Cereals

Abhinav Reddy

2024-04-08

Required Libraries.

```
library(cluster)
```

```
## Warning: package 'cluster' was built under R version 4.3.3
```

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(dendextend)
```

```
## Warning: package 'dendextend' was built under R version 4.3.3
```

```
##
## ---------------------
## Welcome to dendextend version 1.17.1
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## You may ask questions at stackoverflow, use the r and dendextend tags:
##   https://stackoverflow.com/questions/tagged/dendextend
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------
```

```
##
## Attaching package: 'dendextend'
```

```
## The following object is masked from 'package:stats':
##
##     cutree
```

```
library(knitr)
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 4.3.3
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(readr)
```

- Creating a data collection with only Numbers by Importing Cereals Dataset.

```
Cereals <- read.csv("C:/Users/Abhinav Reddy/Desktop/Assignment5/Cereals.csv")
View(Cereals)
df <- data.frame(Cereals[,4:16])
```

- Removing all Cereals with the missing values.

```
df <- na.omit(df)
```

- Normalizing the data using the Scale Function

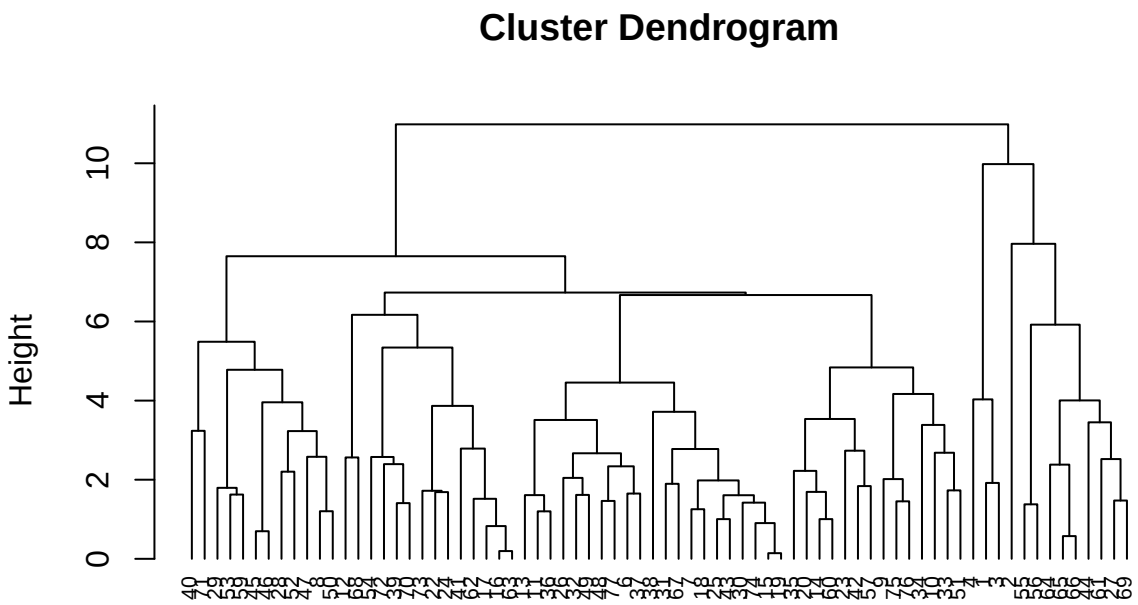
```
df_normalize <- scale(df)
```

Task 1

Apply hierarchical clustering to the data using Euclidean distance to the normalized measurements. Use Agnes to compare the clustering from single linkage, complete linkage, average linkage, and Ward. Choose the best method.

```
Distance <- dist(df_normalize, method = "euclidean")
H_cluster <- hclust(Distance, method = "complete")
```

```
#Dendogram Plot Process
plot(H_cluster, cex = 0.7, hang = -1)
```



- Perform calculations with several linkage techniques and the AGNES clustering algorithm.

```
single.Hcluster <- agnes(df_normalize, method = "single")
complete.Hcluster <- agnes(df_normalize, method = "complete")
average.Hcluster <- agnes(df_normalize, method = "average")
ward.Hcluster <- agnes(df_normalize, method = "ward")
```

- Choose the Appropriate Course of Action

```
print(single.Hcluster$ac)
```

```
## [1] 0.6067859
```

```
print(complete.Hcluster$ac)
```

```
## [1] 0.8353712
```

```
print(average.Hcluster$ac)
```

```
## [1] 0.7766075
```

```
print(ward.Hcluster$ac)
```

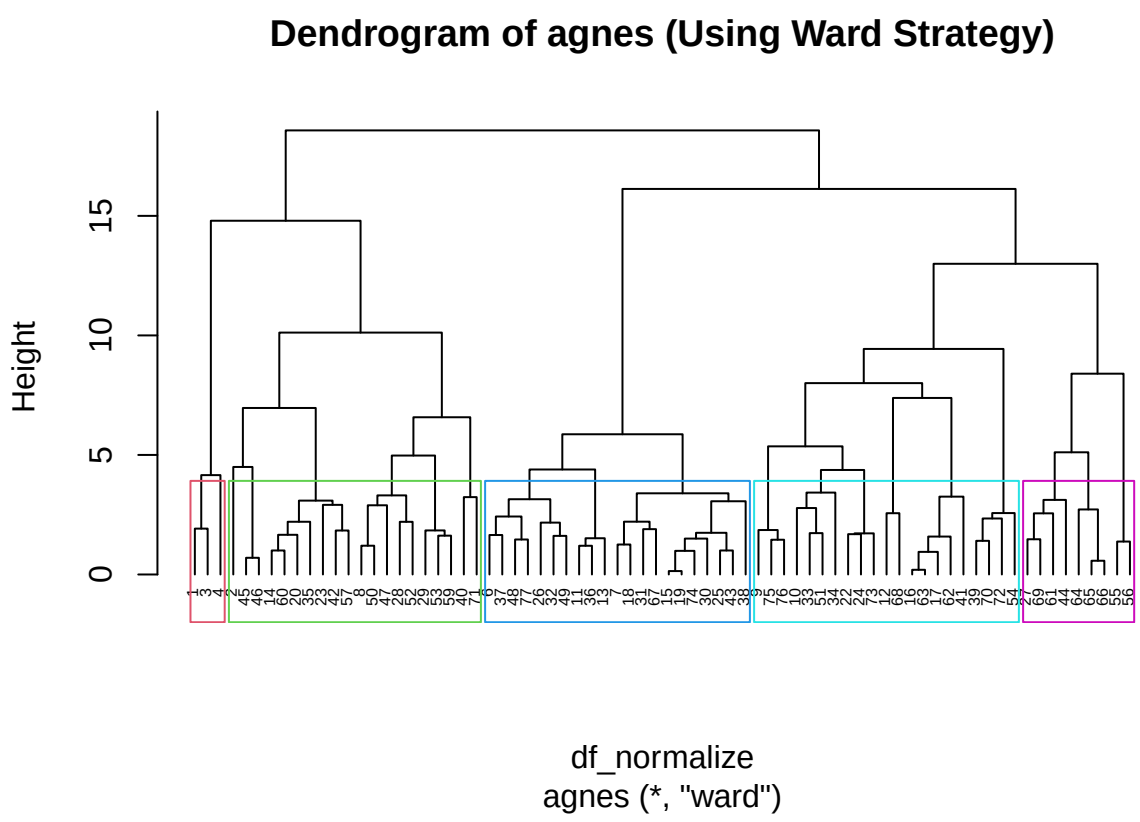
```
## [1] 0.9046042
```

- By this we can conclude that the Ward Strategy is the most appropriate and value of 0.9029485 by the facts provided upon.

Task 2

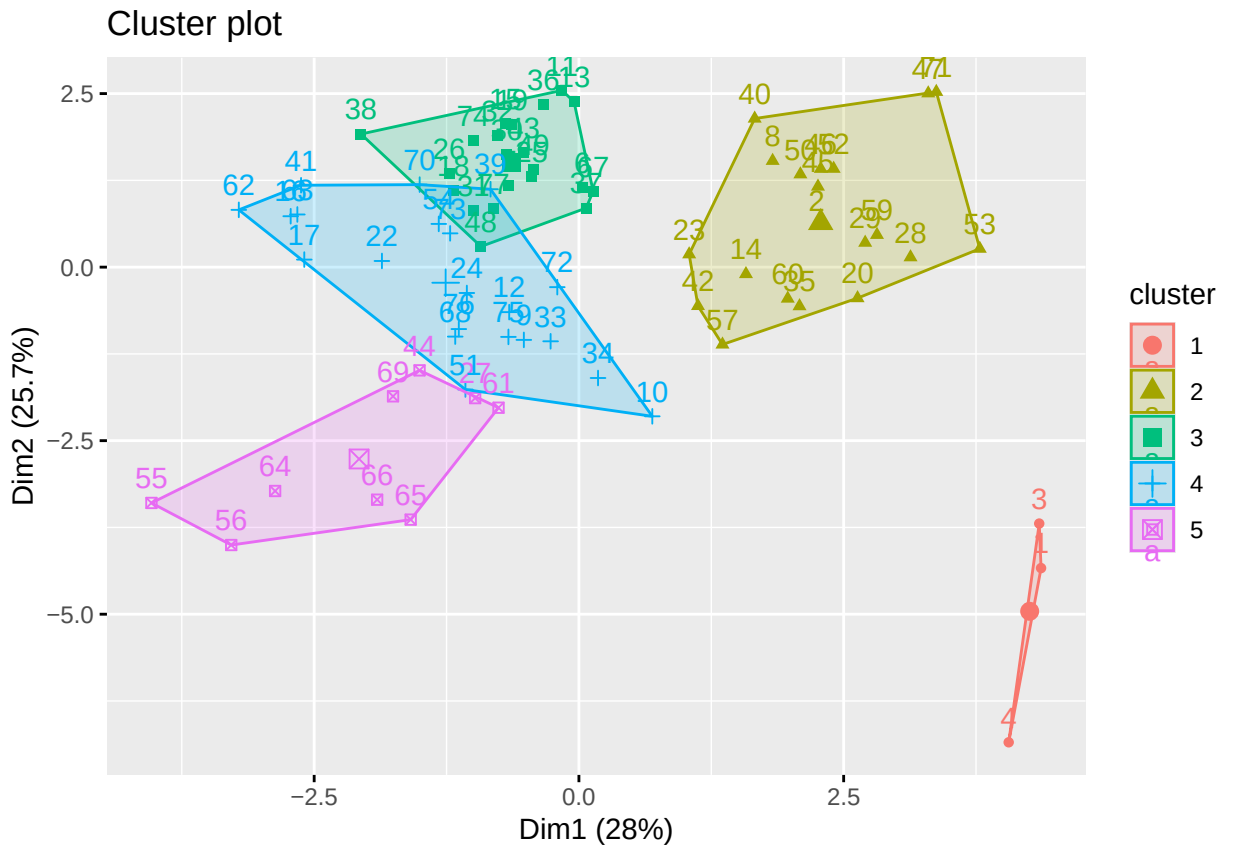
- How many clusters would you choose?

```
pltree(ward.Hcluster, cex = 0.5, hang = -1, main = "Dendrogram of agnes (Using Ward Strategy)")
rect.hclust(ward.Hcluster, k = 5, border = 2:7)
```




```
f.Group <- cutree(ward.Hcluster, k=5)
Dframe_2 <- as.data.frame(cbind(df_normalize,f.Group))
```

```
fviz_cluster(list(data = Dframe_2, cluster = f.Group))
```



By the formed clusters above seen we can conclude 5 Clusters.

Task 3

- Comment on the structure of the clusters and on their stability.

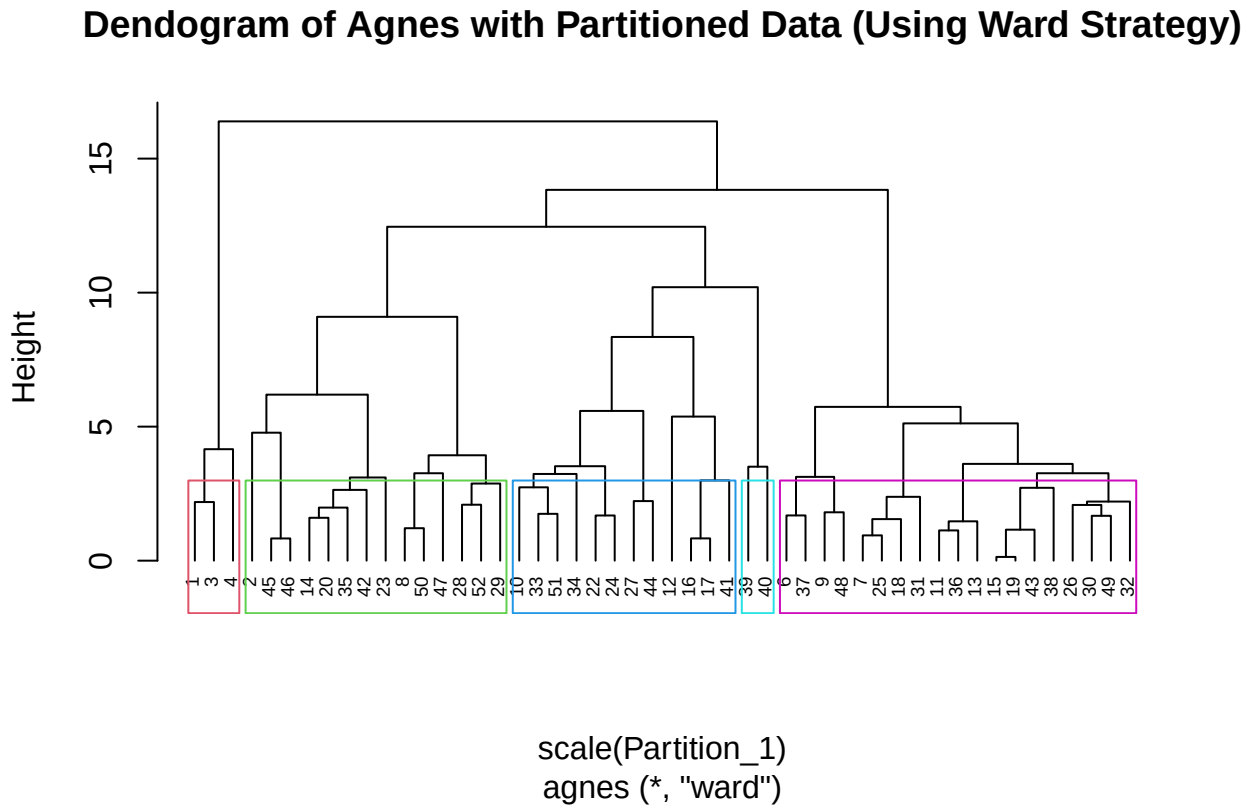
```
#Dividing the Dataframe into 2 Partitions
set.seed(123)
Partition_1 <- df[1:50,]
Partition_2 <- df[51:74,]
```

- Applying Hierarchical Clustering with k = 5 in mind. Compute for the training dataset using AGNES and various linking techniques.

```
single.dframe <- agnes(scale(Partition_1), method = "single")
complete.dframe <- agnes(scale(Partition_1), method = "complete")
average.dframe <- agnes(scale(Partition_1), method = "average")
ward.dframe <- agnes(scale(Partition_1), method = "ward")
cbind(single=single.dframe$ac , complete=complete.dframe$ac , average= average.dframe$ac , ward= ward.d
```

```
##         single  complete   average      ward
## [1,] 0.6393338 0.8138238 0.7408904 0.8764323
```

```
pltree(ward.dframe, cex = 0.6, hang = -1, main = "Dendogram of Agnes with Partitioned Data (Using Ward S
rect.hclust(ward.dframe, k = 5, border = 2:7)
```




```
cut.2 <- cutree(ward.dframe, k = 5)
```

- Centeroids to be Calculated.

```
dframe.result <- as.data.frame(cbind(Partition_1, cut.2))
dframe.result[dframe.result$cut.2==1,]
```

```
##   calories protein fat sodium fiber carbo sugars potass vitamins shelf weight
## 1       70       4   1    130    10     5      6    280       25     3      1
## 3       70       4   1    260     9     7      5    320       25     3      1
## 4       50       4   0    140    14     8      0    330       25     3      1
##   cups    rating cut.2
## 1 0.33 68.40297     1
## 3 0.33 59.42551     1
## 4 0.50 93.70491     1
```

```
centroid.1 <- colMeans(dframe.result[dframe.result$cut.2==1,])
dframe.result[dframe.result$cut.2==2,]
```

```
##    calories protein fat sodium fiber carbo sugars potass vitamins shelf weight
```

```
## 2       120       3   5     15   2.0   8.0       8    135         0     3   1.00
## 8       130       3   2    210   2.0  18.0       8    100        25     3   1.33
## 14      110       3   2    140   2.0  13.0       7    105        25     3   1.00
## 20      110       3   3    140   4.0  10.0       7    160        25     3   1.00
## 23      100       2   1    140   2.0  11.0      10    120        25     3   1.00
## 28      120       3   2    160   5.0  12.0      10    200        25     3   1.25
## 29      120       3   0    240   5.0  14.0      12    190        25     3   1.33
## 35      120       3   3     75   3.0  13.0       4    100        25     3   1.00
## 42      100       4   2    150   2.0  12.0       6     95        25     2   1.00
## 45      150       4   3     95   3.0  16.0      11    170        25     3   1.00
## 46      150       4   3    150   3.0  16.0      11    170        25     3   1.00
## 47      160       3   2    150   3.0  17.0      13    160        25     3   1.50
## 50      140       3   2    220   3.0  21.0       7    130        25     3   1.33
## 52      130       3   2    170   1.5  13.5      10    120        25     3   1.25
##    cups   rating cut.2
## 2  1.00 33.98368     2
## 8  0.75 37.03856     2
## 14 0.50 40.40021     2
## 20 0.50 40.44877     2
## 23 0.75 36.17620     2
## 28 0.67 40.91705     2
## 29 0.67 41.01549     2
## 35 0.33 45.81172     2
## 42 0.67 45.32807     2
## 45 1.00 37.13686     2
## 46 1.00 34.13976     2
## 47 0.67 30.31335     2
## 50 0.67 40.69232     2
## 52 0.50 30.45084     2
```

```
centroid.2 <- colMeans(dframe.result[dframe.result$cut.2==2,])
dframe.result[dframe.result$cut.2==3,]
```

```
##    calories protein fat sodium fiber carbo sugars potass vitamins shelf weight
## 6       110       2   2    180   1.5  10.5     10     70       25     1      1
## 7       110       2   0    125   1.0  11.0     14     30       25     2      1
## 9        90       2   1    200   4.0  15.0      6    125       25     1      1
## 11      120       1   2    220   0.0  12.0     12     35       25     2      1
## 13      120       1   3    210   0.0  13.0      9     45       25     2      1
## 15      110       1   1    180   0.0  12.0     13     55       25     2      1
## 18      110       1   0     90   1.0  13.0     12     20       25     2      1
## 19      110       1   1    180   0.0  12.0     13     65       25     2      1
## 25      110       2   1    125   1.0  11.0     13     30       25     2      1
## 26      110       1   0    200   1.0  14.0     11     25       25     1      1
## 30      110       1   1    135   0.0  13.0     12     25       25     2      1
## 31      100       2   0     45   0.0  11.0     15     40       25     1      1
## 32      110       1   1    280   0.0  15.0      9     45       25     2      1
## 36      120       1   2    220   1.0  12.0     11     45       25     2      1
## 37      110       3   1    250   1.5  11.5     10     90       25     1      1
## 38      110       1   0    180   0.0  14.0     11     35       25     1      1
## 43      110       2   1    180   0.0  12.0     12     55       25     2      1
## 48      100       2   1    220   2.0  15.0      6     90       25     1      1
## 49      120       2   1    190   0.0  15.0      9     40       25     2      1
##    cups   rating cut.2
```

```
## 6  0.75 29.50954     3
## 7  1.00 33.17409     3
## 9  0.67 49.12025     3
## 11 0.75 18.04285     3
## 13 0.75 19.82357     3
## 15 1.00 22.73645     3
## 18 1.00 35.78279     3
## 19 1.00 22.39651     3
## 25 1.00 32.20758     3
## 26 0.75 31.43597     3
## 30 0.75 28.02576     3
## 31 0.88 35.25244     3
## 32 0.75 23.80404     3
## 36 1.00 21.87129     3
## 37 0.75 31.07222     3
## 38 1.33 28.74241     3
## 43 1.00 26.73451     3
## 48 1.00 40.10596     3
## 49 0.67 29.92429     3
```

```
centroid.3 <- colMeans(dframe.result[dframe.result$cut.2==3,])
dframe.result[dframe.result$cut.2==4,]
```

```
##    calories protein fat sodium fiber carbo sugars potass vitamins shelf weight
## 10       90       3   0    210     5    13      5    190       25     3      1
## 12      110       6   2    290     2    17      1    105       25     1      1
## 16      110       2   0    280     0    22      3     25       25     1      1
## 17      100       2   0    290     1    21      2     35       25     1      1
## 22      110       2   0    220     1    21      3     30       25     3      1
## 24      100       2   0    190     1    18      5     80       25     3      1
## 27      100       3   0      0     3    14      7    100       25     2      1
## 33      100       3   1    140     3    15      5     85       25     3      1
## 34      110       3   0    170     3    17      3     90       25     3      1
## 41      110       2   1    260     0    21      3     40       25     2      1
## 44      100       4   1      0     0    16      3     95       25     2      1
## 51       90       3   0    170     3    18      2     90       25     3      1
##    cups   rating cut.2
## 10 0.67 53.31381     4
## 12 1.25 50.76500     4
## 16 1.00 41.44502     4
## 17 1.00 45.86332     4
## 22 1.00 46.89564     4
## 24 0.75 44.33086     4
## 27 0.80 58.34514     4
## 33 0.88 52.07690     4
## 34 0.25 53.37101     4
## 41 1.50 39.24111     4
## 44 1.00 54.85092     4
## 51 1.00 59.64284     4
```

```
centroid.4 <- colMeans(dframe.result[dframe.result$cut.2==4,])
CentRoids <- rbind(centroid.1, centroid.2, centroid.3, centroid.4)
x2 <- as.data.frame(rbind(CentRoids[,-14], Partition_2))
```

- Distance between the Centeroids to be calculated

```
Distance.1 <- get_dist(x2)
matrix.1 <- as.matrix(Distance.1)
Dataframe.1 <- data.frame(data=seq(1,nrow(Partition_2),1), Clusters = rep(0,nrow(Partition_2)))
for(i in 1:nrow(Partition_2))
  {Dataframe.1[i,2] <- which.min(matrix.1[i+4, 1:4])}
Dataframe.1
```

```
##    data Clusters
## 1     1        1
## 2     2        4
## 3     3        3
## 4     4        2
## 5     5        2
## 6     6        1
## 7     7        2
## 8     8        2
## 9     9        3
## 10   10        3
## 11   11        2
## 12   12        2
## 13   13        2
## 14   14        3
## 15   15        4
## 16   16        2
## 17   17        3
## 18   18        2
## 19   19        4
## 20   20        4
## 21   21        3
## 22   22        4
## 23   23        4
## 24   24        3
```

```
cbind(Dframe_2$f.Group[51:74], Dataframe.1$Clusters)
```

```
##       [,1] [,2]
##  [1,]    2    1
##  [2,]    4    4
##  [3,]    5    3
##  [4,]    5    2
##  [5,]    2    2
##  [6,]    2    1
##  [7,]    2    2
##  [8,]    5    2
##  [9,]    4    3
## [10,]    4    3
## [11,]    5    2
## [12,]    5    2
## [13,]    5    2
## [14,]    3    3
## [15,]    4    4
```

```
## [16,]    5    2
## [17,]    4    3
## [18,]    2    2
## [19,]    4    4
## [20,]    4    4
## [21,]    3    3
## [22,]    4    4
## [23,]    4    4
## [24,]    3    3
```

```
table(Dframe_2$f.Group[51:74] == Dataframe.1$Clusters)
```

```
##
## FALSE  TRUE
##    12    12
```

- By the observations mentioned above we conclude that 12 are True and 12 are False, so we can tell that the model is Evenly distributed and Partially Unstabled.

Task 4

- The elementary public schools would like to choose a set of cereals to include in their daily cafeterias. Every day a different cereal is offered, but all cereals should support a healthy diet. For this goal, you are requested to find a cluster of "healthy cereals." Should the data be normalized? If not, how should they be used in the cluster analysis?

```
#Clustering the data into clusters with a aim of Healthy Cereal cluster
healthy.data <- df
healthy.data_RD <- na.omit(healthy.data)
clust <- cbind(healthy.data_RD, f.Group)
clust[clust$f.Group==1,]
```

```
##   calories protein fat sodium fiber carbo sugars potass vitamins shelf weight
## 1       70       4   1    130    10     5      6    280       25     3      1
## 3       70       4   1    260     9     7      5    320       25     3      1
## 4       50       4   0    140    14     8      0    330       25     3      1
##   cups    rating f.Group
## 1 0.33 68.40297       1
## 3 0.33 59.42551       1
## 4 0.50 93.70491       1
```

```
clust[clust$f.Group==2,]
```

```
##    calories protein fat sodium fiber carbo sugars potass vitamins shelf weight
## 2       120       3   5     15   2.0   8.0      8    135        0     3   1.00
## 8       130       3   2    210   2.0  18.0      8    100       25     3   1.33
## 14      110       3   2    140   2.0  13.0      7    105       25     3   1.00
## 20      110       3   3    140   4.0  10.0      7    160       25     3   1.00
```

```
## 23      100       2   1    140   2.0  11.0     10    120       25     3   1.00
## 28      120       3   2    160   5.0  12.0     10    200       25     3   1.25
## 29      120       3   0    240   5.0  14.0     12    190       25     3   1.33
## 35      120       3   3     75   3.0  13.0      4    100       25     3   1.00
## 40      140       3   1    170   2.0  20.0      9     95      100     3   1.30
## 42      100       4   2    150   2.0  12.0      6     95       25     2   1.00
## 45      150       4   3     95   3.0  16.0     11    170       25     3   1.00
## 46      150       4   3    150   3.0  16.0     11    170       25     3   1.00
## 47      160       3   2    150   3.0  17.0     13    160       25     3   1.50
## 50      140       3   2    220   3.0  21.0      7    130       25     3   1.33
## 52      130       3   2    170   1.5  13.5     10    120       25     3   1.25
## 53      120       3   1    200   6.0  11.0     14    260       25     3   1.33
## 57      100       4   1    135   2.0  14.0      6    110       25     3   1.00
## 59      120       3   1    210   5.0  14.0     12    240       25     2   1.33
## 60      100       3   2    140   2.5  10.5      8    140       25     3   1.00
## 71      140       3   1    190   4.0  15.0     14    230      100     3   1.50
##    cups   rating f.Group
## 2  1.00 33.98368       2
## 8  0.75 37.03856       2
## 14 0.50 40.40021       2
## 20 0.50 40.44877       2
## 23 0.75 36.17620       2
## 28 0.67 40.91705       2
## 29 0.67 41.01549       2
## 35 0.33 45.81172       2
## 40 0.75 36.47151       2
## 42 0.67 45.32807       2
## 45 1.00 37.13686       2
## 46 1.00 34.13976       2
## 47 0.67 30.31335       2
## 50 0.67 40.69232       2
## 52 0.50 30.45084       2
## 53 0.67 37.84059       2
## 57 0.50 49.51187       2
## 59 0.75 39.25920       2
## 60 0.50 39.70340       2
## 71 1.00 28.59278       2
```

```
clust[clust$f.Group==3,]
```

```
##    calories protein fat sodium fiber carbo sugars potass vitamins shelf weight
## 6       110       2   2    180   1.5  10.5     10     70       25     1      1
## 7       110       2   0    125   1.0  11.0     14     30       25     2      1
## 11      120       1   2    220   0.0  12.0     12     35       25     2      1
## 13      120       1   3    210   0.0  13.0      9     45       25     2      1
## 15      110       1   1    180   0.0  12.0     13     55       25     2      1
## 18      110       1   0     90   1.0  13.0     12     20       25     2      1
## 19      110       1   1    180   0.0  12.0     13     65       25     2      1
## 25      110       2   1    125   1.0  11.0     13     30       25     2      1
## 26      110       1   0    200   1.0  14.0     11     25       25     1      1
## 30      110       1   1    135   0.0  13.0     12     25       25     2      1
## 31      100       2   0     45   0.0  11.0     15     40       25     1      1
## 32      110       1   1    280   0.0  15.0      9     45       25     2      1
## 36      120       1   2    220   1.0  12.0     11     45       25     2      1
```

```
## 37      110       3   1    250   1.5  11.5     10     90       25     1       1
## 38      110       1   0    180   0.0  14.0     11     35       25     1       1
## 43      110       2   1    180   0.0  12.0     12     55       25     2       1
## 48      100       2   1    220   2.0  15.0      6     90       25     1       1
## 49      120       2   1    190   0.0  15.0      9     40       25     2       1
## 67      110       2   1     70   1.0   9.0     15     40       25     2       1
## 74      110       1   1    140   0.0  13.0     12     25       25     2       1
## 77      110       2   1    200   1.0  16.0      8     60       25     1       1
##    cups   rating f.Group
## 6  0.75 29.50954       3
## 7  1.00 33.17409       3
## 11 0.75 18.04285       3
## 13 0.75 19.82357       3
## 15 1.00 22.73645       3
## 18 1.00 35.78279       3
## 19 1.00 22.39651       3
## 25 1.00 32.20758       3
## 26 0.75 31.43597       3
## 30 0.75 28.02576       3
## 31 0.88 35.25244       3
## 32 0.75 23.80404       3
## 36 1.00 21.87129       3
## 37 0.75 31.07222       3
## 38 1.33 28.74241       3
## 43 1.00 26.73451       3
## 48 1.00 40.10596       3
## 49 0.67 29.92429       3
## 67 0.75 31.23005       3
## 74 1.00 27.75330       3
## 77 0.75 36.18756       3
```

```
clust[clust$f.Group==4,]
```

```
##    calories protein fat sodium fiber carbo sugars potass vitamins shelf weight
## 9        90       2   1    200     4    15       6    125       25     1       1
## 10       90       3   0    210     5    13       5    190       25     3       1
## 12      110       6   2    290     2    17       1    105       25     1       1
## 16      110       2   0    280     0    22       3     25       25     1       1
## 17      100       2   0    290     1    21       2     35       25     1       1
## 22      110       2   0    220     1    21       3     30       25     3       1
## 24      100       2   0    190     1    18       5     80       25     3       1
## 33      100       3   1    140     3    15       5     85       25     3       1
## 34      110       3   0    170     3    17       3     90       25     3       1
## 39      110       2   1    170     1    17       6     60      100     3       1
## 41      110       2   1    260     0    21       3     40       25     2       1
## 51       90       3   0    170     3    18       2     90       25     3       1
## 54      100       3   0    320     1    20       3     45      100     3       1
## 62      110       1   0    240     0    23       2     30       25     1       1
## 63      110       2   0    290     0    22       3     35       25     1       1
## 68      110       6   0    230     1    16       3     55       25     1       1
## 70      110       2   1    200     0    21       3     35      100     3       1
## 72      100       3   1    200     3    16       3    110      100     3       1
## 73      110       2   1    250     0    21       3     60       25     3       1
## 75      100       3   1    230     3    17       3    115       25     1       1
```

```
## 76      100       3   1    200     3    17      3    110       25     1      1
##    cups   rating f.Group
## 9  0.67 49.12025       4
## 10 0.67 53.31381       4
## 12 1.25 50.76500       4
## 16 1.00 41.44502       4
## 17 1.00 45.86332       4
## 22 1.00 46.89564       4
## 24 0.75 44.33086       4
## 33 0.88 52.07690       4
## 34 0.25 53.37101       4
## 39 1.00 36.52368       4
## 41 1.50 39.24111       4
## 51 1.00 59.64284       4
## 54 1.00 41.50354       4
## 62 1.13 41.99893       4
## 63 1.00 40.56016       4
## 68 1.00 53.13132       4
## 70 1.00 38.83975       4
## 72 1.00 46.65884       4
## 73 0.75 39.10617       4
## 75 0.67 49.78744       4
## 76 1.00 51.59219       4
```

- Mean is used for the analysis to make out the Best Cluster

```
mean(clust[clust$f.Group==1,"rating"])
```

```
## [1] 73.84446
```

```
mean(clust[clust$f.Group==2,"rating"])
```

```
## [1] 38.26161
```

```
mean(clust[clust$f.Group==3,"rating"])
```

```
## [1] 28.84825
```

```
mean(clust[clust$f.Group==4,"rating"])
```

```
## [1] 46.46513
```

- By the mean of all the clusters among the four, Cluster 1 has the most value which is 73.84446 while comparing with the remaining clusters. By this we can confirm that the Group 1 will be the most healthiest diet and the cluster of “Healthy cereals” for the Elementary Public Schools.

Thank You !!!