assign4

Shaoyi Zhang

May 24, 2016

Question 1.1

```
setwd("/Users/Shawn/Desktop/PSTAT 231/PSTAT-231/assign4")
set.seed(2)
library(data.table)
food.data = read.table("food.txt",header = T,row.names = 1)
st.food = scale(food.data)
```

If we choose to have 2 clusters:

```
km.2 = kmeans(food.data[2:ncol(food.data)], centers = 2, nstart = 50)

# the centroids
km.2$centers
```

```
##    Protein       Fat  Calcium     Iron
## 1 19.08333 14.208333  21.0000 2.470833
## 2 18.33333  7.666667 227.6667 1.666667
```

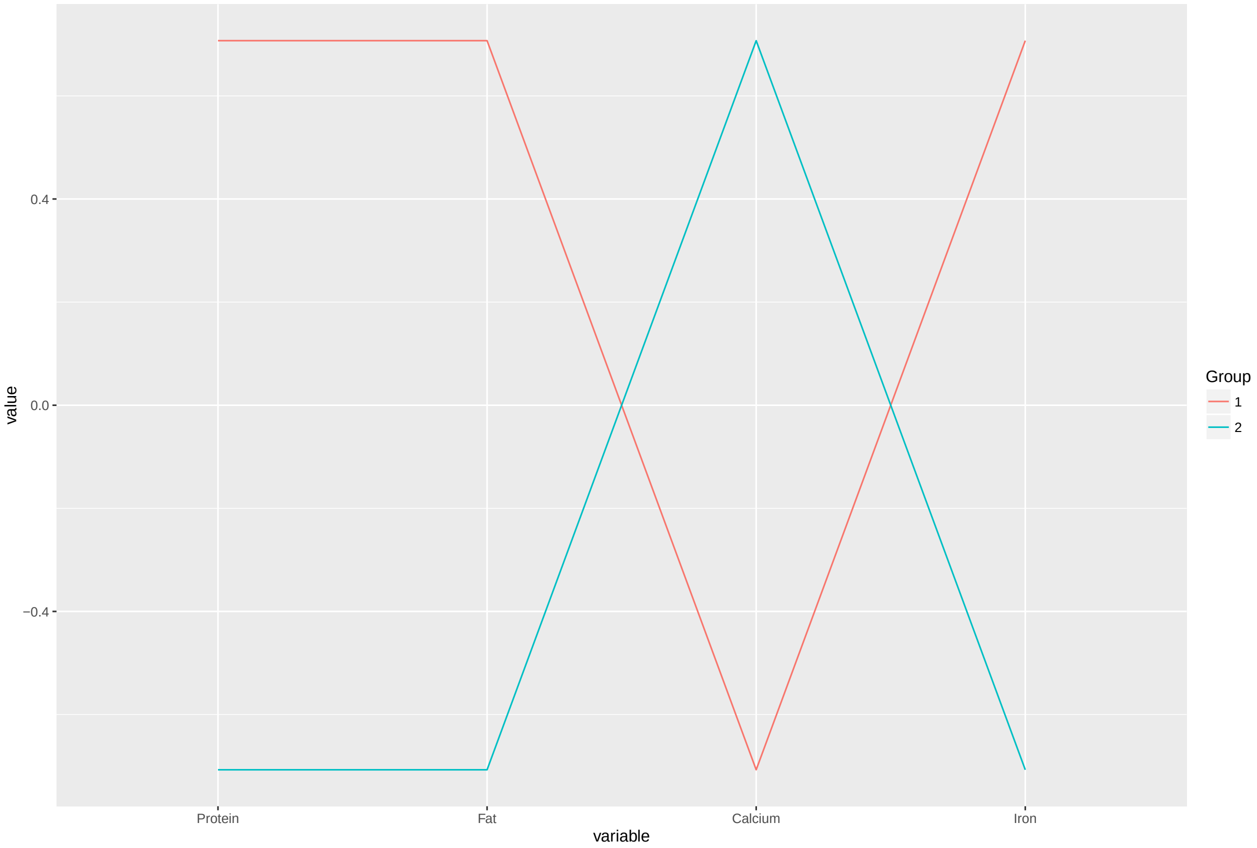
```
# The size of each center
km.2$size
```

```
## [1] 24  3
```

```
# The ratio of between-SS/total-SS
km.2$betweenss/km.2$totss
```

```
## [1] 0.7031662
```

```
centers = as.data.table(km.2$centers)
centers[,Group:=as.factor(c(1,2))]
library(GGally)
gpd <- ggparcoord(data = centers, columns = 1:4, groupColumn = 5)
gpd
```



Group 1 has high Protein, Fat and Iron, but Low Calcium.

Group 2 has High Calcium, but low Protein, Fat and Iron.

```
# Group membership
km.2$cluster
```

```
##        Braised beef           Hamburger          Roast beef
##                   1                   1                   1
##           Beefsteak         Canned beef     Broiled chicken
##                   1                   1                   1
##      Canned chicken          Beef heart      Roast lamb leg
##                   1                   1                   1
## Roast lamb shoulder          Smoked ham          Pork roast
##                   1                   1                   1
##       Pork simmered         Beef tongue         Veal cutlet
##                   1                   1                   1
##      Baked bluefish           Raw clams        Canned clams
##                   1                   1                   1
##     Canned crabmeat       Fried haddock    Broiled mackerel
##                   1                   1                   1
##     Canned mackerel         Fried perch       Canned salmon
##                   2                   1                   2
##     Canned sardines         Canned tuna       Canned shrimp
##                   2                   1                   1
```

From the cluster membership, we noticed that Group 2 are all Canned fish. Since seafood tends to have high

Calcium, our grouping is reasonable. But there are still many “fish” categorized in Group 1 which are mostly beef/lamb/chicken.

Then, if we choose to have 3 clusters:

```
km.3 = kmeans(food.data[2:ncol(food.data)], centers = 3, iter.max = 10, nstart = 50)

# the centroids
km.3$centers
```

```
##    Protein      Fat   Calcium     Iron
## 1 14.80000  3.40000 114.00000 3.300000
## 2 19.85714 16.09524  11.90476 2.157143
## 3 22.00000  9.00000 367.00000 2.500000
```

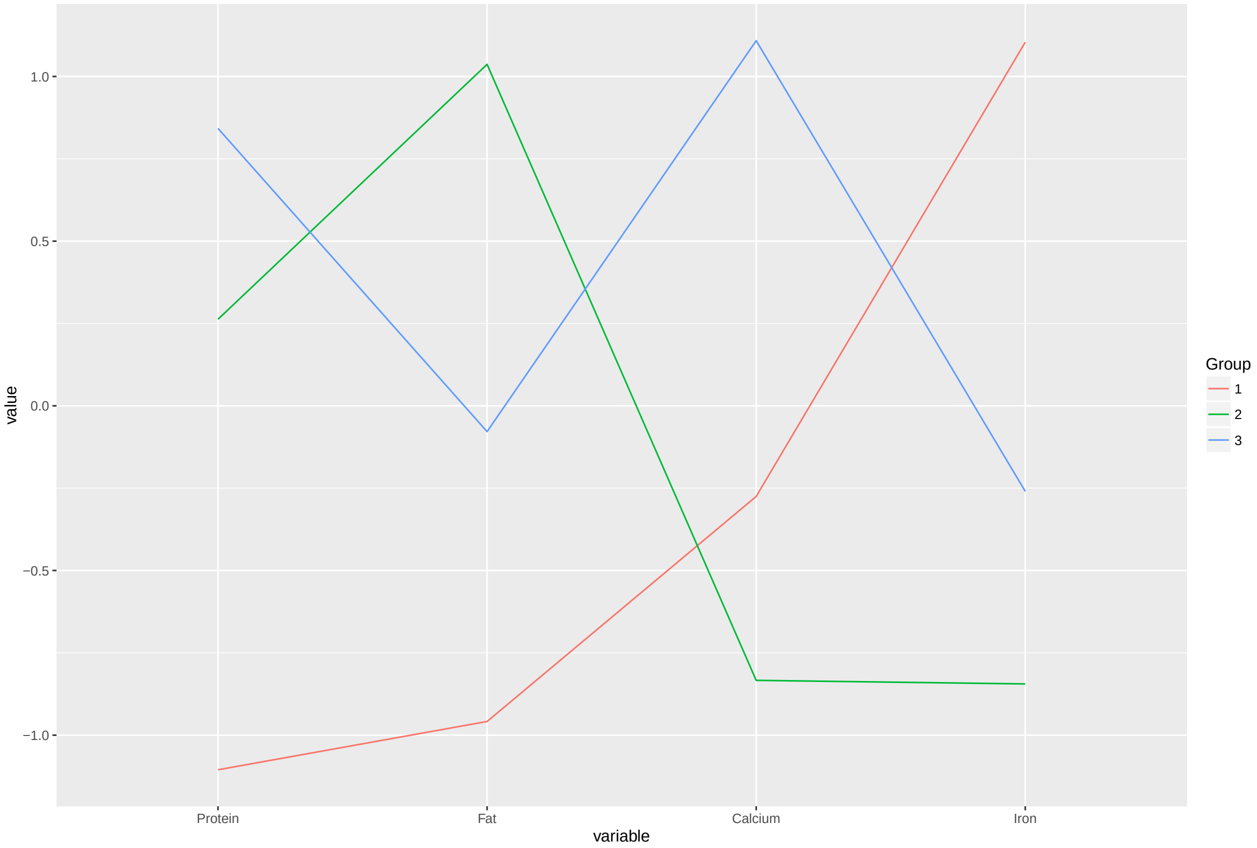
```
# The size of each center
km.3$size
```

```
## [1]  5 21  1
```

```
# The ratio of between-SS/total-SS
km.3$betweenss/km.2$totss
```

```
## [1] 0.9328222
```

```
centers = as.data.table(km.3$centers)
centers[,Group:=as.factor(c(1,2,3))]
library(GGally)
gpd <- ggparcoord(data = centers, columns = 1:4, groupColumn = 5)
gpd
```



Group 1 have high Fat, medium Protein, but low Calcium and Iron.

Group 2 have high Iron, medium Calcium, but low Protein and Fat.

Group 3 have high Protein and Calicium, but low Fat and Iron.

```
# Group membership
km.3$cluster
```

```
##        Braised beef           Hamburger          Roast beef
##                   2                   2                   2
##           Beefsteak         Canned beef     Broiled chicken
##                   2                   2                   2
##      Canned chicken          Beef heart      Roast lamb leg
##                   2                   2                   2
## Roast lamb shoulder          Smoked ham          Pork roast
##                   2                   2                   2
##       Pork simmered         Beef tongue         Veal cutlet
##                   2                   2                   2
##      Baked bluefish           Raw clams        Canned clams
##                   2                   1                   1
##     Canned crabmeat       Fried haddock    Broiled mackerel
##                   2                   2                   2
##     Canned mackerel         Fried perch       Canned salmon
##                   1                   2                   1
##     Canned sardines         Canned tuna       Canned shrimp
##                   3                   2                   1
```

```r
km.4 = kmeans(food.data[2:ncol(food.data)], centers = 4, iter.max = 10, nstart = 50)

# the centroids
km.4$centers
```

```
##    Protein      Fat   Calcium     Iron
## 1 22.00000  9.00000 367.00000 2.500000
## 2 13.66667  1.00000  84.66667 4.666667
## 3 16.50000  7.00000 158.00000 1.250000
## 4 19.85714 16.09524  11.90476 2.157143
```

```r
# The size of each center
km.4$size
```

```
## [1]  1  3  2 21
```

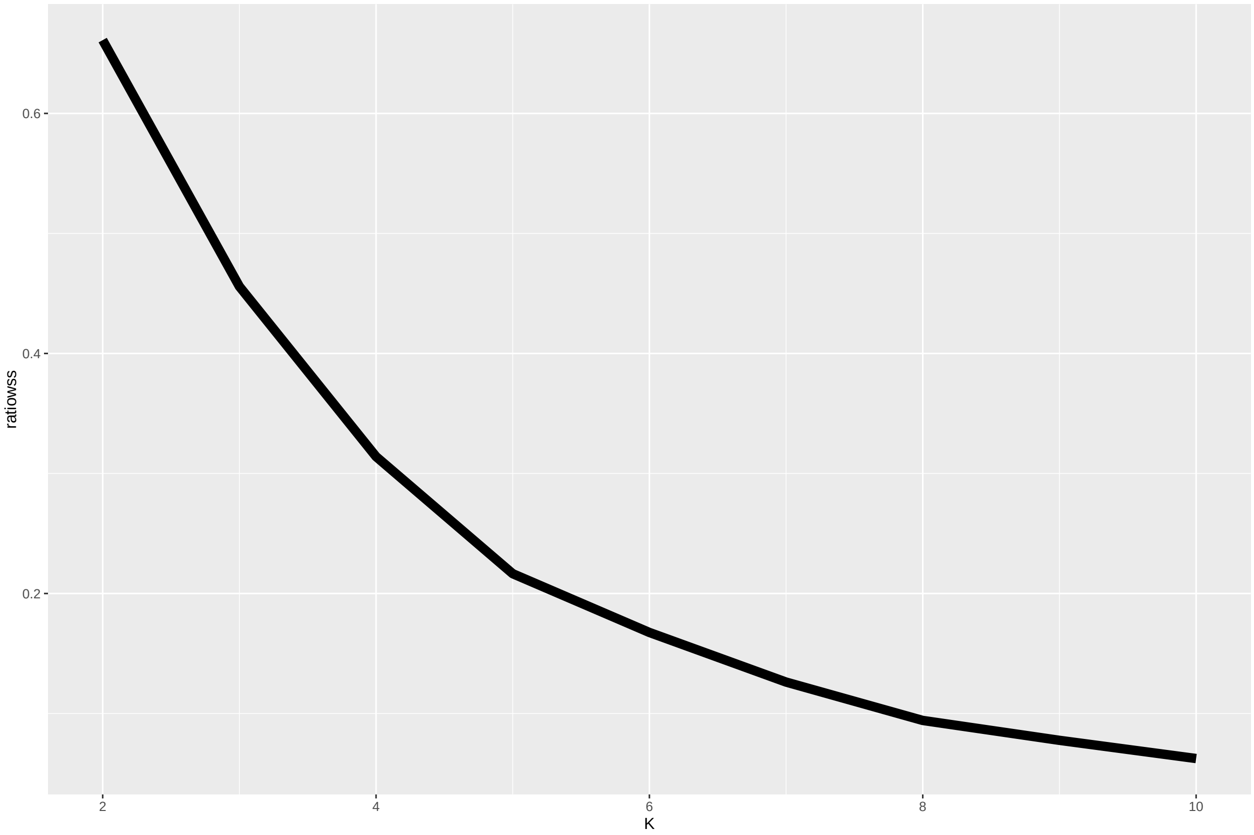
```r
# The ratio of between-SS/total-SS
km.4$betweenss/km.2$totss
```

```
## [1] 0.9730347
```

The ratio of between-SS/total-SS didn't inrease much when we increase K = 3 to K = 4. This means 3 clusters have differentiated our data set very well. We will continue our analysis with Optimal K = 3.

Before we continue, it's also important to see the graph of the ratio of between-SS/total-SS

```r
library(ggplot2)
maxK = 10
st.food=scale(food.data[c(1:5)])
# k- means clustering loop
ratiowss=vector()
for (k in 2:maxK){
  km=kmeans(st.food,k,nstart=50)
  ratiowss[k]=km$tot.withinss/km$totss
}
dt=data.table("K"=2:maxK,"ratiowss"=ratiowss[2:maxK])
ggplot(dt,aes(x=K,y=ratiowss))+geom_line(size=3)
```



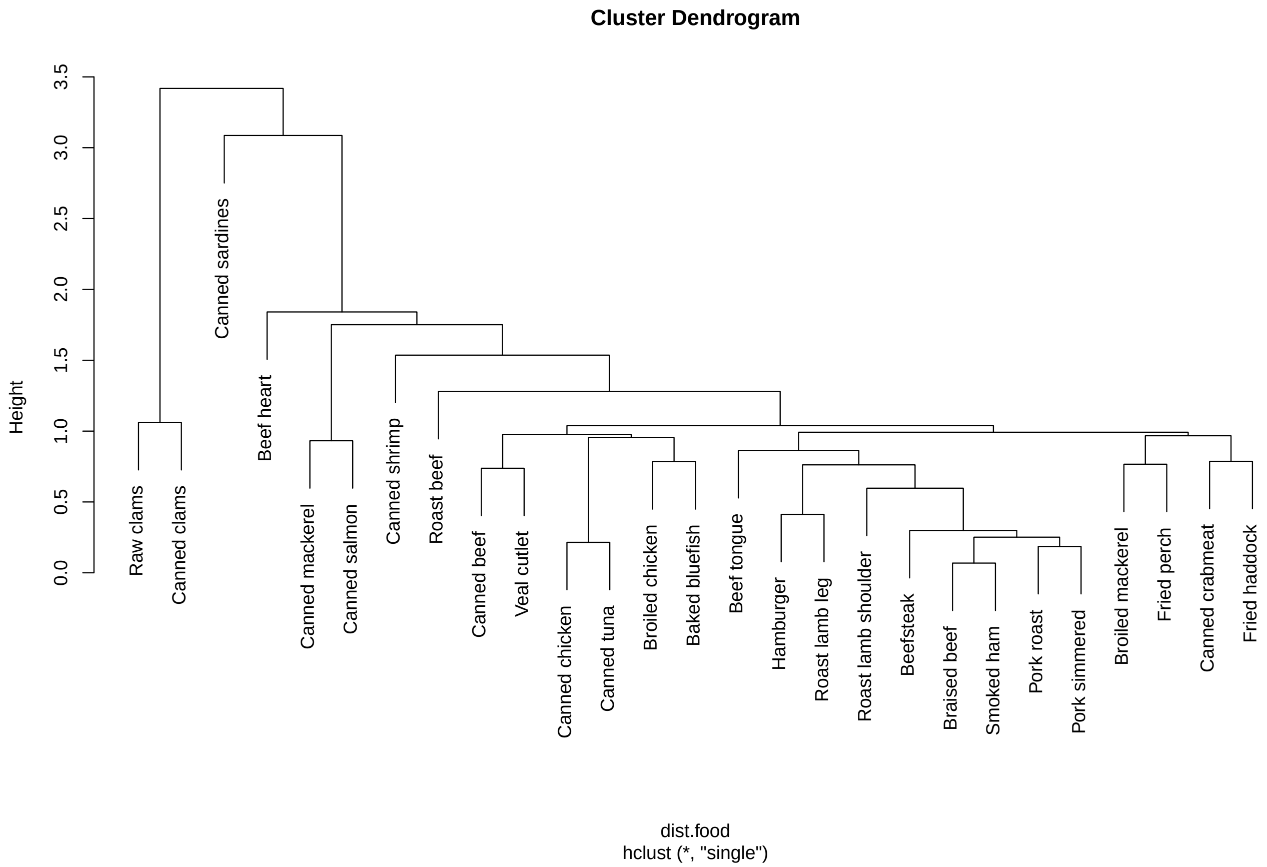
As we increase the number of cluster, the ratio of between-SS/total-SS will always increase. That's expected because when we add new centroids, there must be reduction in distance from some points to the new centriod.

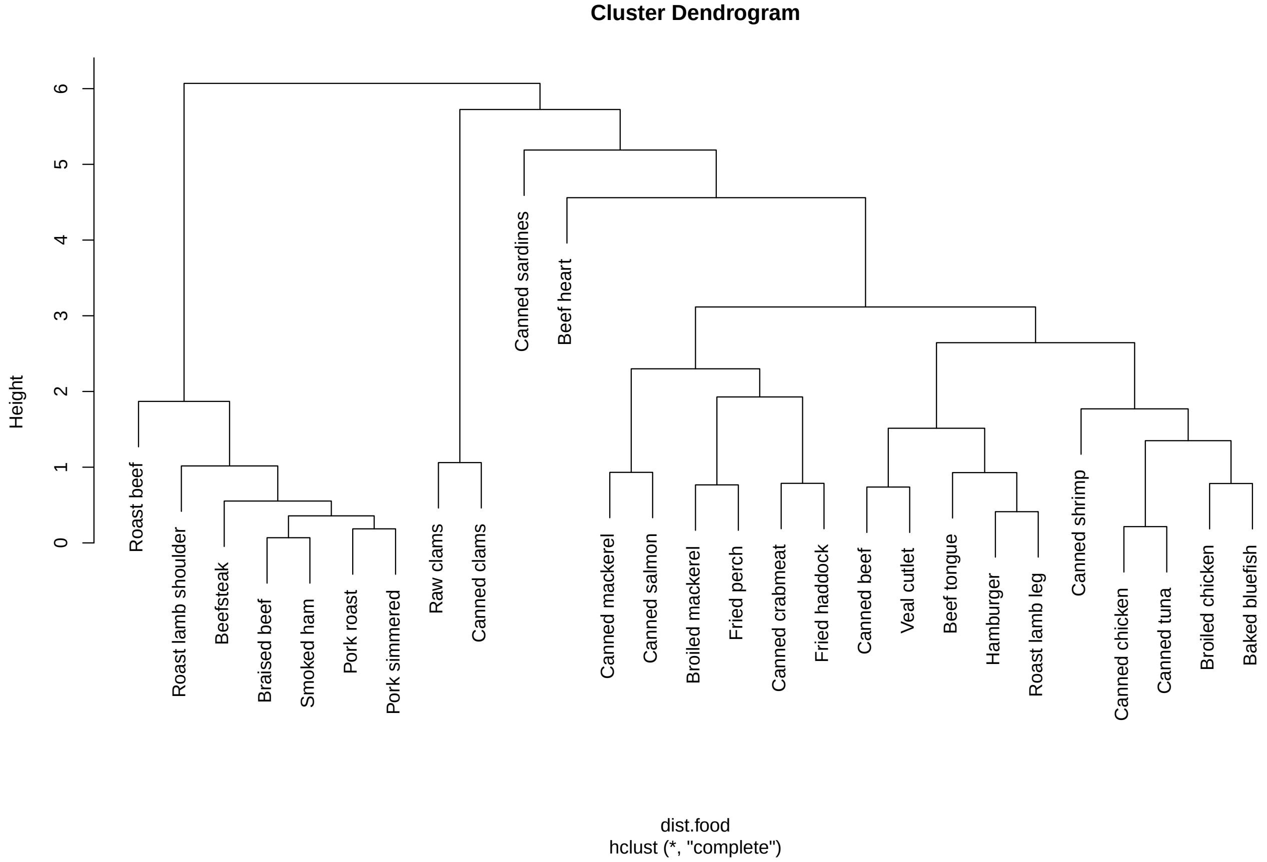
```r
# use clusters = 3 for further analysis
km.out = kmeans(st.food,centers = 3)
centers = data.table(km.out$centers)
```

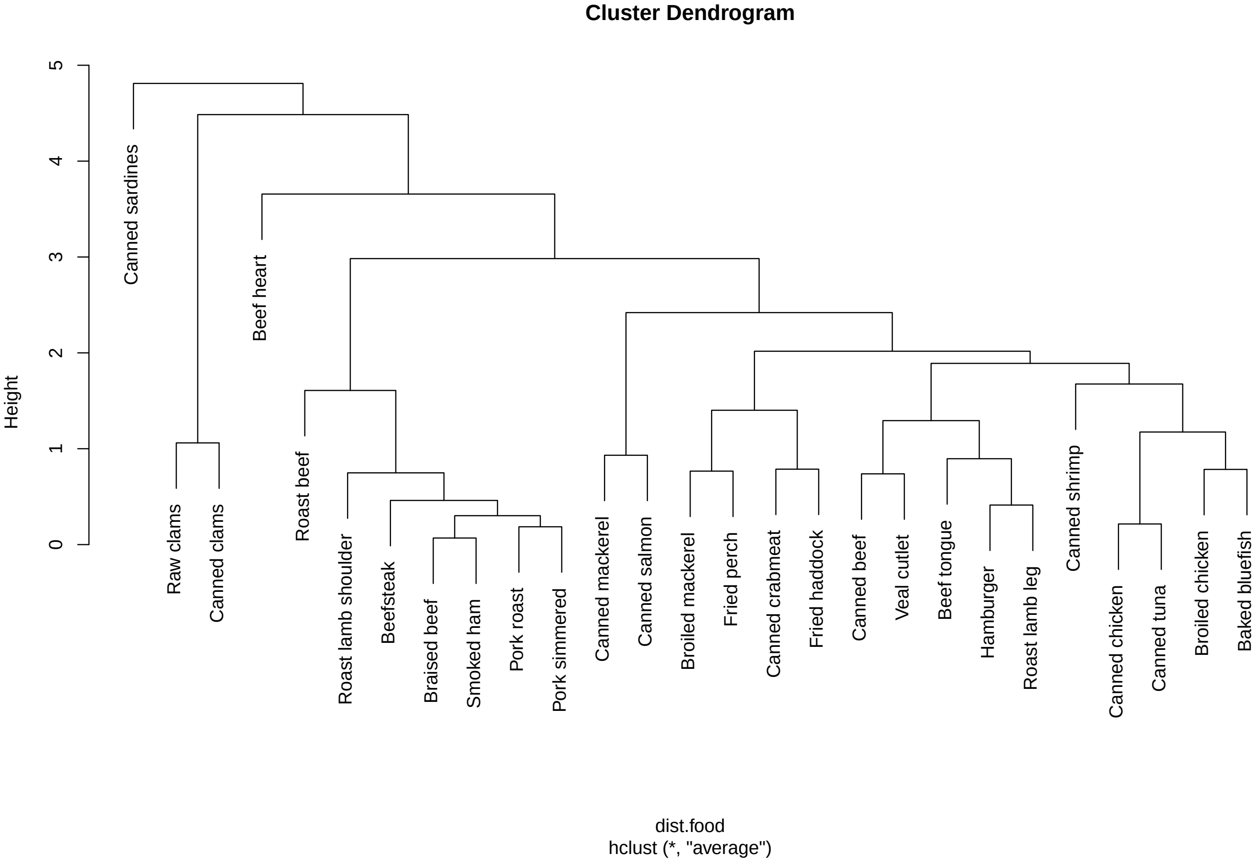
Let's take a look at the centers of the clusters

Question 1.2

Now, we will try Hierachical Clustering.


Cluster Dendrogram
Height
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
Raw clams
Canned clams
Canned sardines
Beef heart
Canned mackerel
Canned salmon
Canned shrimp
Roast beef
Canned beef
Veal cutlet
Canned chicken
Canned tuna
Broiled chicken
Baked bluefish
Beef tongue
Hamburger
Roast lamb leg
Roast lamb shoulder
Beefsteak
Braised beef
Smoked ham
Pork roast
Pork simmered
Broiled mackerel
Fried perch
Canned crabmeat
Fried haddock
dist.food
hclust (*, "single")


Cluster Dendrogram
Height
0
1
2
3
4
5
6
Roast beef
Roast lamb shoulder
Beefsteak
Braised beef
Smoked ham
Pork roast
Pork simmered
Raw clams
Canned clams
Canned sardines
Beef heart
Canned mackerel
Canned salmon
Broiled mackerel
Fried perch
Canned crabmeat
Fried haddock
Canned beef
Veal cutlet
Beef tongue
Hamburger
Roast lamb leg
Canned shrimp
Canned chicken
Canned tuna
Broiled chicken
Baked bluefish
dist.food
hclust (*, "complete")



In min-distance method, we get the dendrogram with the lowest height.

In average-distance method, it tends to seperate one “outlier” out of the group in each split.

In max-distance method, it to have a more “cluster-like” dendrogram compared to average-distance method.

I can individuate clusters in Hierachical clustering similar in K-means clustering. However, it’s more difficult now. In K-means clustering, since there aren’t any “Subgroup”, the interpretation is straightforward – simply summarize the centroids. However, in Hierachical clustering, groups, except leaves, are always subgroups of more general groups. In Hierachical clustering, we can clearly see that food with high Protein and Fat are grouped together. The h-cluster algorithm also does a good job on split seafood from the others.

Question 2

```
pr.out = prcomp(food.data[1:5], center = T, scale=T)
summary(pr.out)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4     PC5
## Standard deviation     1.4820 1.0705 0.9207 0.8992 0.04002
## Proportion of Variance 0.4393 0.2292 0.1695 0.1617 0.00032
## Cumulative Proportion  0.4393 0.6684 0.8380 0.9997 1.00000
```

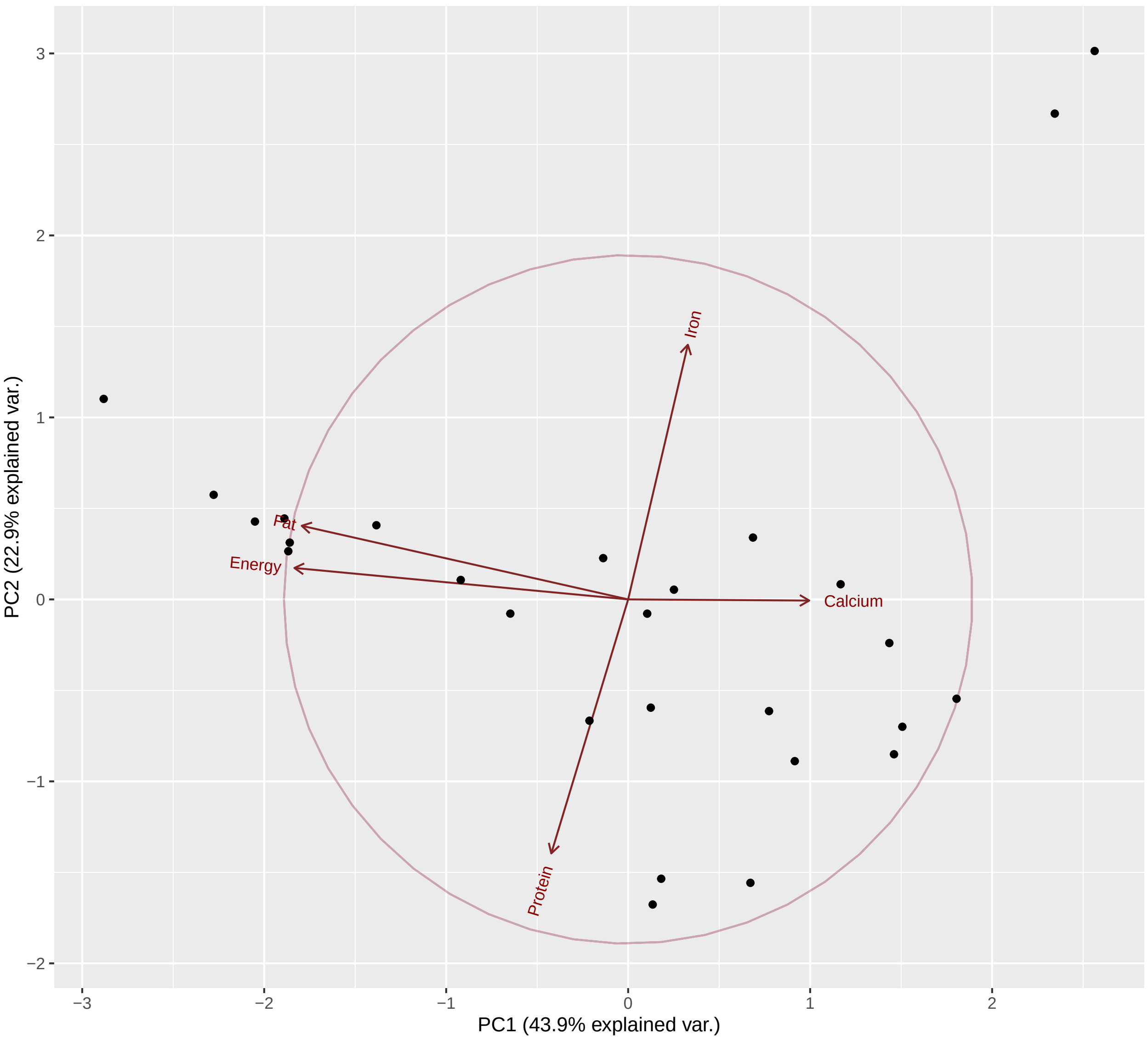
```
# The first 4 PC explained 99.97% of the variance
library(ggbiplot)
```

```
## Loading required package: plyr

## Loading required package: scales

## Loading required package: grid
```

```
# Use ggbiplot to create PCA graph
ggbiplot(pr.out, obs.scale = 1, var.scale = 1,
         ellipse = TRUE, circle = TRUE) +
  scale_color_discrete(name = '') +
  theme(legend.direction = 'horizontal', legend.position = 'top')
```



```
pr.out
```

```
## Standard deviations:
## [1] 1.48199981 1.07046241 0.92071355 0.89915054 0.04002061
```

```
##
## Rotation:
##                PC1          PC2        PC3        PC4          PC5
## Energy  -0.6542825  0.085330063 -0.1506898 -0.1970344  0.709297704
## Protein -0.1507319 -0.689332416  0.4626844 -0.5264918 -0.104068653
## Fat     -0.6399107  0.199787072 -0.2174827 -0.1317095 -0.697103521
## Calcium  0.3549785 -0.003144364 -0.6513825 -0.6705754  0.003160074
## Iron     0.1170424  0.691096838  0.5400166 -0.4657952  0.010157612
```

By PCA, we noticed that the first 4 principle components expained more than 99% of the variance in the data. And the grouping is similar to the results we obtained using other methods.

In PC1, Energy and Fat are very important. This is similar to the Hierachical clustering, which seperates high fat/energy fooe, e.g. roast beef, from other food.

In PC2, Iron and Protein are very important. This is leads to the seperation of “Clams” from other food since clams has very low Protein, but very high Iron.

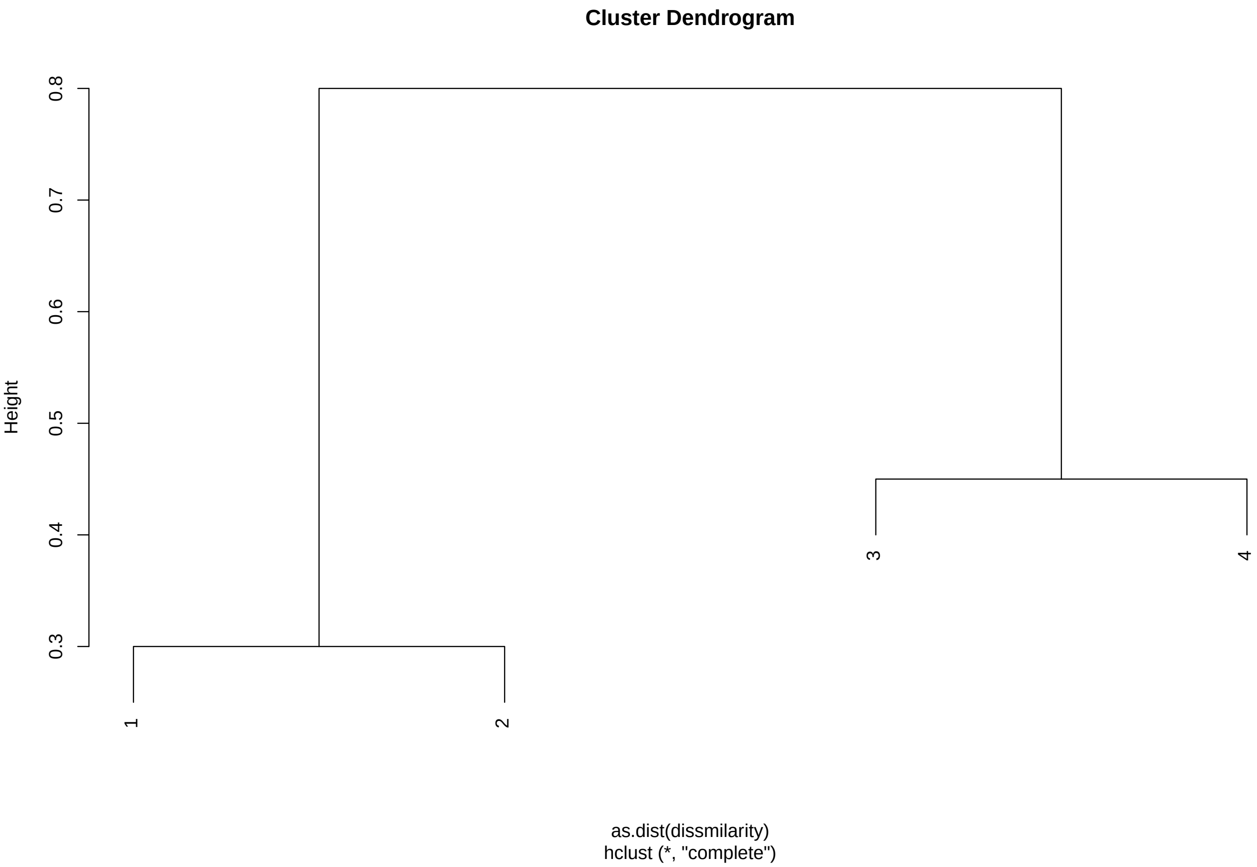
In PC3, Calcium and Iron plays a big role. This distinction seperates “Canned finshes” from other food.

In PC4, Calcium and Protein explained the rest of variability.

I prefer hierachical clustering, because K-means clustering is too general and PCA is only a measure of variance which don’t provide any clustering by itself.

Question 3

```
dissmilarity = matrix(c(0,0.3,0.4,0.7,0.3,0,0.5,0.8,0.4,0.5,0,0.45,0.7,0.8,0.45,0),nrow = 4)
clust.dist = hclust(as.dist(dissmilarity),method = "complete")
plot(clust.dist)
```



If we only want 2 cluster, then observation 1 and 2 will be in the same cluster, while observation 3 and 4 will be in another cluster.