

# Assignment\_5

Hanish Bhogadi

4/17/2022

```
#Importing required libraries
```

```
library(cluster)
library(caret)
```

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

```
## Warning in register(): Can't find generic 'scale_type' in package ggplot2 to
## register S3 method.
```

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

```
library(dendextend)
```

```
## Warning: package 'dendextend' was built under R version 4.1.3
```

```
##
## -----
## Welcome to dendextend version 1.15.2
## 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 4.1.3
```

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

```
#Importing dataset
```

```
Cereals<- read.csv("C:/Machine Learning - 1/64060_hbhogadi/Assignment_5/Cereals.csv")
```

```
Data_cereals <- data.frame(Cereals[,4:16])
```

```
#Preprocessing the data
```

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

```
#Data Normalization
```

```
Data_cereals_normalized <- scale(Data_cereals)
```

```
#Applying hierarchical clustering to the data using Euclidean distance to the normalize measurements.
```

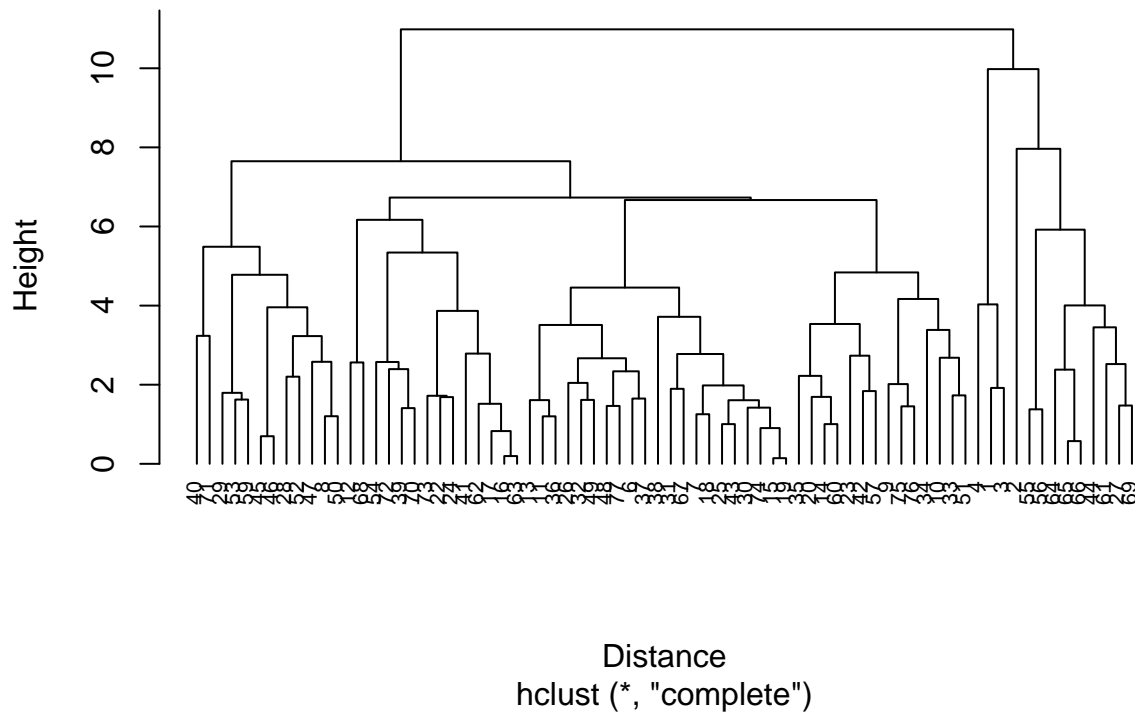
```
Distance <- dist(Data_cereals_normalized, method = "euclidean")
```

```
hierarchial.clust_complete <- hclust(Distance, method = "complete")
```

```
#Plotting the dendogram
```

```
plot(hierarchial.clust_complete, cex = 0.7, hang = -1)
```

## Cluster Dendrogram



*#Using agnes function to perform clustering with single linkage,  
#complete linkage, average linkage and Ward.*

```
hierarchial.clust_single <- agnes(Data_cereals_normalized, method = "single")
hierarchial.clust_complete <- agnes(Data_cereals_normalized, method = "complete")
hierarchial.clust_average <- agnes(Data_cereals_normalized, method = "average")
hierarchial.clust_ward <- agnes(Data_cereals_normalized, method = "ward")
```

*#Single Linkage vs Complete Linkage vs Average Linkage vs Ward*

```
print(hierarchial.clust_single$ac)
```

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

```
print(hierarchial.clust_complete$ac)
```

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

```
print(hierarchial.clust_average$ac)
```

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

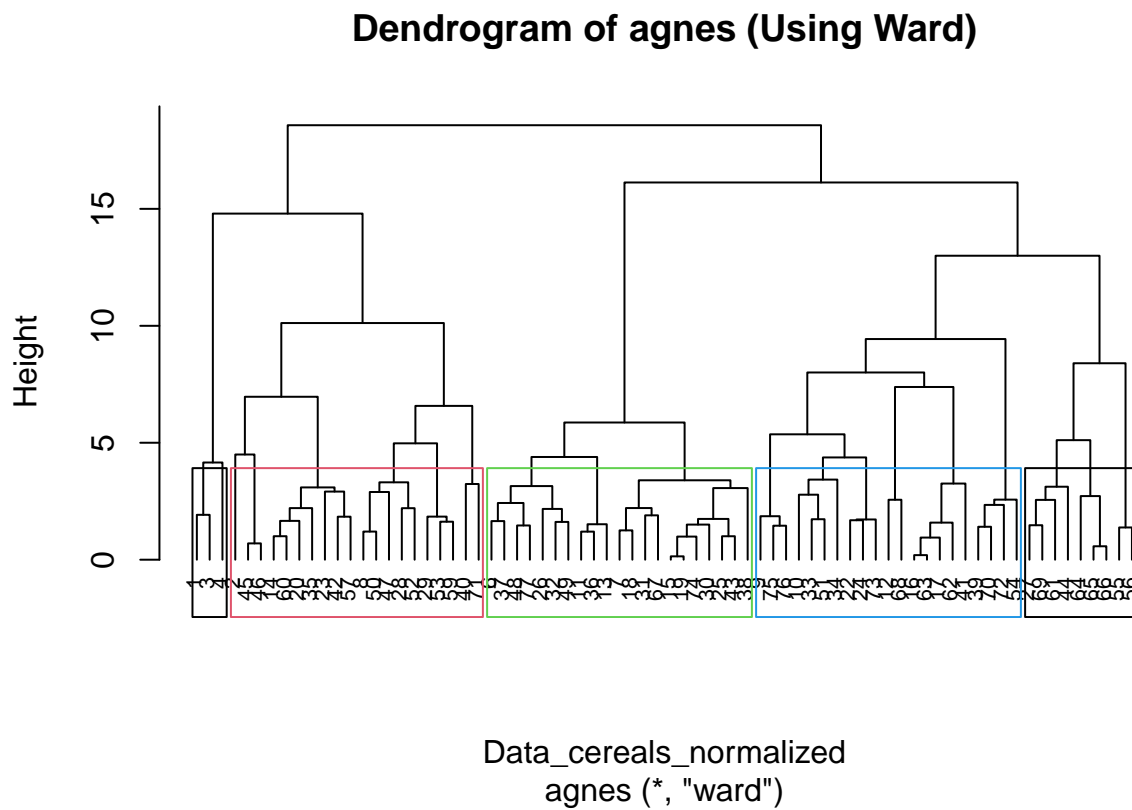
```
print(hierarchial.clust_ward$ac)
```

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

*#Since WARD method has the highest value of 0.9046042, we will consider it.*

*#(2) Choosing the clusters:*

```
pltree(hierarchial.clust_ward, cex = 0.7, hang = -1, main =  
      "Dendrogram of agnes (Using Ward)")  
rect.hclust(hierarchial.clust_ward, k = 5, border = 1:4)
```



```
Cluster1 <- cutree(hierarchial.clust_ward, k=5)
```

```
dataframe2 <- as.data.frame(cbind(Data_cereals_normalized, Cluster1))
```

*#We will choose 5 clusters after observing the distance.*

*#Commenting on the structure of the clusters and on their stability*

```
#Creating Partitions
```

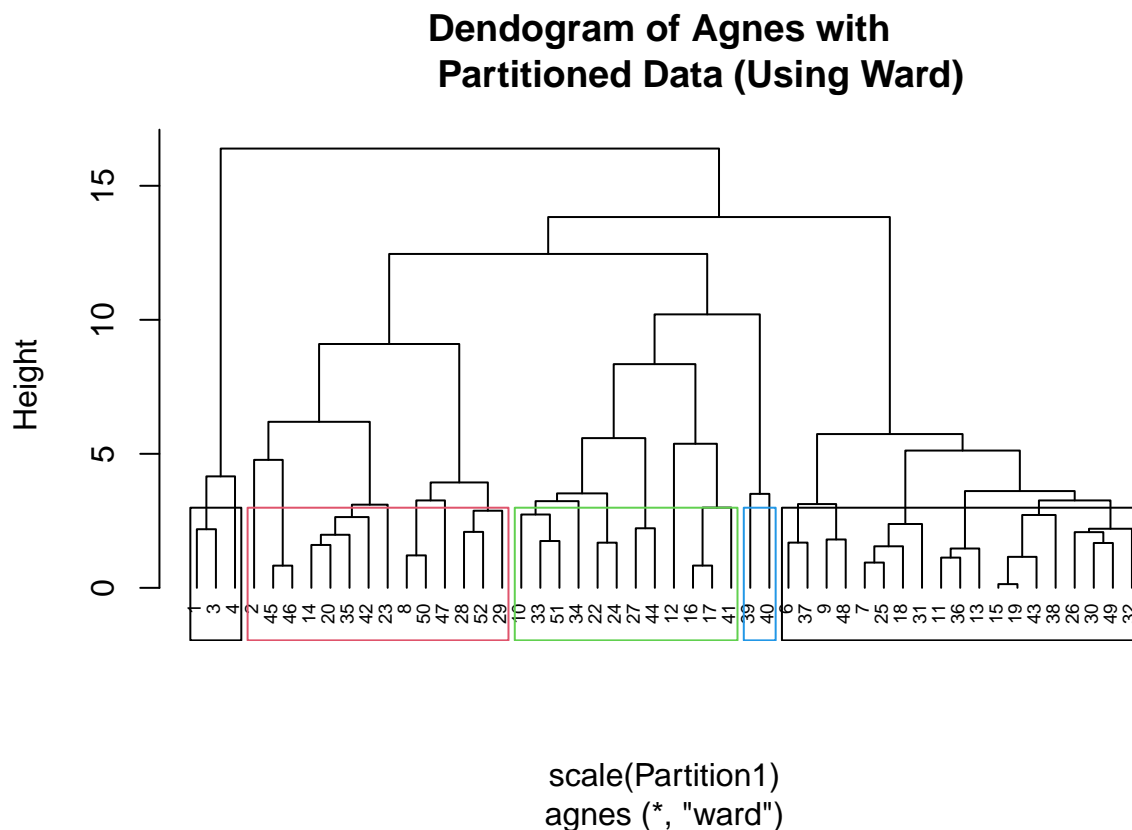
```
set.seed(123)
Partition1 <- Data_cereals[1:50,]
Partition2 <- Data_cereals[51:74,]
```

```
#Performing Hierarchical Clustering, considering k = 5.
```

```
AG_single <- agnes(scale(Partition1), method = "single")
AG_complete <- agnes(scale(Partition1), method = "complete")
AG_average <- agnes(scale(Partition1), method = "average")
AG_ward <- agnes(scale(Partition1), method = "ward")
cbind(single=AG_single$ac , complete=AG_complete$ac , average= AG_average$ac ,
      ward= AG_ward$ac)
```

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

```
pltree(AG_ward, cex = 0.6, hang = -1, main = "Dendrogram of Agnes with
      Partitioned Data (Using Ward)")
rect.hclust(AG_ward, k = 5, border = 1:4)
```



```
cut_2 <- cutree(AG_ward, k = 5)
```

```
#Calculating the centroids.
```

```
result <- as.data.frame(cbind(Partition1, cut_2))
result[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(result[result$cut_2==1,])
result[result$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(result[result$cut_2==2,])
result[result$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(result[result$cut_2==3,])
result[result$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(result[result$cut_2==4,])
centroids <- rbind(centroid_1, centroid_2, centroid_3, centroid_4)
x2 <- as.data.frame(rbind(centroids[, -14], Partition2))
```

*#Calculating the Distance*

```
Distance_1 <- get_dist(x2)
Matrix_1 <- as.matrix(Distance_1)
dataframe1 <- data.frame(data=seq(1,nrow(Partition2),1), Clusters =
                        rep(0,nrow(Partition2)))
for(i in 1:nrow(Partition2))
{dataframe1[i,2] <- which.min(Matrix_1[i+4, 1:4])}
dataframe1
```

```
##      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(dataframe2$Cluster1[51:74], dataframe1$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(dataframe2$Cluster1[51:74] == dataframe1$Clusters)
```

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

*#We can say that the model is partially stable as we are getting 12 FALSE and 12 TRUE*

*#3) 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."*

*#Clustering Healthy Cereals.*

```

Healthy_Cereals <- Cereals
Healthy_Cereals_new <- na.omit(Healthy_Cereals)
HealthyClust <- cbind(Healthy_Cereals_new, Cluster1)
HealthyClust[HealthyClust$Cluster1==1,]

```

```

##              name mfr type calories protein fat sodium fiber carbo
## 1          100%_Bran   N   C       70        4   1   130    10     5
## 3              All-Bran   K   C       70        4   1   260     9     7
## 4 All-Bran_with_Extra_Fiber   K   C       50        4   0   140    14     8
##   sugars potass vitamins shelf weight cups   rating Cluster1
## 1      6     280       25    3      1 0.33 68.40297         1
## 3      5     320       25    3      1 0.33 59.42551         1
## 4      0     330       25    3      1 0.50 93.70491         1

```

```

HealthyClust[HealthyClust$Cluster1==2,]

```

```

##              name mfr type calories protein fat sodium
## 2          100%_Natural_Bran   Q   C      120        3   5    15
## 8              Basic_4       G   C      130        3   2   210
## 14             Clusters       G   C      110        3   2   140
## 20      Cracklin'_Oat_Bran   K   C      110        3   3   140
## 23      Crispy_Wheat_&_Raisins   G   C      100        2   1   140
## 28 Fruit_&_Fibre_Dates,_Walnuts,_and_Oats   P   C      120        3   2   160
## 29             Fruitful_Bran   K   C      120        3   0   240
## 35      Great_Grains_Pecan   P   C      120        3   3    75
## 40      Just_Right_Fruit_&_Nut   K   C      140        3   1   170
## 42              Life       Q   C      100        4   2   150
## 45      Muesli_Raisins,_Dates,_&_Almonds   R   C      150        4   3    95
## 46      Muesli_Raisins,_Peaches,_&_Pecans   R   C      150        4   3   150
## 47      Mueslix_Crispy_Blend   K   C      160        3   2   150
## 50      Nutri-Grain_Almond-Raisin   K   C      140        3   2   220
## 52      Oatmeal_Raisin_Crisp   G   C      130        3   2   170
## 53      Post_Nat._Raisin_Bran   P   C      120        3   1   200
## 57      Quaker_Oat_Squares   Q   C      100        4   1   135
## 59      Raisin_Bran   K   C      120        3   1   210
## 60      Raisin_Nut_Bran   G   C      100        3   2   140
## 71      Total_Raisin_Bran   G   C      140        3   1   190
##   fiber carbo sugars potass vitamins shelf weight cups   rating Cluster1
## 2    2.0   8.0      8    135        0    3    1.00 1.00 33.98368         2
## 8    2.0  18.0      8    100       25    3    1.33 0.75 37.03856         2
## 14   2.0  13.0      7    105       25    3    1.00 0.50 40.40021         2
## 20   4.0  10.0      7    160       25    3    1.00 0.50 40.44877         2
## 23   2.0  11.0     10    120       25    3    1.00 0.75 36.17620         2
## 28   5.0  12.0     10    200       25    3    1.25 0.67 40.91705         2
## 29   5.0  14.0     12    190       25    3    1.33 0.67 41.01549         2
## 35   3.0  13.0      4    100       25    3    1.00 0.33 45.81172         2
## 40   2.0  20.0      9     95      100    3    1.30 0.75 36.47151         2
## 42   2.0  12.0      6     95       25    2    1.00 0.67 45.32807         2
## 45   3.0  16.0     11    170       25    3    1.00 1.00 37.13686         2
## 46   3.0  16.0     11    170       25    3    1.00 1.00 34.13976         2
## 47   3.0  17.0     13    160       25    3    1.50 0.67 30.31335         2
## 50   3.0  21.0      7    130       25    3    1.33 0.67 40.69232         2

```

```
## 52  1.5  13.5    10   120    25    3   1.25 0.50 30.45084    2
## 53  6.0  11.0    14   260    25    3   1.33 0.67 37.84059    2
## 57  2.0  14.0     6   110    25    3   1.00 0.50 49.51187    2
## 59  5.0  14.0    12   240    25    2   1.33 0.75 39.25920    2
## 60  2.5  10.5     8   140    25    3   1.00 0.50 39.70340    2
## 71  4.0  15.0    14   230   100    3   1.50 1.00 28.59278    2
```

```
HealthyClust[HealthyClust$Cluster1==3,]
```

```
##              name mfr type calories protein fat sodium fiber carbo
## 6  Apple_Cinnamon_Cheerios G C    110      2  2   180   1.5  10.5
## 7              Apple_Jacks K C    110      2  0   125   1.0  11.0
## 11             Cap'n'Crunch Q C    120      1  2   220   0.0  12.0
## 13  Cinnamon_Toast_Crunch G C    120      1  3   210   0.0  13.0
## 15             Cocoa_Puffs G C    110      1  1   180   0.0  12.0
## 18             Corn_Pops K C    110      1  0    90   1.0  13.0
## 19             Count_Chocula G C    110      1  1   180   0.0  12.0
## 25             Froot_Loops K C    110      2  1   125   1.0  11.0
## 26             Frosted_Flakes K C    110      1  0   200   1.0  14.0
## 30             Fruity_Pebbles P C    110      1  1   135   0.0  13.0
## 31             Golden_Crisp P C    100      2  0    45   0.0  11.0
## 32             Golden_Grahams G C    110      1  1   280   0.0  15.0
## 36             Honey_Graham_Ohs Q C    120      1  2   220   1.0  12.0
## 37             Honey_Nut_Cheerios G C    110      3  1   250   1.5  11.5
## 38             Honey-comb P C    110      1  0   180   0.0  14.0
## 43             Lucky_Charms G C    110      2  1   180   0.0  12.0
## 48  Multi-Grain_Cheerios G C    100      2  1   220   2.0  15.0
## 49             Nut&Honey_Crunch K C    120      2  1   190   0.0  15.0
## 67             Smacks K C    110      2  1    70   1.0   9.0
## 74             Trix G C    110      1  1   140   0.0  13.0
## 77  Wheaties_Honey_Gold G C    110      2  1   200   1.0  16.0
##      sugars potass vitamins shelf weight cups rating Cluster1
## 6      10      70      25      1      1 0.75 29.50954      3
## 7      14      30      25      2      1 1.00 33.17409      3
## 11     12      35      25      2      1 0.75 18.04285      3
## 13      9      45      25      2      1 0.75 19.82357      3
## 15     13      55      25      2      1 1.00 22.73645      3
## 18     12      20      25      2      1 1.00 35.78279      3
## 19     13      65      25      2      1 1.00 22.39651      3
## 25     13      30      25      2      1 1.00 32.20758      3
## 26     11      25      25      1      1 0.75 31.43597      3
## 30     12      25      25      2      1 0.75 28.02576      3
## 31     15      40      25      1      1 0.88 35.25244      3
## 32      9      45      25      2      1 0.75 23.80404      3
## 36     11      45      25      2      1 1.00 21.87129      3
## 37     10      90      25      1      1 0.75 31.07222      3
## 38     11      35      25      1      1 1.33 28.74241      3
## 43     12      55      25      2      1 1.00 26.73451      3
## 48      6      90      25      1      1 1.00 40.10596      3
## 49      9      40      25      2      1 0.67 29.92429      3
## 67     15      40      25      2      1 0.75 31.23005      3
## 74     12      25      25      2      1 1.00 27.75330      3
## 77      8      60      25      1      1 0.75 36.18756      3
```

```
HealthyClust[HealthyClust$Cluster1==4,]
```

```
##               name mfr type calories protein fat sodium fiber carbo
## 9           Bran_Chex R   C      90        2  1   200     4    15
## 10          Bran_Flakes P   C      90        3  0   210     5    13
## 12           Cheerios G   C     110        6  2   290     2    17
## 16           Corn_Chex R   C     110        2  0   280     0    22
## 17          Corn_Flakes K   C     100        2  0   290     1    21
## 22           Crispix K   C     110        2  0   220     1    21
## 24          Double_Chex R   C     100        2  0   190     1    18
## 33          Grape_Nuts_Flakes P   C     100        3  1   140     3    15
## 34           Grape-Nuts P   C     110        3  0   170     3    17
## 39 Just_Right_Crunchy__Nuggets K   C     110        2  1   170     1    17
## 41                Kix G   C     110        2  1   260     0    21
## 51          Nutri-grain_Wheat K   C      90        3  0   170     3    18
## 54          Product_19 K   C     100        3  0   320     1    20
## 62           Rice_Chex R   C     110        1  0   240     0    23
## 63          Rice_Krispies K   C     110        2  0   290     0    22
## 68           Special_K K   C     110        6  0   230     1    16
## 70          Total_Corn_Flakes G   C     110        2  1   200     0    21
## 72          Total_Whole_Grain G   C     100        3  1   200     3    16
## 73           Triples G   C     110        2  1   250     0    21
## 75           Wheat_Chex R   C     100        3  1   230     3    17
## 76           Wheaties G   C     100        3  1   200     3    17
```

```
##      sugars potass vitamins shelf weight cups   rating Cluster1
## 9         6    125      25     1      1 0.67 49.12025         4
## 10        5    190      25     3      1 0.67 53.31381         4
## 12        1    105      25     1      1 1.25 50.76500         4
## 16        3     25      25     1      1 1.00 41.44502         4
## 17        2     35      25     1      1 1.00 45.86332         4
## 22        3     30      25     3      1 1.00 46.89564         4
## 24        5     80      25     3      1 0.75 44.33086         4
## 33        5     85      25     3      1 0.88 52.07690         4
## 34        3     90      25     3      1 0.25 53.37101         4
## 39        6     60     100     3      1 1.00 36.52368         4
## 41        3     40      25     2      1 1.50 39.24111         4
## 51        2     90      25     3      1 1.00 59.64284         4
## 54        3     45     100     3      1 1.00 41.50354         4
## 62        2     30      25     1      1 1.13 41.99893         4
## 63        3     35      25     1      1 1.00 40.56016         4
## 68        3     55      25     1      1 1.00 53.13132         4
## 70        3     35     100     3      1 1.00 38.83975         4
## 72        3    110     100     3      1 1.00 46.65884         4
## 73        3     60      25     3      1 0.75 39.10617         4
## 75        3    115      25     1      1 0.67 49.78744         4
## 76        3    110      25     1      1 1.00 51.59219         4
```

```
#Mean ratings to determine the best cluster.
```

```
mean(HealthyClust[HealthyClust$Cluster1==1,"rating"])
```

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

```
mean(HealthyClust[HealthyClust$Cluster1==2,"rating"])
```

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

```
mean(HealthyClust[HealthyClust$Cluster1==3,"rating"])
```

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

```
mean(HealthyClust[HealthyClust$Cluster1==4,"rating"])
```

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

```
#We can consider cluster 1 since mean ratings of the cluster1 is the  
#highest (i.e. 73.84446).
```