Computer Assignment 5 - Hierarchical Clustering

Machine Learning, Spring 2020

Rui Li

```
# If necessary, make sure to install.packages("gplots") so that the following
 command works:
library(gplots)

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

#Load the data
trial.sample = read.table("TCGA_sample.txt", header = TRUE)
#Store the subtypes of tissue and the gene expression data
Subtypes = trial.sample[,1]
Gene.Expression = as.matrix(trial.sample[,2:2001])
```

Hierarchical Clustering - TCGA Data

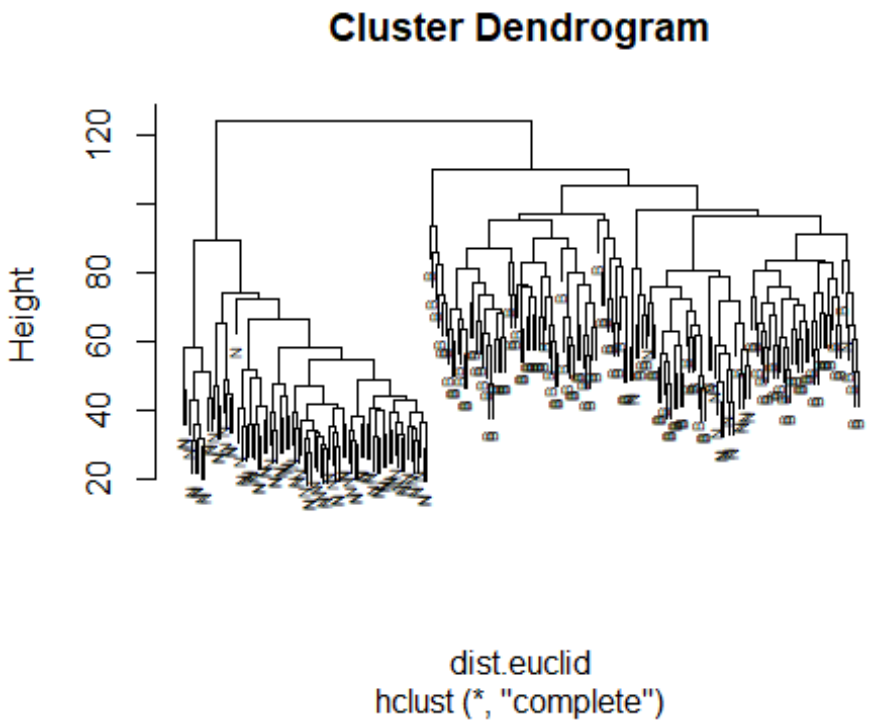
Because *k*-means works by finding the physical mean point in space for each cluster, we give it the raw data as input. However, our next two methods do not need the raw data, only the distances between points. As we are clustering the patients, we first must calculate pairwise distances between the patients. The *dist(X, method)* function can be used to calculate the pairwise distances between the rows of a matrix *X* using the specified *method* as a distance metric. Calculate the average Euclidean and average absolute pairwise distances between patients (a 217 × 217 matrix) using the following code:

```
dist.euclid = dist(Gene.Expression, method = "euclidean")
dist.abs = dist(Gene.Expression, method = "manhattan") #absolute distance
```

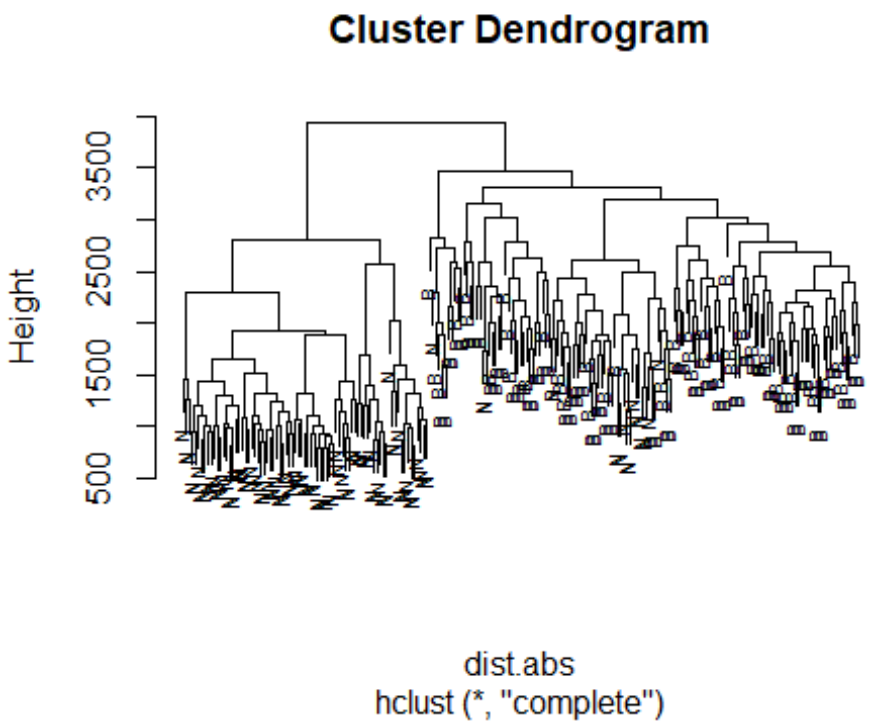
There are many ways in **R** to perform hierarchical clustering. We will use the most basic version, *hclust()*. Run hierarchical clustering on the patients using the Euclidean and absolute distance matrix using the following code. Using *plot()* on the output object automatically draws the dendrogram. For a quick look at how well these clusters match the true cancer subtypes, we use *labels = types*.

```
types = rep(0,217)
types[which(Subtypes == "Basal")] = "B"
types[which(Subtypes == "Normal")] = "N"
```

```r
#Euclidean distance
hc.euclid = hclust(dist.euclid)
plot(hc.euclid, labels = types, cex = 0.5)
```

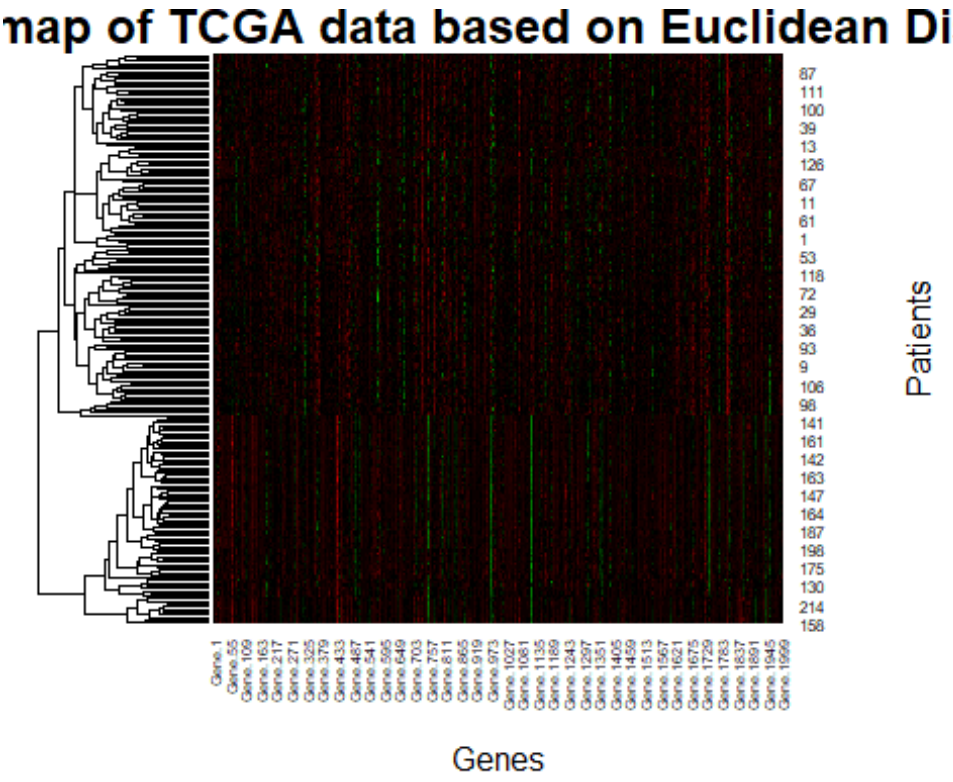



```r
#Absolute distance
hc.abs = hclust(dist.abs)
plot(hc.abs, labels = types, cex = 0.6)
```



Note that the dendrograms are generated based on pairwise distances between the patients. We can visualize this by using *heatmap( )* to plot the original data according to the hierarchical cluster ordering.

```
heatmap(Gene.Expression, Rowv = as.dendrogram(hc.euclid), Colv = NA,
        main = "Heatmap of TCGA data based on Euclidean Distance",
        xlab = "Genes", ylab = "Patients", col = redgreen(50))
```



Questions

1. Comment on the differences between the two dendrograms. Does one distance appear to cluster differently than the other? Which do you prefer?

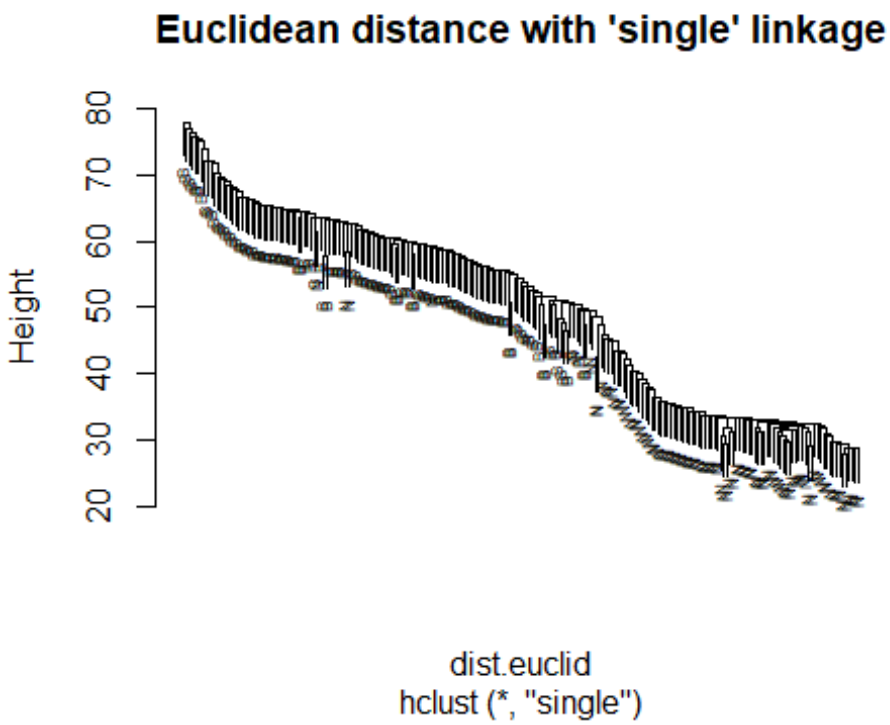
```
Both dendrograms have generally cluster the two subtypes into the two groups.
 However, the group on the right has mixed some 'N' type cases. For 'Euclidea
n Distance', those 'N' cases belong to the sub groups close to each other, wh
ile the 'Absolute Distance' has some cases belong to sub groups far away from
 each other. Therefore, I prefer the 'Euclidean Distance'.
```

2. What agglomeration (linkage) method was used for the above clusterings? (Hint: check the manual page)

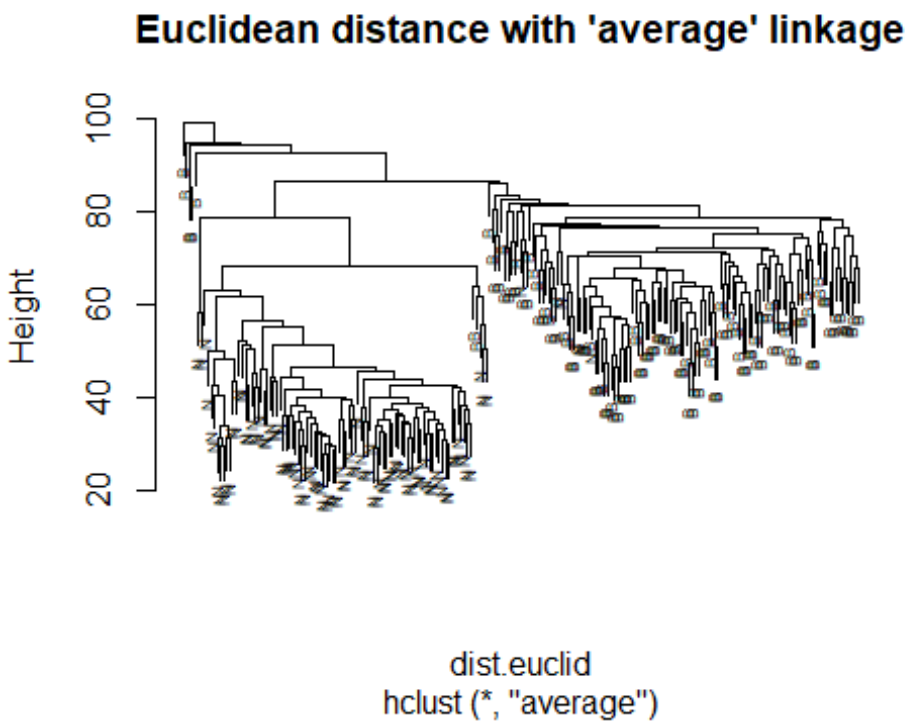
```
According to the manual page, the default method of 'hclust' is complete.
```

3. Perform the euclidean distance clustering with `single` and `average` linkage. Print out their respective dendrograms and comment on how they differ. Do they differ at all from the above euclidean distance clustering?

```
#Euclidean distance with 'single' linkage
hc.euclid.s = hclust(dist.euclid, method = "single")
plot(hc.euclid.s, labels = types, cex = 0.5, main = "Euclidean distance with
'single' linkage")
```



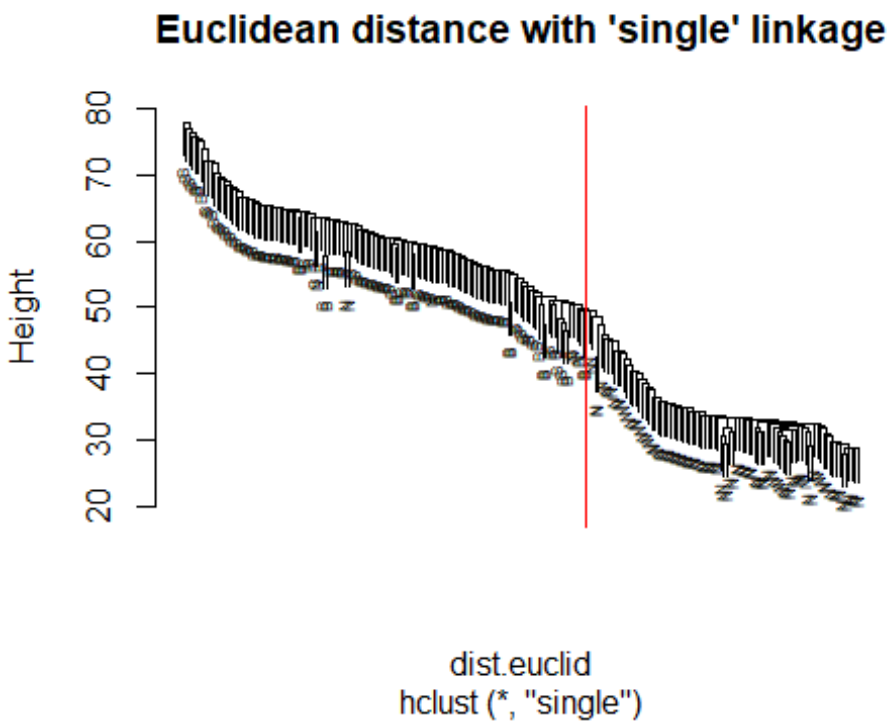

```r
#Euclidean distance with 'average' linkage
hc.euclid.a = hclust(dist.euclid, method = "average")
plot(hc.euclid.a, labels = types, cex = 0.5, main = "Euclidean distance with
'average' linkage")
```



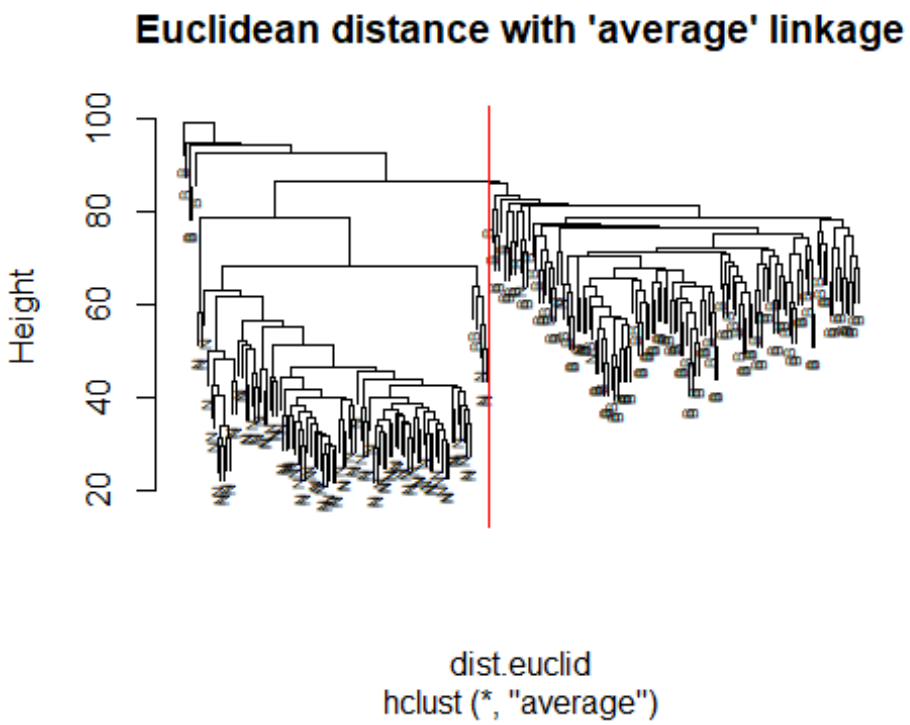

```
The 'single' linkage forms as a line, with single sample in each group, while
 the 'average' linkage scatters the patients into different groups. The 'sing
le' looks totally different from the above euclidean distance clustering, and
 the plot shows groups in a linear way. The 'average' generally looks the sam
e as the above euclidean distance clustering, and largely sperates two sub ty
pes in to two groups.
```

4. Hierarchical clustering does not automatically make a certain number of clusters from the data - this depends on where you "cut" the dendrogram. Draw a line on your plots showing where you cut the dendrogram. How closely do the subtypes appear to cluster in these two groups?

```
#Euclidean distance with 'single' linkage
plot(hc.euclid.s, labels = types, cex = 0.5, main = "Euclidean distance with
'single' linkage")
abline(v=130,col='red')
```



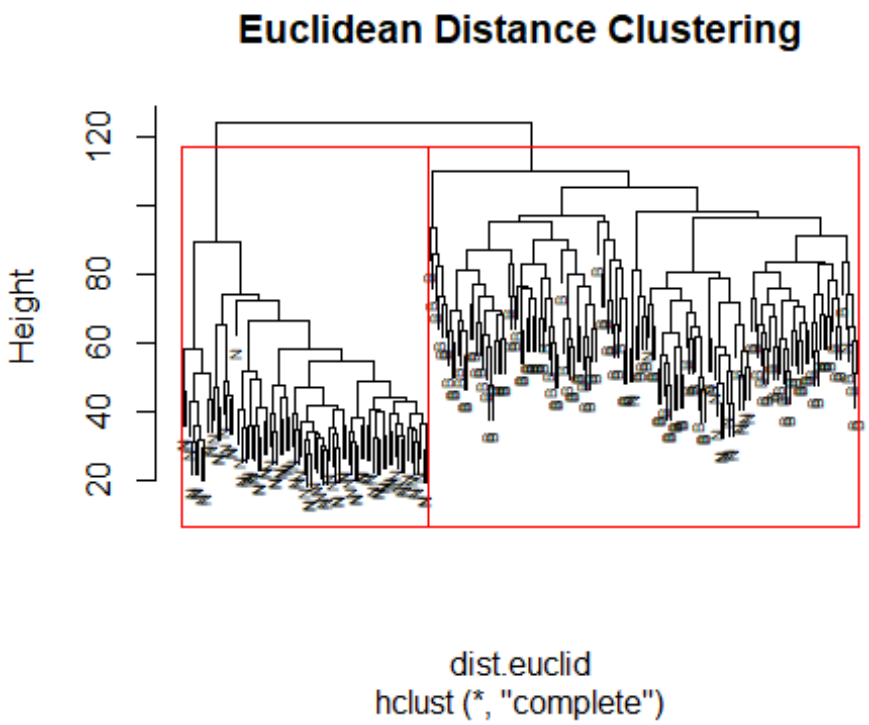

```r
#Euclidean distance with 'average' linkage
plot(hc.euclid.a, labels = types, cex = 0.5, main = "Euclidean distance with
'average' linkage")
abline(v=99,col='red')
```




```
For the 'single', the subtypes show similarly as the groups in clustering. Ho
wever, for the 'average', two subtypes have some sample mixed into the other
group of clustering.
```

5. The function *cutree(tree, …)* will produce clusters based on a certain cut of the dendrogram *tree*. We can specify either height (*h*) or number of clusters (*k*). Use *cutree()* on the Euclidean distance clustering to assign clusters. What percentage of the each cluster is Normal and Basal? (Hint: Use *?cutree* to figure out exactly how to use this function.)

```
ct.n <- cutree(hc.euclid,k=2)
plot(hc.euclid, labels = types, cex=0.5, main = "Euclidean Distance Clusterin
g")
rect.hclust(hc.euclid, 2)
```




```
library(plyr)
#Count Cluster 1 and 2
cluster_1 = count(ct.n)[1,2]
cluster_2 = count(ct.n)[2,2]

#Count 'Normal' and 'Basal'
cluster_1_Basal = count(ct.n==1&Subtypes=='Basal')[2,2]
count(ct.n==2&Subtypes=='Basal')[2,2]# The value is NA.

## [1] NA

cluster_2_Basal = 0
cluster_1_Normal = count(ct.n==1&Subtypes=='Normal')[2,2]
cluster_2_Normal = count(ct.n==2&Subtypes=='Normal')[2,2]

print(paste("Cluster 1 contains 'Normal': ", cluster_1_Normal/cluster_1))

## [1] "Cluster 1 contains 'Normal':  0.115942028985507"

print(paste("Cluster 1 contains 'Basal': ", cluster_1_Basal/cluster_1))

## [1] "Cluster 1 contains 'Basal':  0.884057971014493"

print(paste("Cluster 2 contains 'Normal': ", cluster_2_Normal/cluster_2))

## [1] "Cluster 2 contains 'Normal':  1"

print(paste("Cluster 2 contains 'Basal': ", cluster_2_Basal/cluster_2))
```

```
## [1] "Cluster 2 contains 'Basal':  0"
```

6. Suppose that you read a scientific paper where the authors use hierarchical clustering on a data set and show a figure similar to what you just created. What kinds of questions might you be inclined to ask the authors regarding their clustering? Does the flexibility of the clustering (the choice of distance, the choice of linkage, etc.) make you more or less confident in the authors' clustering?

```
Since different choice of distance calculation will affect the quality of clu
stering. I will ask how and why he choose such a way of distance. Also, diffe
rent choice of linkage will have different result. I will ask how he know whi
ch choice is more appropriate to the data, and how he choose it.
```

Hierarchical Clustering - Cereals

Now let's practice Hierarchical Clustering (HC) with a different data set. The cereal data (named `cereals.csv`) contains cereal brands, manufacturers (also a variable called group, which is the same info, but group is numeric and manufacturer is categorical), and nutrition information (calories, protein, fat, sodium, fiber, carbs, sugar, potassium) per serving. Do a brief analysis of the data.

```
library(readr)
cereals <- read_csv("cereals.csv")

## Warning: Missing column names filled in: 'X1' [1]

## Parsed with column specification:
## cols(
##   X1 = col_double(),
##   brand = col_character(),
##   manufacturer = col_character(),
##   calories = col_double(),
##   protein = col_double(),
##   fat = col_double(),
##   sodium = col_double(),
##   fiber = col_double(),
##   carbs = col_double(),
##   sugar = col_double(),
##   potassium = col_double(),
##   group = col_double()
## )

summary(cereals)

##        X1          brand           manufacturer         calories
##  Min.   : 1.0   Length:43          Length:43          Min.   : 50.0
##  1st Qu.:11.5   Class :character   Class :character   1st Qu.:100.0
##  Median :22.0   Mode  :character   Mode  :character   Median :110.0
##  Mean   :22.0                                         Mean   :107.9
##  3rd Qu.:32.5                                         3rd Qu.:110.0
```

```
##  Max.   :43.0                                          Max.   :160.0
##     protein           fat              sodium          fiber
##  Min.   :1.000   Min.   :0.0000   Min.   :  0.0   Min.   :0.000
##  1st Qu.:2.000   1st Qu.:0.0000   1st Qu.:145.0   1st Qu.:0.500
##  Median :2.000   Median :1.0000   Median :190.0   Median :1.000
##  Mean   :2.465   Mean   :0.9767   Mean   :180.5   Mean   :1.714
##  3rd Qu.:3.000   3rd Qu.:1.5000   3rd Qu.:220.0   3rd Qu.:2.850
##  Max.   :6.000   Max.   :3.0000   Max.   :320.0   Max.   :9.000
##      carbs           sugar          potassium          group
##  Min.   : 1.00   Min.   : 0.000   Min.   : 15.00   Min.   :1.000
##  1st Qu.:12.00   1st Qu.: 3.000   1st Qu.: 37.50   1st Qu.:1.000
##  Median :14.00   Median : 8.000   Median : 60.00   Median :2.000
##  Mean   :14.26   Mean   : 7.605   Mean   : 84.42   Mean   :1.744
##  3rd Qu.:17.00   3rd Qu.:12.000   3rd Qu.:110.00   3rd Qu.:2.000
##  Max.   :22.00   Max.   :15.000   Max.   :320.00   Max.   :3.000
```

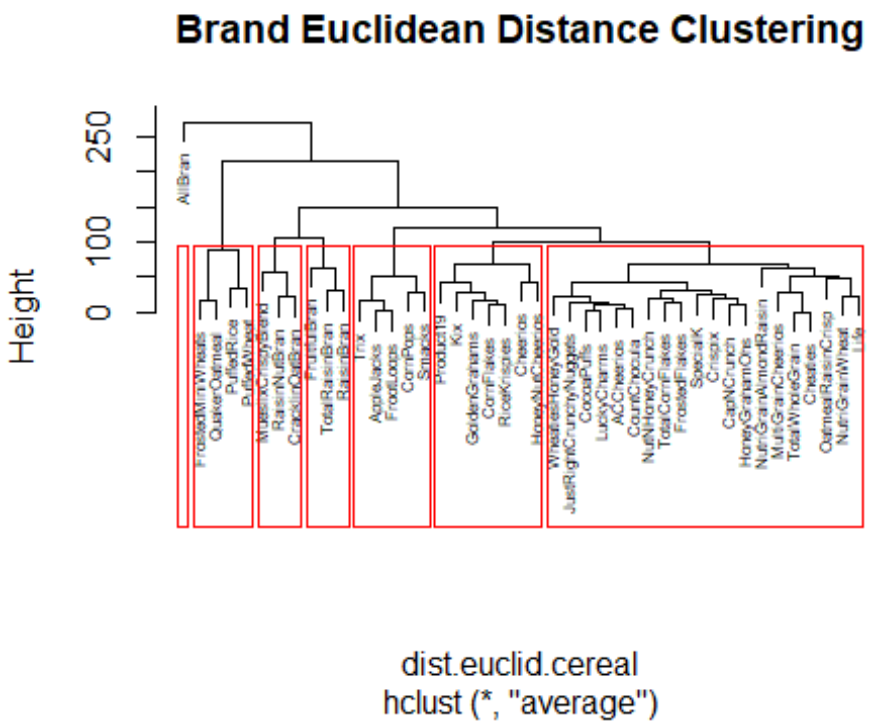
Perform HC using euclidean distance and `average` linkage and all variables *except* `brand`, `manufacturer`, and `group`. Make sure to plot the dendogram, and label each leaf by `brand`. Do you see any interesting clusterings based on cereal brands? Use the `cutree()` function and experiment with *h* and *k* to see if you can find any meaningful clusters.

```r
#Store the subtypes of brand and the cereals data
subtypes_brand = cereals[,2]
cereal_data = as.matrix(cereals[,-c(1:3,12)])

#Calculate the euclidean distance
dist.euclid.cereal = dist(cereal_data, method = "euclidean")

#Plot the dendrogram
hc.cereal=hclust(dist.euclid.cereal, method="average")
plot(hc.cereal, labels=cereals$brand, cex=0.5, main = "Brand Euclidean Distan
ce Clustering")

#Cutree()
ct.cereal <- cutree(hc.cereal,k=7)
rect.hclust(hc.cereal, 7)
```

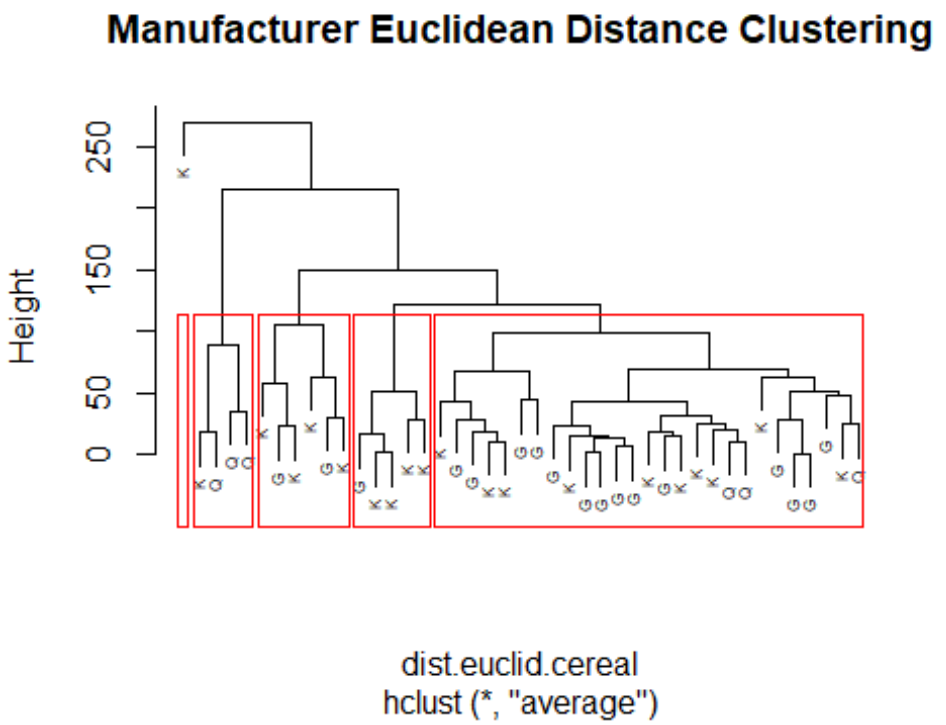



```
There are some clustering based on the brand, but it is not obvious, each bra
nd belongs to a sub group.
```

Now, change the labels on the dendrogram to `manufacturer` and see if you can find a meaningful *k* and *h*. Comment on your findings.

```
plot(hc.cereal, labels=cereals$manufacturer, cex=0.5, main = "Manufacturer Eu
clidean Distance Clustering")

#Cutree()
ct.cereal <- cutree(hc.cereal,k=5)
rect.hclust(hc.cereal, 5)
```



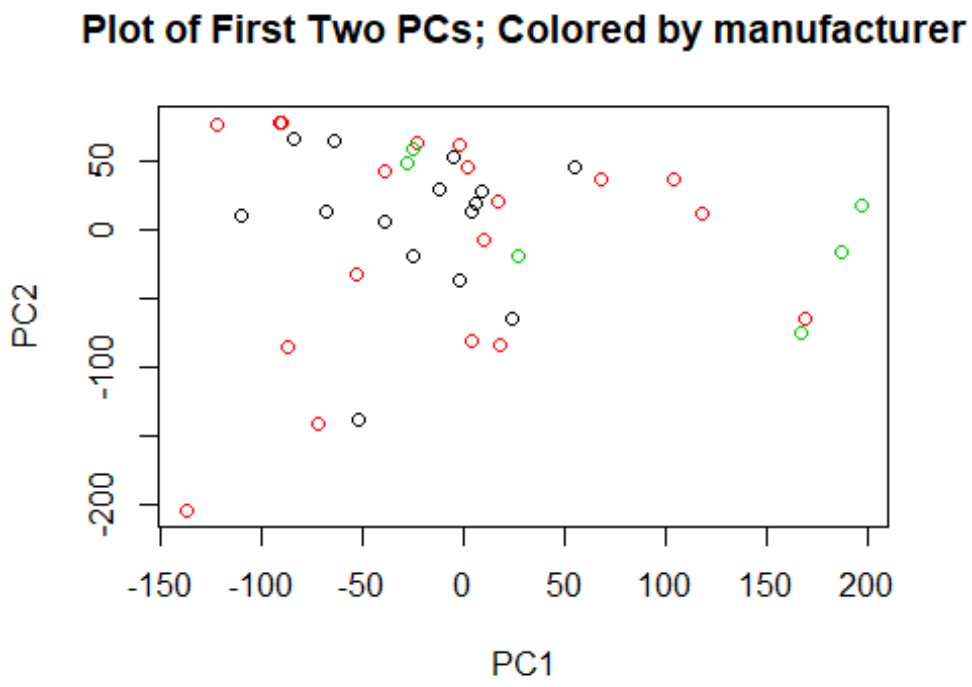
Generally, there is a meaningful k=5, largely separates subtypes of G,K,Q.

Using the *k* you decided upon using HC with `manufacturer` as your labels, run a k-means clustering on the cereal data. Compare to the clustering you found using HC. This comparison can be done a number of ways: you can project onto the first two PCs and plot, you can print out the cluster labels from each method and calculate how often the matched/didn't match, etc.

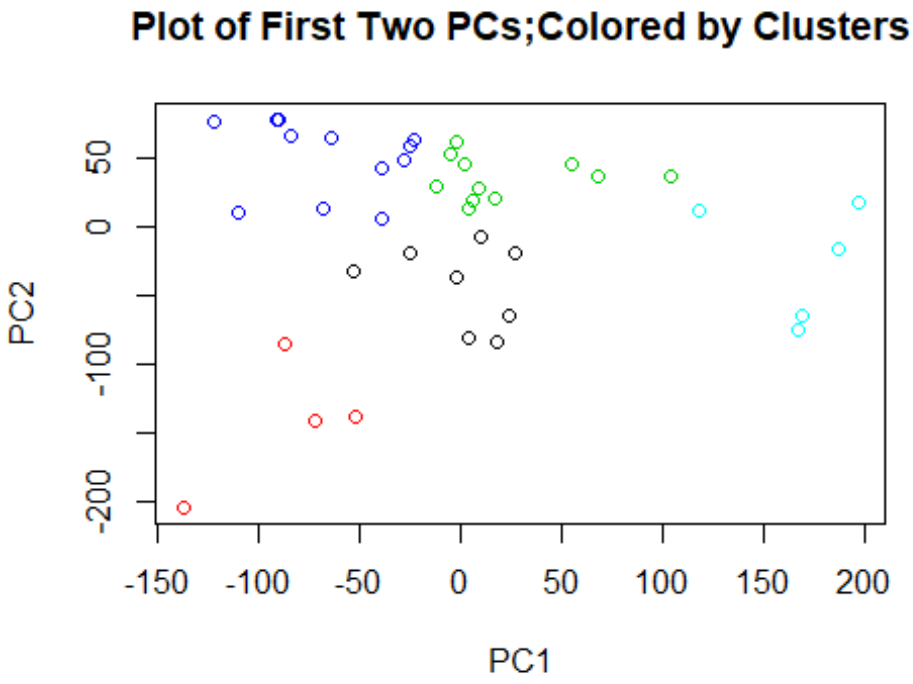
Note: This question is intentionally open ended. Use code from previous assignments, and suggestions online, to provide a FULL hierarchical clustering analysis of this data.

```
#PCA on the dataset cereals
pc.cereal = prcomp(cereal_data)

#Plot the projections in the first two PC dimensions.
cereal.pred = predict(pc.cereal, cereal_data)
plot(cereal.pred[,1:2],col=as.factor(cereals$manufacturer), main = "Plot of F
irst Two PCs; Colored by manufacturer")
```




```r
#K=5
k_5 = kmeans(cereal_data,5,iter.max = 100)
plot(cereal.pred[,1:2], col=k_5$cluster, main = "Plot of First Two PCs;Colore
d by Clusters")
```


Plot of First Two PCs;Colored by Clusters
PC2
50
0
-100
-200
-150
-100
-50
0
50
100
150
200
PC1