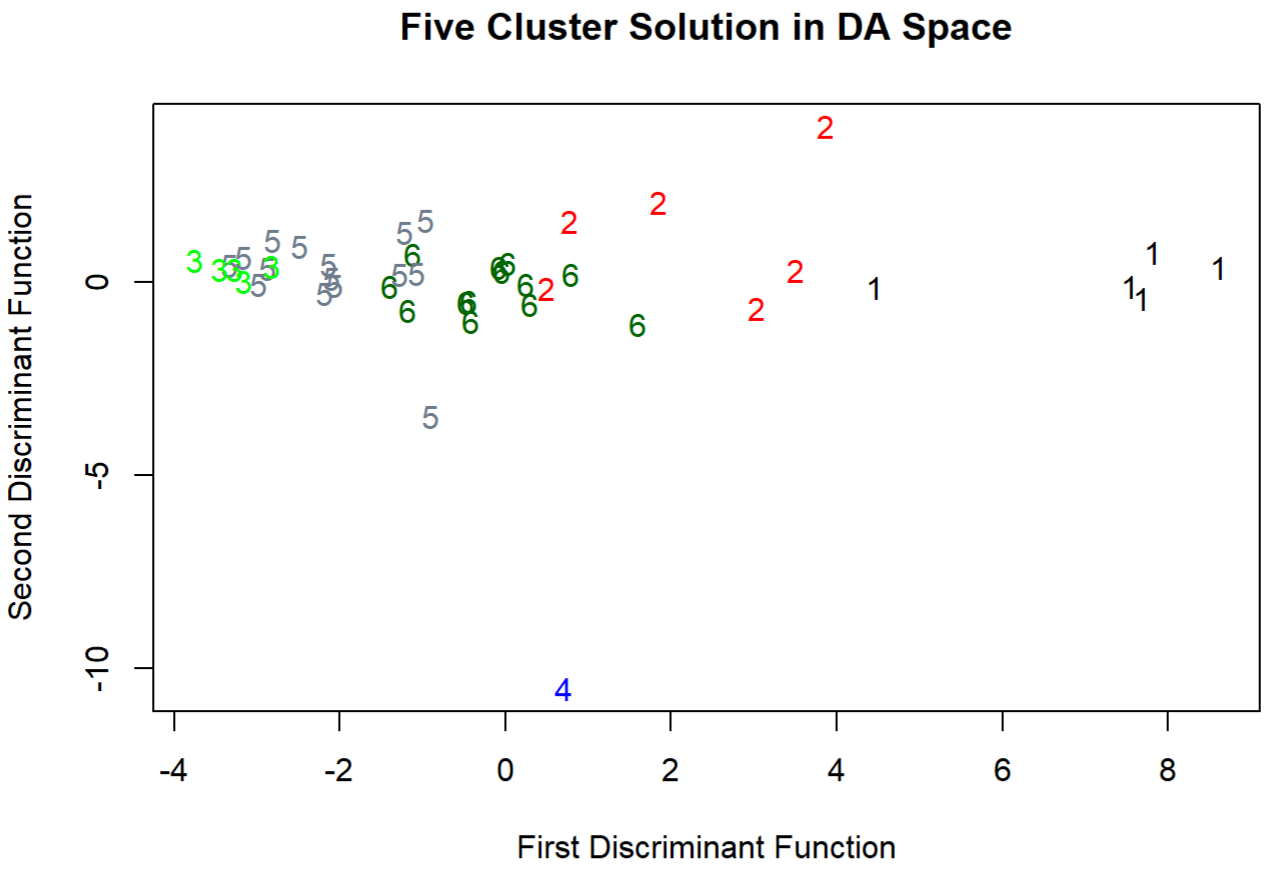
We see that some groups are smaller. This could be the case as for some industries, we only have a few companies, and those companies have a different profitability/risk structure than many of the other companies that belong to different industries.

```
clust.out <- fit$cluster
kclust <- as.matrix(clust.out)
kclust.out <- cbind(kclust, data_f)
# Display Principal Components plot of data with clusters identified
clusplot(kdata, fit$cluster, shade=F, labels=2, lines=0, color=T, lty=4, main='Principal Compone
nts plot showing K-means clusters')
```



Below is our k-means result in discriminant analysis space.

```
plotcluster(kdata, fit$cluster, main="Five Cluster Solution in DA Space",
            xlab="First Discriminant Function", ylab="Second Discriminant Function")
```



PROBLEM 5

5. Comment on the number of groups that seem to be present based on what you find above.

Per our analysis, 7 groups seem to be present. Originally, we hypothesized that there could be 6 to 8 clusters. We based this hypothesis on different industries represented in our data set. Upon analyzing our data through scatter plots, histograms and boxplots, we observed that different industries had different profitability, liquidity and risk characteristics, and we believed that our clustering analysis would represent that.

First we performed hierarchical analysis using both Euclidean and Manhattan techniques to calculate distance, as well as single and complete agglomeration techniques to determine our clusters. After seeing that the first model in which we used single (q2 model 2) gave clusters which were not informational, since most stocks were grouped into one cluster, we decided not to use single agglomeration and proceed with complete agglomeration.

The models that gave us the most information in hierarchical clustering were model 1 and model 3. In both of these models, R appeared to group stocks by aspects of their industry. This meant that the same stock could belong in different clusters between models (for example, a biopharmaceutical and healthcare technology company could be grouped with other technology stocks, or other medical stocks. This makes sense, because when investors are choosing investments, different groups of investors might see the same stock as belonging to different industries, or be interested in them for different reasons. Secondary characteristics of companies also contributed to the degree of hierarchical clustering in some way, which explains variance within clusters. More information about this was obtained through later techniques.

We then analyzed RSQ, RMMSTD plots and concluded in part 3 that there were 7 clusters. This was supported by our distance clustering analysis, as we observed that data points on average followed industry grouping when they were partitioned in 7 clusters.

As observed, not every cluster represented a unique industry. Some similar industries were grouped, while some companies had a slightly different structure than their industry peers. These ‘outlier’ different companies had their own cluster in 2 cases. BIIB (Biogen) and APA (Apache Corporation) were examples. One reason these companies were isolated could be their growth phase differed from their industry peers. Yet, even with this, BIIB is connected to other biotech companies in the next branch, suggesting our earlier hypothesis about industry grouping.

In Question 4, we performed k-means clustering. When we looked at the distance between our internal sum of squares blue line and red line, and ask at what point does the difference stops decreasing (improving), we observed that the optimal number of clusters were 6.

While our analysis in Part 4 offered a cluster value that is one lower than what we concluded in Part 2 and 3, our final conclusion was choosing 7 clusters. This was mainly because cluster number that seemed optimal in Part 4 was changing between 6 and 7, depending on the random iteration. That is, as we were sampling 250 randomly simulated data sets, each simulation was offering a different chart, that favored 6 to 7 clusters. Hence, having 6 clusters in the random iteration in Part 4 did not necessarily contradict with having 7 clusters. Listing out the industries from our stocks shows roughly similar groupings (see appendix).

APPENDIX

List of Industries: 1) manufacturing 2) pharma/health 3) tech 4) transportation 5) utilities 6) oil and gas 7) communications

List of Companies: 1. AAL - American Airlines Group - airline 2. ABBV - AbbVie Inc - pharma 3. ABT - Abbott Laboratories- healthcare 4. ADM - Archer Daniels - food 5. ADS - Alliance Data Systems Corp - marketing 6. AEE - Ameren Corp - utilities 7. AKAM - Akamai Tech - media/tech 8. ALB - Albemarle Corporation - chemicals 9. ALK - Alaska air - airline 10. ALLE - Allegion PLC - security tech 11. ALXN - Alexion Pharma - pharma 12. AME - African Methodist Episcopal - church 13. AMGN - Amgen Inc - biopharma 14. AMT - American Tower Corp - comms 15. AMZN - Amazon - tech 16. AN - AutoNation - cars 17. APA - Apache Corporation - oil & gas 18. APC - Anadarko Petroleum - oil & gas 19. APH - Amphenol Corporation - electronics 20. ARNC - Arconic Inc - manufacturing 21. ATVI - Activision Blizzard, 22. AWK - American Water Works Co - utilities 23. BA - Boeing Co - manufacturing 24. BAX - Baxter International Inc - medical products 25. BCR - CR Bard - biotech 26. BHI - Baker Hughes Inc - oil and gas 27. BIIB - Biogen Inc - biotech 28. BLL - Ball Corporation - manufacturing 29. BMY - Bristol-Myers Squibb - pharma 30. BSX - Boston Scientific Corporation - biotech 31. BWA - BorgWarner Inc - cars 32. CAT - Caterpillar Inc - manufacturing 33. CBG - CBRE Group - real estate 34. CCI - Crown Castle IN/SH - comms 35. CELG - Celgene Corporation - biopharma for cancer and inflammation 36. CF - CF Industries Holdings, Inc - chemicals/agricultural fertilizers 37. CHD - Church & Dwight Co, Inc - household products 38. CHK - Chesapeake Energy Corporation - oil and gas 39. CHRW - C.H. Robinson Worldwide Inc - multimodal transportation and storage 40. CHTR - Charter Communications Inc - media/comms 41. CL - Colgate-Palmolive Company - health care 42. CMG - Chipotle Mexican Grill, Inc - food 43. CMI - Cummins Inc - utilities 44. CMS - CMS Energy Corporation - utilities 45. CNP - CenterPoint Energy, Inc - utilities