Assignment 5

Jayasri

11/29/2021

Hierarchical Clustering:

The purpose of this assignment is to use Hierarchical Clustering.

The dataset Cereals.csv includes nutritional information, store display, and consumer ratings for 77 breakfast cereals. This dataset consists of 77 observations and 16 features The variables in the dataset are:

- **Name** : Name of the Cereal
- **Mfr** : Manufacturer of the Cereal, which has different classifications

```
1. A = American HomeFood Products

2. G = General Mills

3. K = Kelloggs

4. N = Nabisco

5. P = Post

6. Q = Quaker Oats

7. R = Ralston Purina
```

- **Type** : This is categorized as two types 1. Hot 2. Cold
- **calories**: Number of calories per serving
- **Protein** : grams of protein
- **Fat** : grams of fat
- **Sodium** : milligrams of sodium
- **Fiber** : grams of dietary fiber
- **Carbo** : grams of carbohydrates
- **Sugars** : grams of sugars
- **Potass** : milligrams of potassium
- **Vitamins:** Vitamins and minerals and it is classified by the typical percentage of FDA recommended (0,25,100)

- **Shelf** : display shelf (1, 2 or 3, counting from floor)
- **Weight** : weight in ounces of one serving
- **Cups** : number of cups in one serving
- **Rating** : rating of the cereals(from consumer reports)

Loading the required libraries

```
library(factoextra)
library(cowplot)
library(caret)
library(knitr)
library(dummies)
library(psych)
library(cluster)
library(dplyr)
library(tibble)
library(tidyverse)
library(tidyr)
library(fpc)

options(knitr.duplicate.label = "allow")
```

Data Exploration

```
## [1] 77 16
```

The cereal data has 77 observations and 16 features.

Data Preparation

```
# Checking for NA Values
colMeans(is.na(cereal_data))
```

```
##       name        mfr       type   calories    protein        fat     sodium
## 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
##      fiber      carbo     sugars     potass   vitamins      shelf     weight
## 0.00000000 0.01298701 0.01298701 0.02597403 0.00000000 0.00000000 0.00000000
##       cups     rating
## 0.00000000 0.00000000
```

We can see that carbo, sugars and potass variables have missing values in the dataset. So we have removed the NA values.

```
cereal_data<- cereal_data[complete.cases(cereal_data),]
# Transform the cereal name into a row name for better visualization of the data
cereal<-data.frame(cereal_data,row.names = 'name')
# Remove the 1st, 2nd and 12 columns as they are qualitative variables
cereal_Numerical <- cereal[,-c(1,2,12)]
```

Data Normalization

Since we are dealing with variables of very different nature that are expressed in different units, we have to normalized the data before applying any algorithm.

```
#Normalize the data using scale function
cereal_Norm <- scale(cereal_Numerical)

#Top rows of the dataset after preprocessing
head(cereal_Norm)
```

```
##                              calories   protein        fat     sodium
## 100%_Bran                  -1.8659155  1.3817478  0.0000000 -0.3910227
## 100%_Natural_Bran           0.6537514  0.4522084  3.9728810 -1.7804186
## All-Bran                   -1.8659155  1.3817478  0.0000000  1.1795987
## All-Bran_with_Extra_Fiber -2.8737823  1.3817478 -0.9932203 -0.2702057
## Apple_Cinnamon_Cheerios     0.1498180 -0.4773310  0.9932203  0.2130625
## Apple_Jacks                 0.1498180 -0.4773310 -0.9932203 -0.4514312
##                                 fiber      carbo     sugars     potass
## 100%_Bran                   3.22866747 -2.5001396 -0.2542051  2.5605229
## 100%_Natural_Bran          -0.07249167 -1.7292632  0.2046041  0.5147738
## All-Bran                    2.81602258 -1.9862220 -0.4836096  3.1248675
## All-Bran_with_Extra_Fiber   4.87924705 -1.7292632 -1.6306324  3.2659536
## Apple_Cinnamon_Cheerios    -0.27881412 -1.0868662  0.6634132 -0.4022862
## Apple_Jacks                -0.48513656 -0.9583868  1.5810314 -0.9666308
##                              vitamins     weight       cups     rating
## 100%_Bran                  -0.1818422 -0.2008324 -2.0856582  1.8549038
## 100%_Natural_Bran          -1.3032024 -0.2008324  0.7567534 -0.5977113
## All-Bran                   -0.1818422 -0.2008324 -2.0856582  1.2151965
## All-Bran_with_Extra_Fiber -0.1818422 -0.2008324 -1.3644493  3.6578436
## Apple_Cinnamon_Cheerios    -0.1818422 -0.2008324 -0.3038480 -0.9165248
## Apple_Jacks                -0.1818422 -0.2008324  0.7567534 -0.6553998
```

```
# To check for the dimensions of the dataset after data preprocessing
dim(cereal_Norm)
```

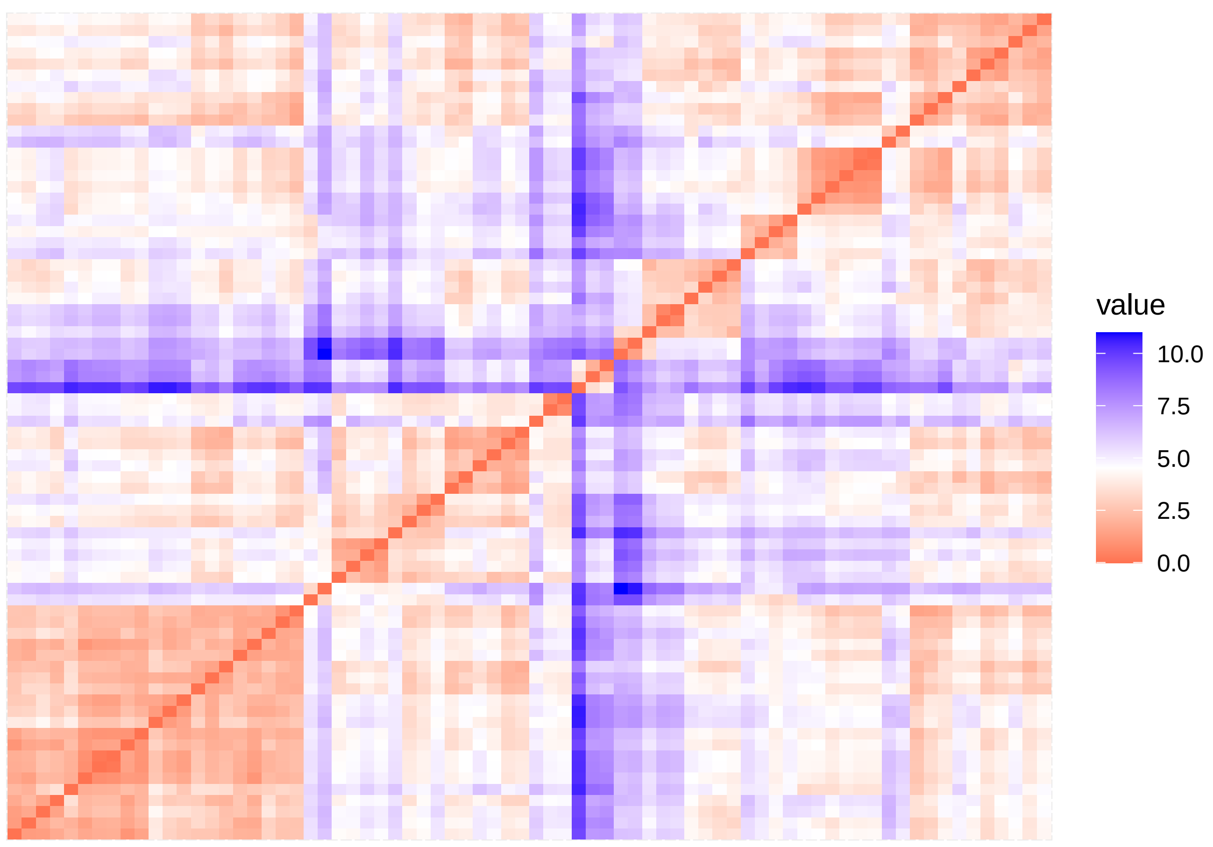
```
## [1] 74 12
```

There are 74 observations and 12 features.

Distance Measure

For computing distance, we are going to use the **get__distance** function.It uses the Euclidean distance as default metric and **fviz__clust** is to visualize the distance matrix.

```
#Computing distance. Euclidean distance as a default metric
distance<- get_dist(cereal_Norm)
# Plotting the distance.
fviz_dist(distance,show_labels = FALSE)
```



Distance Matrix Plot

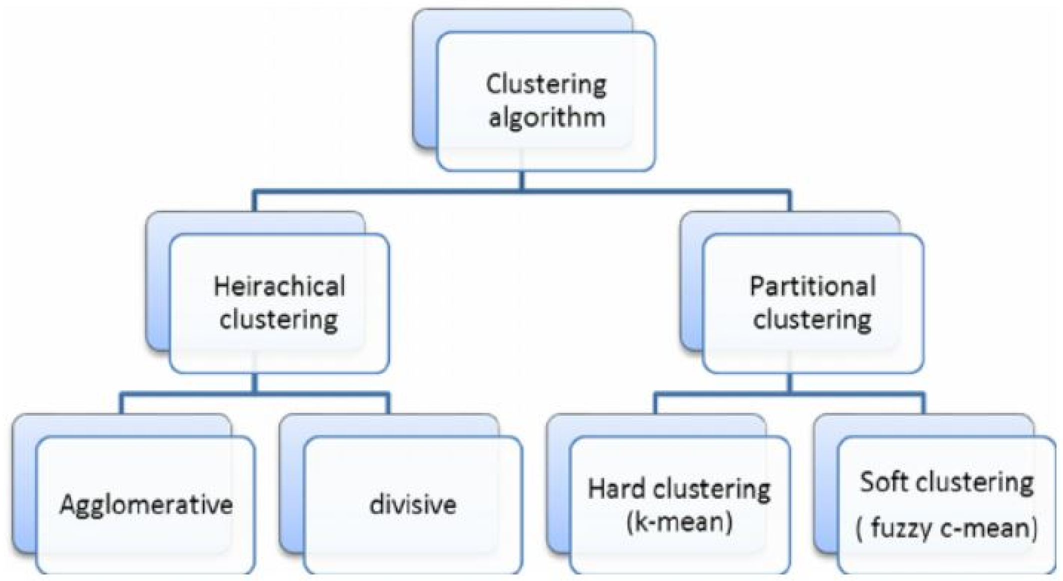
The above plot shows the different intensity of color for different distances. As we can see , the diagonal has a value equal to zero because it indicates the distance of an observation from itself.The purple indicates that farthest distance between the point and red indicates the nearest distance.

Questions:

A. 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.

Hierarchical Clustering:

This clustering is one of the unsupervised machine learning algorithm to group data points based on the similarities.In other words, it is a hierarchical decomposition of the data based on the group similarities.



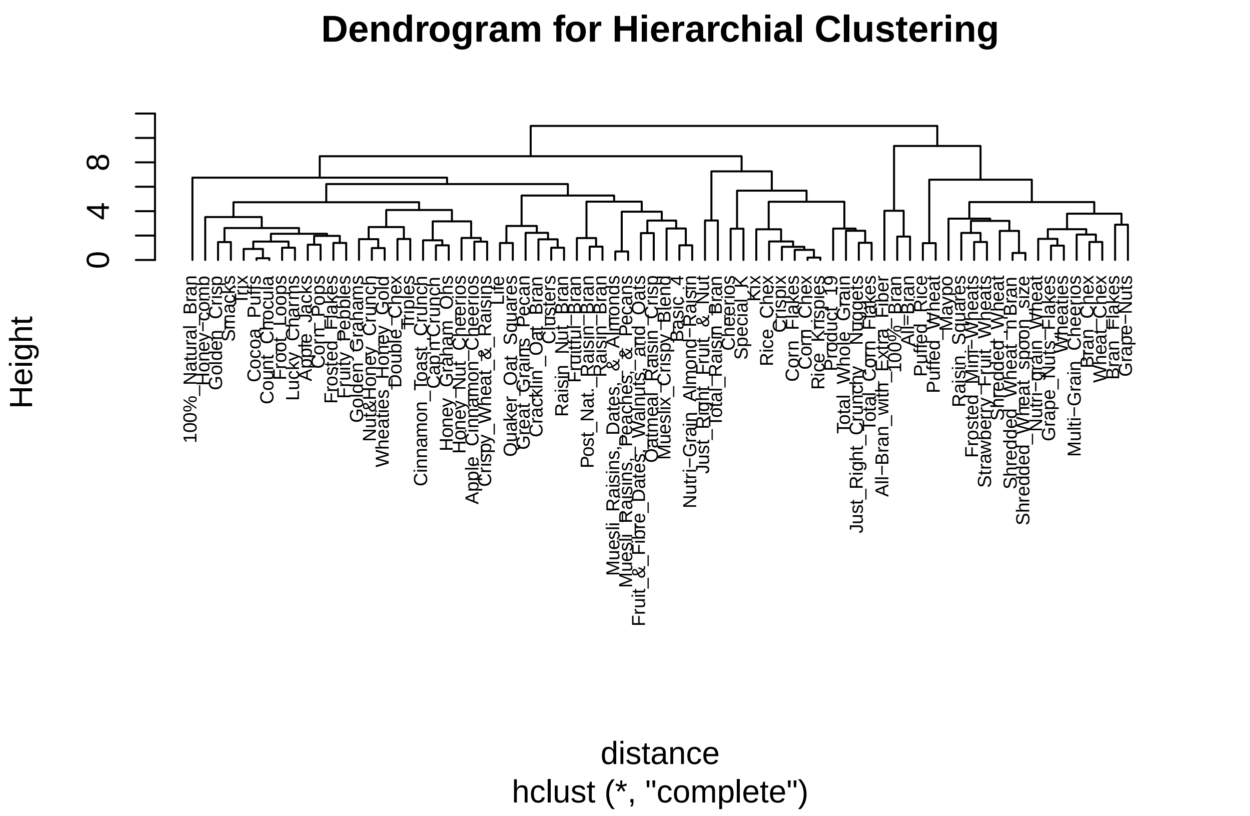
As we can see in the above image that, clustering is divided into two categories. Hierarchical clustering uses two techniques to group the data.

- **Agglomerative Nesting (AGNES)** - Bottom to Up approach
- **Divisive Analysis (DIANA)** - Top to Bottom approach

Now lets perform Hierarchical clustering using hclust function

```
set.seed(123)
## Now we run Hierarchical Clustering using complete linkage
hc1<- hclust(distance,method = "complete")
```

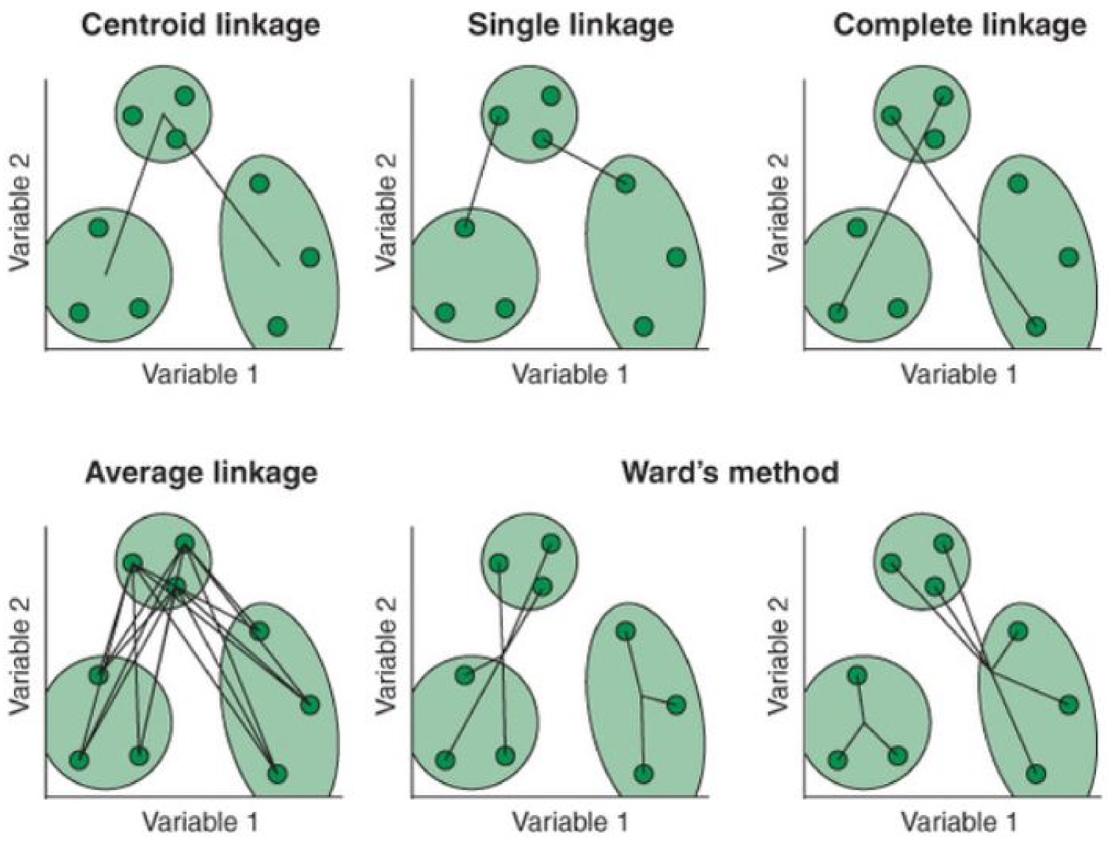
```
set.seed(123)
#plot the obtained dendrogram
plot(hc1,cex=0.6,hang=-1,
     main= "Dendrogram for Hierarchial Clustering")
```



From the above plot we can see that hierarchy resembles as tree structure which is called as dendrogram. This dendrogram helps us to define the number of clusters needed to classify the dataset.

AGNES Method

This AGNES method is a bottom-up approach and we are going to use this method to compare different linkages.



```
set.seed(123)

# Compute with Agnes method and different linkage methods

hc_single<-agnes(distance,method = "single")
hc_complete<-agnes(distance,method = "complete")
hc_average<-agnes(distance,method = "average")
hc_ward <- agnes(distance,method = "ward")

# Compare  Agglomerative Coefficients
print(hc_single$ac)
```

```
## [1] 0.6072384
```

```
print(hc_complete$ac)
```

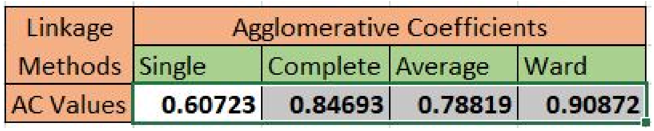
```
## [1] 0.8469328
```

```
print(hc_average$ac)
```

```
## [1] 0.7881955
```

```
print(hc_ward$ac)
```

```
## [1] 0.9087265
```



Linkage	Agglomerative Coefficients			
Methods	Single	Complete	Average	Ward
AC Values	**0.60723**	**0.84693**	**0.78819**	**0.90872**

OBSERVATIONS:

Here, we can see that **Ward's method** is the best method and identifies the strongest clustering structure of the four methods assessed.

Ward method - find the pair of clusters that leads to minimum increase in total within-cluster variance after merging(which is 90.4% accurate)

B. How many clusters would you choose?

To find the optimal value of K:

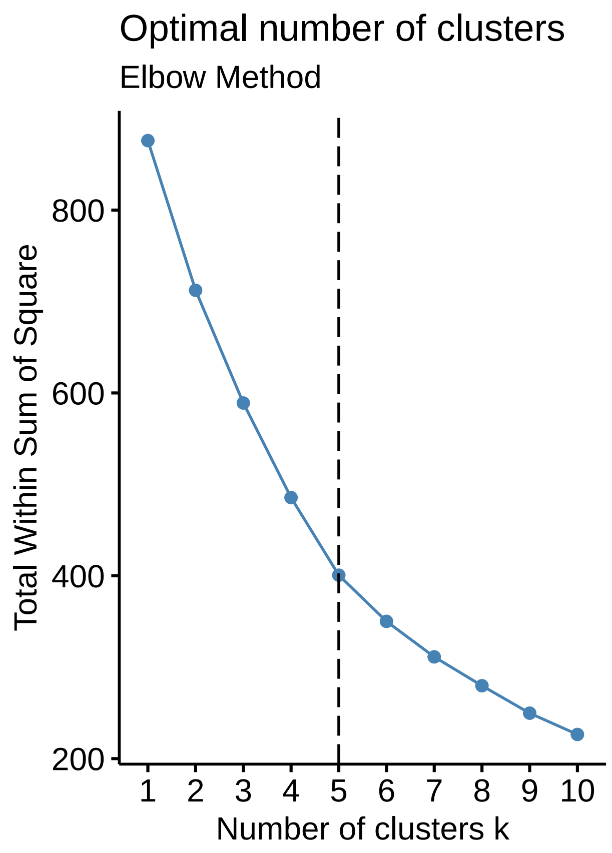
In a dendrogram, the height of each of the branch represents the distance between the two data points being connected. Here, the height describes about variations in the nutrients content in cereals. Taller the branch more cereals with dissimilarities are included in cluster which will increase the nutrition value of the cluster.

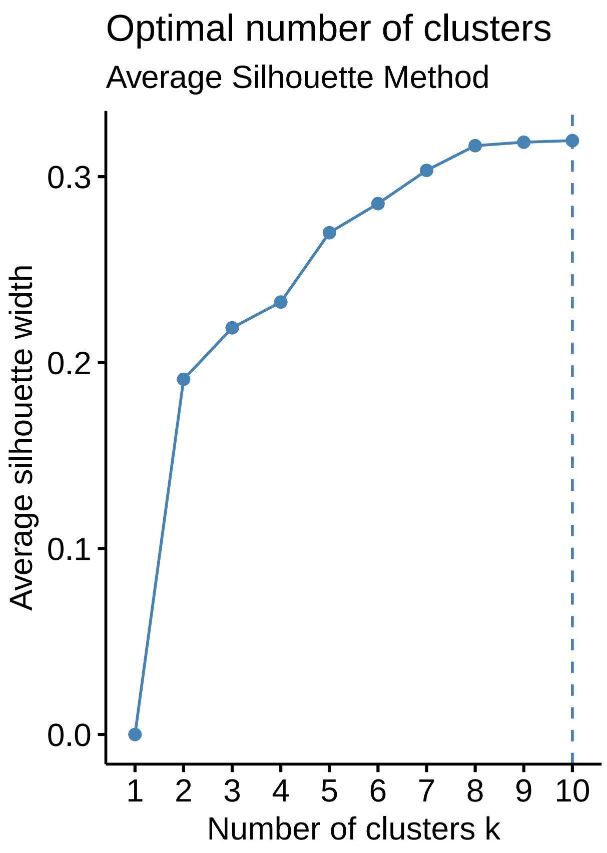
Generally in many cases, The number of clusters(K) can be determined by height of the tree in dendrogram. To be more accurate, I wanted to follow the two standard approaches.

- Elbow Method
- Average Silhouette Method

```
# Elbow Method
elbow<-fviz_nbclust(cereal_Norm, FUN = hcut,method = "wss") +
  geom_vline(xintercept = 5,linetype=5)+
  labs(subtitle = "Elbow Method")

# silhouette Method
sil<-fviz_nbclust(cereal_Norm, FUN = hcut,method = "silhouette") +
  labs(subtitle = "Average Silhouette Method")
plot_grid(elbow,sil)
```





From above plots, we can see that k=5 from elbow method and k= 10 from Average silhouette method. However we can see that with 5 clusters we could get a satisfying result and also looks good for interpretation.

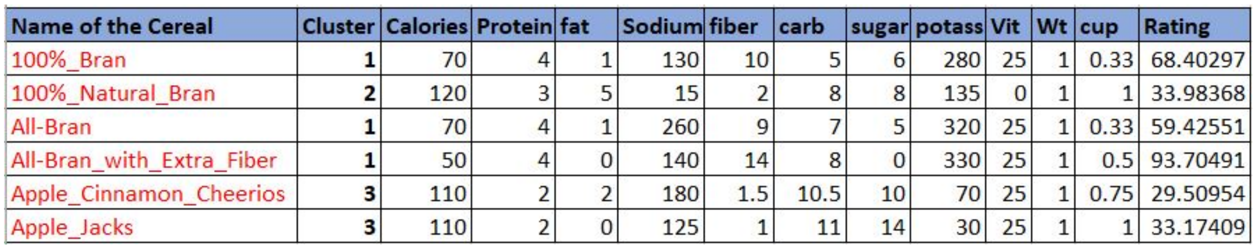
Clustering data with K=5

```
# Cut the tree with k=5
cluster5<-cutree(hc_ward,k=5)
# Number of items in each cluster
table(cluster5)
```

```
## cluster5
##  1  2  3  4  5
##  3 19 21 22  9
```

```
# Interpret the cluster data with their corresponding cluster number
total_cluster <- cbind(cluster5, cereal_Numerical)
colnames(total_cluster)[1] <- "clusters5"
```

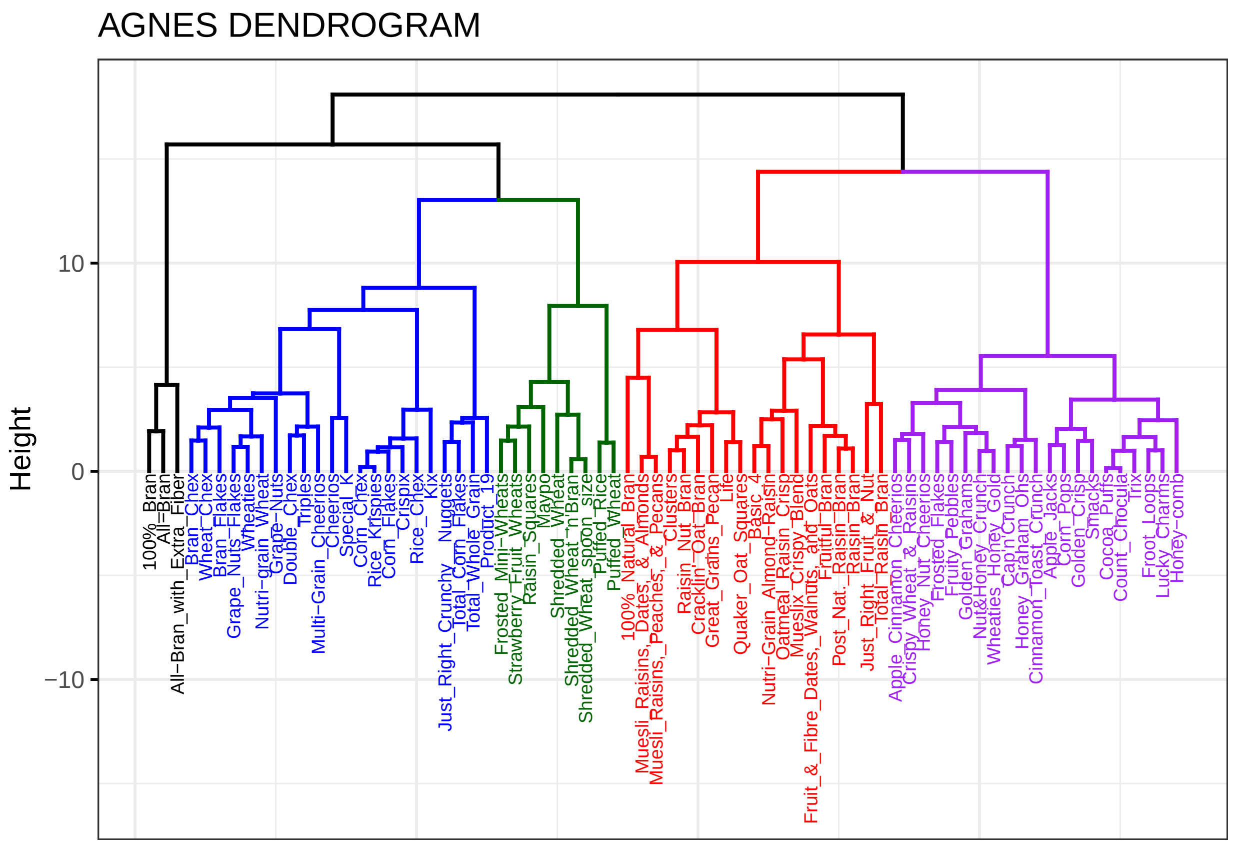
The below table represents the top rows of the data with their corresponding cluster number.



Name of the Cereal	Cluster	Calories	Protein	fat	Sodium	fiber	carb	sugar	potass	Vit	Wt	cup	Rating
100%_Bran	1	70	4	1	130	10	5	6	280	25	1	0.33	68.40297
100%_Natural_Bran	2	120	3	5	15	2	8	8	135	0	1	1	33.98368
All-Bran	1	70	4	1	260	9	7	5	320	25	1	0.33	59.42551
All-Bran_with_Extra_Fiber	1	50	4	0	140	14	8	0	330	25	1	0.5	93.70491
Apple_Cinnamon_Cheerios	3	110	2	2	180	1.5	10.5	10	70	25	1	0.75	29.50954
Apple_Jacks	3	110	2	0	125	1	11	14	30	25	1	1	33.17409

AGNES dendrogram for k=5

```
# AGNES Dendrogram for k=5
fviz_dend(hc_ward,k=5,
          color_labels_by_k = TRUE,
          k_colors = c("Black","Blue","DarkGreen","Red","Purple"),
          cex=0.5,labels_track_height = 16,
          main = "AGNES DENDROGRAM",
          ggtheme=theme_bw())
```



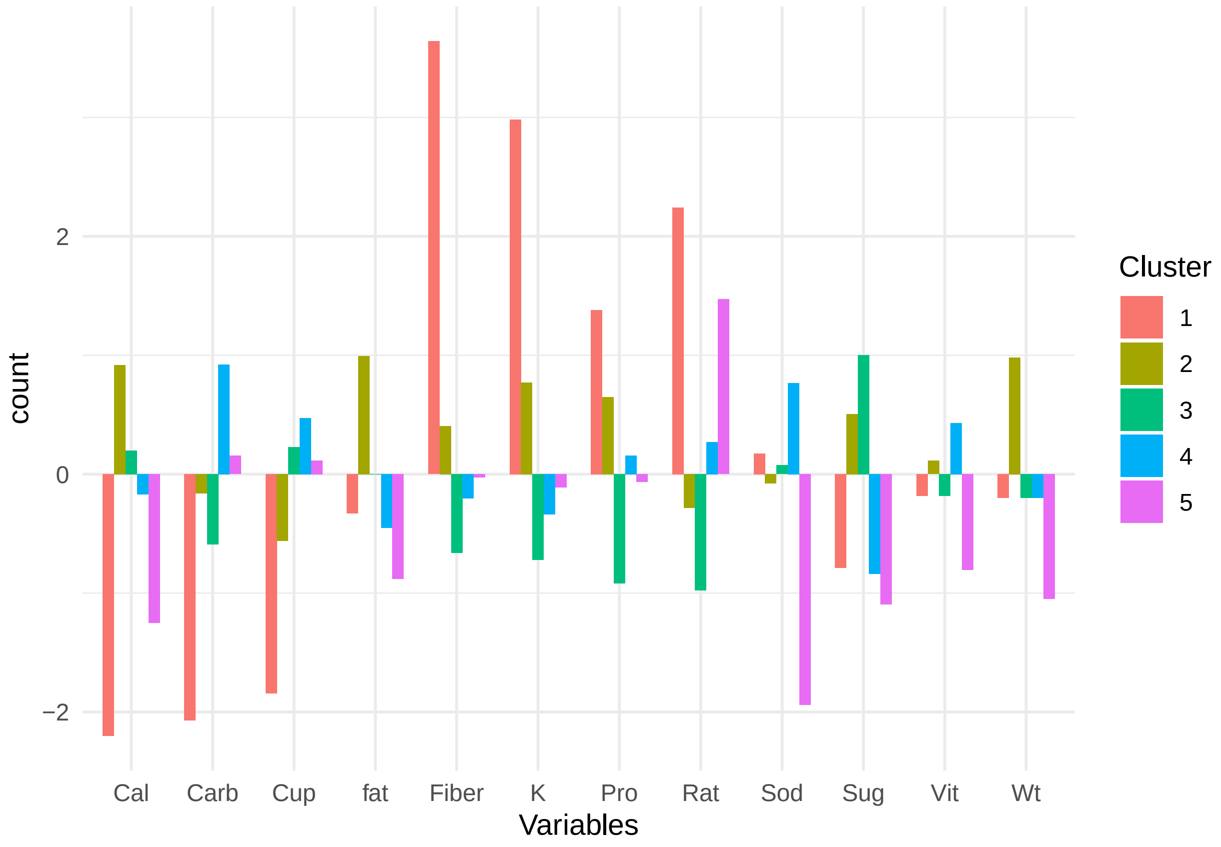
This is the dendrogram of AGNES using Ward linkage method where k=5.

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

To check stability, partition the data and see how well clusters formed based on one part apply to the other part. To do this:

- Cluster partition A.
- Use the cluster centroids from A to assign each record in partition B (each record is assigned to the cluster with the closest centroid).
- Assess how consistent the cluster assignments are compared to the assignments based on all the data.

Interpretation of clusters with respect to the variables.



Data Split:

To check the stability, we can divide the original dataset in 2 parts, partition A and partition B. Then, we see how similar the sets of clusters have been created for the 2 partitions and the original dataset.

More precisely, we are going to allocate 50% of the data to partition A and the rest 50% to partition B.

```r
set.seed(123)
#Partiton A with 50% of data i.e, 37 observations
PartitionA <- cereal_Numerical[1:37,]
#Partiton B with 50% of data i.e, remaining 37 observations
```

```
PartitionB <- cereal_Numerical[38:74,]

# Normalisation of two partitions
Partition_A_Norm<-scale(PartitionA)
Partition_B_Norm<-scale(PartitionB)
```

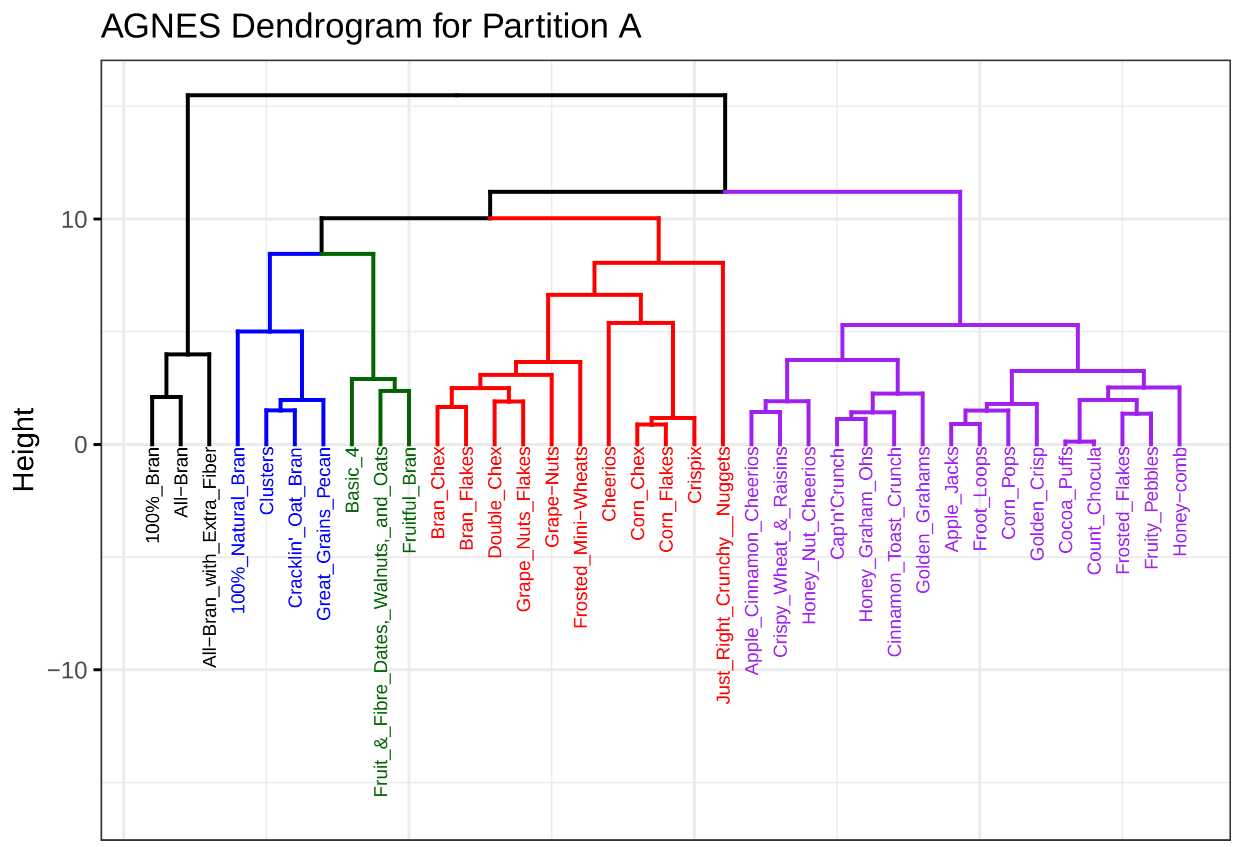
Clustering Partition A

```
set.seed(123)
#Compute the distances, Euclidean Metric is used as default
distance_A <- get_dist(Partition_A_Norm)
#Apply Hierarchical clustering to Partition A using ward method
hc_w_A <- agnes(distance_A, method = "ward")
#Cut tree into 5 groups
cluster_A<-cutree(hc_w_A,k=5)

# Store the clusters in a data frame along with the cereals data
cereal_A <- cbind(cluster_A, Partition_A_Norm)
# Have a look at the head of the new data frame
colnames(cereal_A)[1] <- "cluster_A"

#Number of items in each cluster
table(cluster_A)
```

```
## cluster_A
##  1  2  3  4  5
##  3  4 16  3 11
```

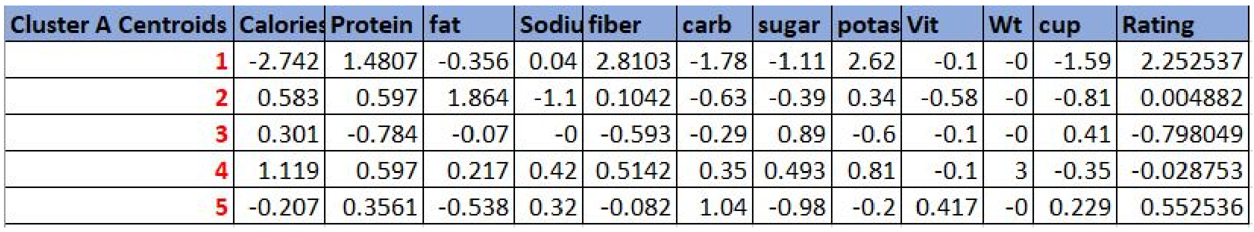


After clustering the partition A, we have to use the cluster centroids from A to assign each record in partition B (each record is assigned to the cluster with the closest centroid).

Calculate Centroids of Cluster A :

```
centroid_A<- aggregate(Partition_A_Norm, by=list(cluster=cluster_A), mean)
```

The below table indicates the Centroids of Partition A Clusters



Cluster A Centroids	Calories	Protein	fat	Sodiu	fiber	carb	sugar	potas	Vit	Wt	cup	Rating
1	-2.742	1.4807	-0.356	0.04	2.8103	-1.78	-1.11	2.62	-0.1	-0	-1.59	2.252537
2	0.583	0.597	1.864	-1.1	0.1042	-0.63	-0.39	0.34	-0.58	-0	-0.81	0.004882
3	0.301	-0.784	-0.07	-0	-0.593	-0.29	0.89	-0.6	-0.1	-0	0.41	-0.798049
4	1.119	0.597	0.217	0.42	0.5142	0.35	0.493	0.81	-0.1	3	-0.35	-0.028753
5	-0.207	0.3561	-0.538	0.32	-0.082	1.04	-0.98	-0.2	0.417	-0	0.229	0.552536

Now after we calculated the centroids of the clusters found by hierarchical clustering we will assign each observation of dataset B to the closest centroids of A clusters.

Assign observations of Partition B to the cluster centroids of Partition A.

```
# Now we can assign observations of B partition based on the clusters centroids of partition A.
Assign<- data.frame(Cer=seq(1,37,1),cluster=rep(0,37))

for(i in 1:37)
{
  q<-as.data.frame(rbind(centroid_A[,-1],Partition_B_Norm[i,]))
  m<-as.matrix(get_dist(q))
  Assign[i,2]<-which.min(m[6,-6])
}

rownames(Assign) <-rownames(Partition_B_Norm)
```

We need to know which clusters the observations of partition B are assigned to.

```
# Display the top rows of the Partition B
head(table(Assign))
```

```
##    cluster
## Cer 2 3 4 5
##   1 0 0 1 0
##   2 0 0 0 1
##   3 1 0 0 0
##   4 0 1 0 0
##   5 0 0 0 1
##   6 1 0 0 0
```

```
table(Assign$cluster==total_cluster[38:74,1])
```

```
##
## FALSE  TRUE
##    18    19
```

When we check for the accuracy of parition B, it is almost 55% and also we can see that all the records in the partition B is assigned to 2,3,4 and 5 clusters. Now we have to check for the cluster stability.

Cluster Stability:

*Here, I would like to explain cluster stability with help of Jaccard Values. Assessment of the clusterwise stability of a clustering of data, which can be cases*variables or dissimilarity data. The data is resampled using several schemes (bootstrap, subsetting, jittering, replacement of points by noise) and the Jaccard similarities of the original clusters to the most similar clusters in the resampled data are computed. The mean over these similarities is used as an index of the stability of a cluster (other statistics can be computed as well). The methods are described in Hennig (2007) (From Help Document).*

Clusterboot is an integrated function that computes the clustering as well, using interface functions for various clustering methods implemented in R

```
set.seed(123)
#Input the scaled cereals_data
hclust_stability = clusterboot(cereal_Norm, clustermethod=hclustCBI, method="ward.D2",
                              k=5,
                              count = FALSE)
#Analyse the clustering results
clus<-hclust_stability$result$partition
```

Cluster stabiltiy values:

- Average Jaccard Values > 0.85 denote "Highly Stable" clusters.
- Values 0.6 - 0.75 , clusters may be considered as indicating patterns in the data, but which points exactly should belong to these clusters is highly doubtful.
- Values below 0.6, clusters should not be trusted.

```
jaccard_mean<-hclust_stability$bootmean
jaccard_mean
```

```
## [1] 0.8242900 0.7898645 0.9500748 0.6871245 0.6540623
```

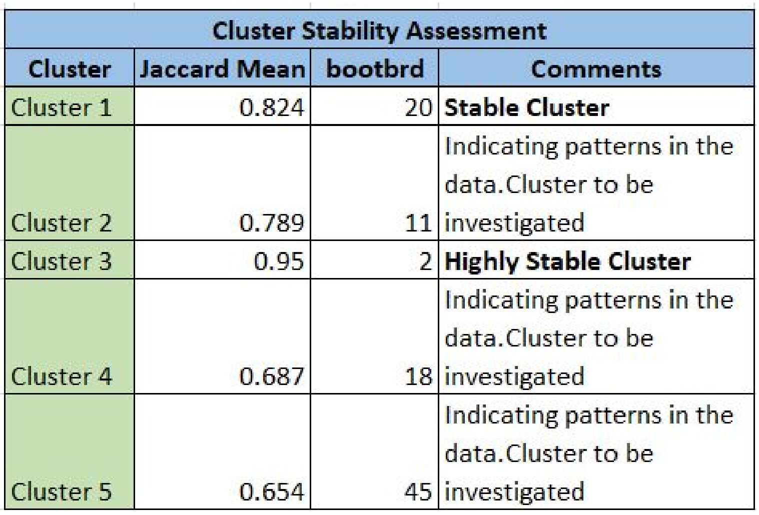
How many times the different clusters were dissolved

bootbrd is for clusterwise number of times a cluster has been dissolved. Lower the value, the better

```
hclust_stability$bootbrd
```

```
## [1] 20 11  2 18 45
```

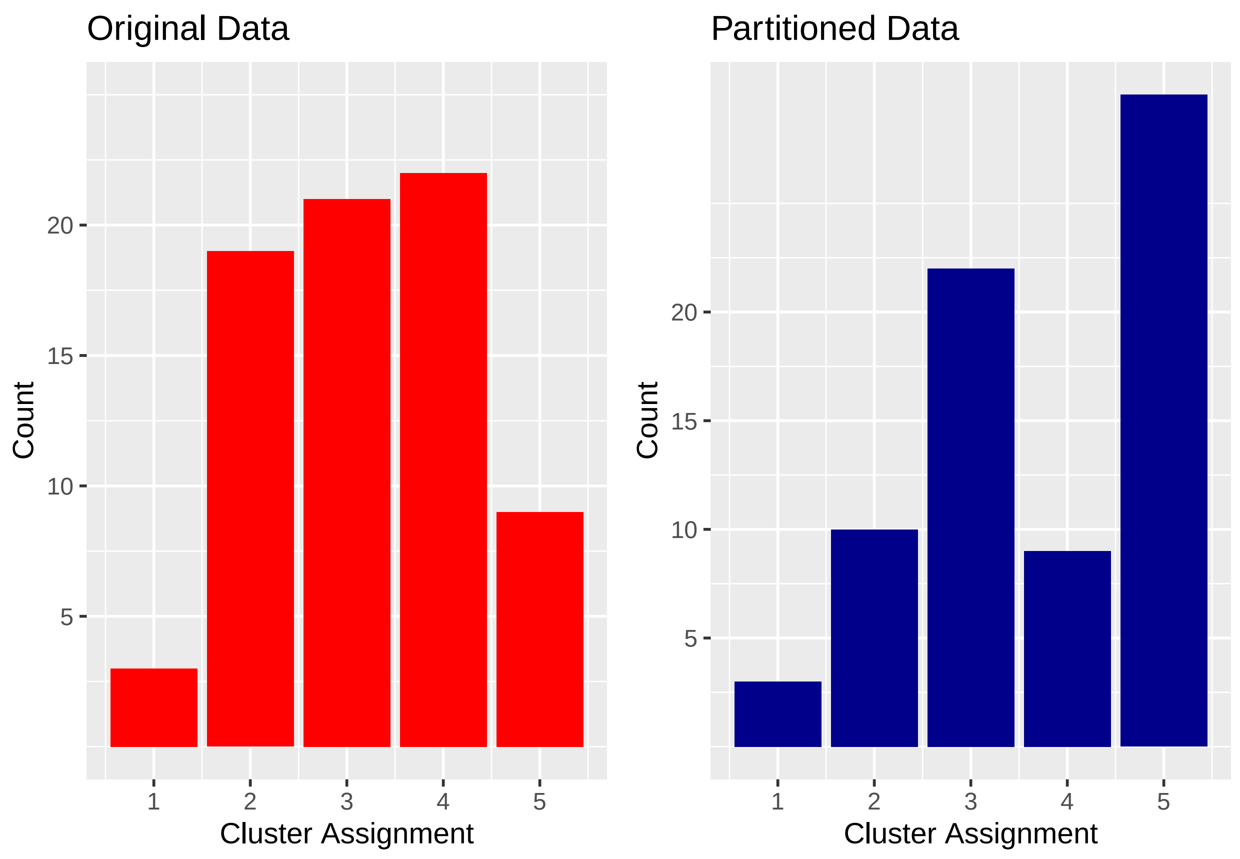
OBSERVATIONS:



Cluster Stability Assessment			
Cluster	Jaccard Mean	bootbrd	Comments
Cluster 1	0.824	20	**Stable Cluster**
Cluster 2	0.789	11	Indicating patterns in the data.Cluster to be investigated
Cluster 3	0.95	2	**Highly Stable Cluster**
Cluster 4	0.687	18	Indicating patterns in the data.Cluster to be investigated
Cluster 5	0.654	45	Indicating patterns in the data.Cluster to be investigated

- We can see that cluster1 and 3 are stable clusters. Whereas clusters 2,4,5 shows some patterns that needs to be investigated more.

Visualize the cluster assignments to see any difference between the original and partitioned data



OBSERVATIONS:

- From the above plot, we can see that clusters 2 and 4 significantly shrunk when using the partitioned data. Whereas, cluster5 has become larger.
- When we consider, Clusters 1 and 3 looks similar as the original data clusters(Which are considered to be stable)
- We found that clusters 2,4 and 5 has to be investigated more in the cluster stability analysis and it is clearly evident that 2,4,5 are showing some patterns but should be investigated even in the partitioned data.

D. 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?

In my opinion, the data should not be normalized when looking at the health options for elementary schools. To support the healthy diet, you would want to find cereals with lowest sugar. You should also consider the cereals with the highest fiber,vitamins, and protein content to support health grow. And also it is more meaningful and easier to compare if we look at the variables in their original scale.

```r
#summary table of the number of cereals per customer
table(total_cluster$clusters5)
```

```
##
##  1  2  3  4  5
##  3 19 21 22  9
```

Interpretation of clusters with their names of the cereals

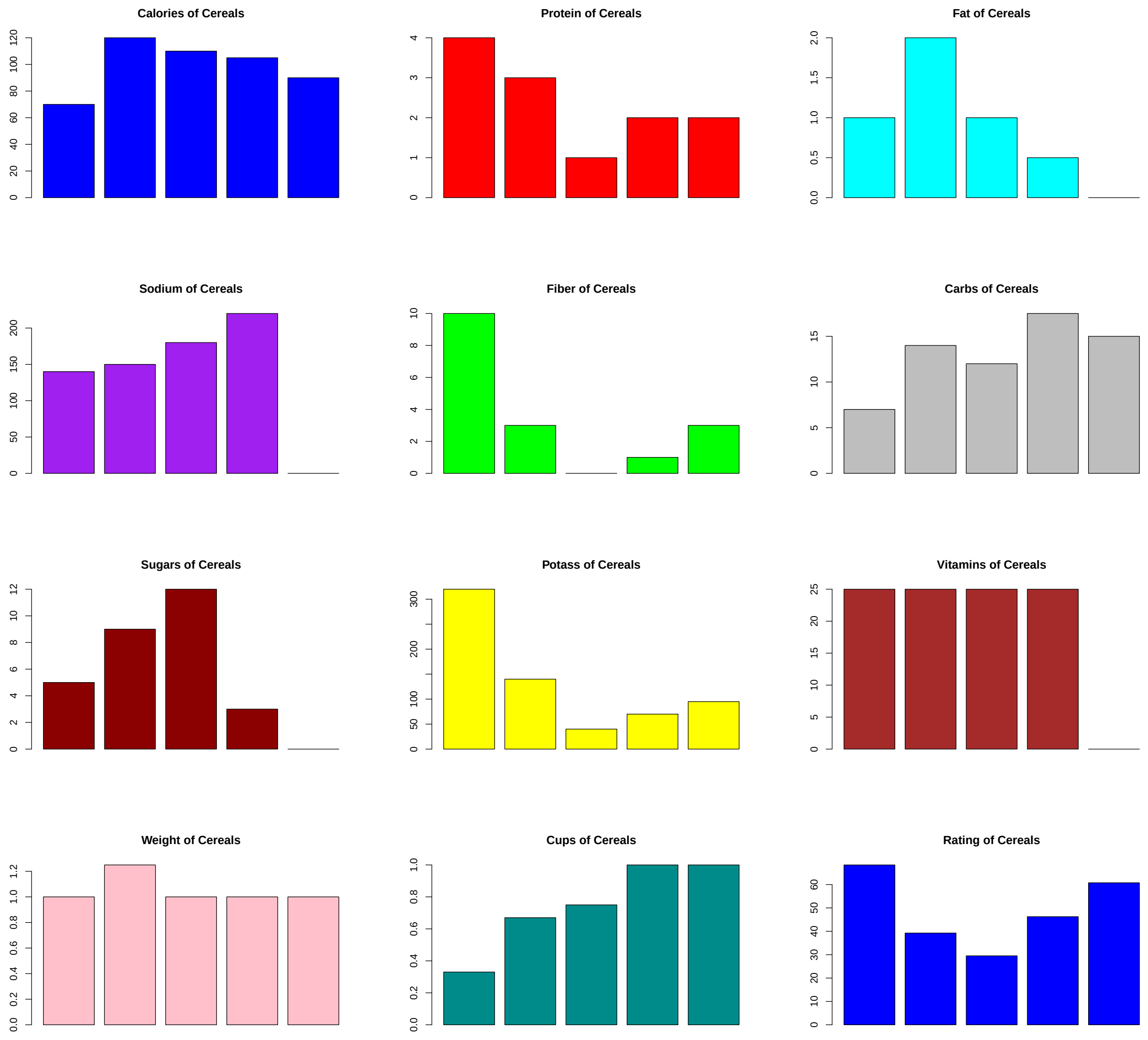
```r
lapply(unique(cluster5),function(clust)cereal_data$name[cluster5==clust])
```

```
## [[1]]
## [1] "100%_Bran"                 "All-Bran"
## [3] "All-Bran_with_Extra_Fiber"
##
## [[2]]
##  [1] "100%_Natural_Bran"
##  [2] "Basic_4"
##  [3] "Clusters"
##  [4] "Cracklin'_Oat_Bran"
##  [5] "Fruit_&_Fibre_Dates,_Walnuts,_and_Oats"
##  [6] "Fruitful_Bran"
##  [7] "Great_Grains_Pecan"
##  [8] "Just_Right_Fruit_&_Nut"
##  [9] "Life"
## [10] "Muesli_Raisins,_Dates,_&_Almonds"
## [11] "Muesli_Raisins,_Peaches,_&_Pecans"
## [12] "Mueslix_Crispy_Blend"
## [13] "Nutri-Grain_Almond-Raisin"
## [14] "Oatmeal_Raisin_Crisp"
## [15] "Post_Nat._Raisin_Bran"
## [16] "Quaker_Oat_Squares"
## [17] "Raisin_Bran"
## [18] "Raisin_Nut_Bran"
## [19] "Total_Raisin_Bran"
##
## [[3]]
##  [1] "Apple_Cinnamon_Cheerios" "Apple_Jacks"
##  [3] "Cap'n'Crunch"            "Cinnamon_Toast_Crunch"
##  [5] "Cocoa_Puffs"             "Corn_Pops"
##  [7] "Count_Chocula"           "Crispy_Wheat_&_Raisins"
##  [9] "Froot_Loops"             "Frosted_Flakes"
## [11] "Fruity_Pebbles"          "Golden_Crisp"
## [13] "Golden_Grahams"          "Honey_Graham_Ohs"
## [15] "Honey_Nut_Cheerios"      "Honey-comb"
## [17] "Lucky_Charms"            "Nut&Honey_Crunch"
## [19] "Smacks"                  "Trix"
## [21] "Wheaties_Honey_Gold"
##
## [[4]]
##  [1] "Bran_Chex"                   "Bran_Flakes"
##  [3] "Cheerios"                    "Corn_Chex"
##  [5] "Corn_Flakes"                 "Crispix"
##  [7] "Double_Chex"                 "Grape_Nuts_Flakes"
##  [9] "Grape-Nuts"                  "Just_Right_Crunchy__Nuggets"
## [11] "Kix"                         "Multi-Grain_Cheerios"
## [13] "Nutri-grain_Wheat"           "Product_19"
## [15] "Rice_Chex"                   "Rice_Krispies"
## [17] "Special_K"                   "Total_Corn_Flakes"
## [19] "Total_Whole_Grain"           "Triples"
## [21] "Wheat_Chex"                  "Wheaties"
##
## [[5]]
## [1] "Frosted_Mini-Wheats"       "Maypo"
## [3] "Puffed_Rice"               "Puffed_Wheat"
## [5] "Raisin_Squares"            "Shredded_Wheat"
## [7] "Shredded_Wheat_'n'Bran"    "Shredded_Wheat_spoon_size"
## [9] "Strawberry_Fruit_Wheats"
```

```
# Display the summary table of the clusters with their median nutritional values.
cluster_summary <-aggregate(cereal[,-c(1:2)],list(cluster5),median)
cluster_summary
```

```
##   Group.1 calories protein fat sodium fiber carbo sugars potass vitamins shelf
## 1       1       70       4 1.0    140    10   7.0      5    320       25   3.0
## 2       2      120       3 2.0    150     3  14.0      9    140       25   3.0
## 3       3      110       1 1.0    180     0  12.0     12     40       25   2.0
## 4       4      105       2 0.5    220     1  17.5      3     70       25   2.5
## 5       5       90       2 0.0      0     3  15.0      0     95        0   2.0
##   weight cups   rating
## 1   1.00 0.33 68.40297
## 2   1.25 0.67 39.25920
## 3   1.00 0.75 29.50954
## 4   1.00 1.00 46.26108
## 5   1.00 1.00 60.75611
```

Summary of the nutrition values by cluster



OBSERVATIONS

When we clearly observe the above plots,

- **Cluster1 :** High Protein and fiber with Low fat and sugar - Low Calorie cereals
- **Cluster2 :** High Calorie and High fat cereals
- **Cluster3 :** High sugar with less fiber and low rated cereals
- **Cluster4 :** High Sodium levels and High Carb cereals
- **Cluster5 :** Low fat and Low sugar with least sodium levels - Moderate vitamins with moderate fiber and protein Cereals.
- Healthy cereals with much dietary filters, less calories and less fats, high proteins extra fibers and potass: 100% Bran, All-Bran with and All-Bran , which grouped by the cluster 1.

- We can notice that cluster 1 has the highest Customer rating.
- Also, Cluster 5 has the less sugar, low fat and low sodium levels with moderate fiber, protein and also with moderate vitamin levels with variety of options which satisfy the constraint of every day a different cereal is offered

In a nutshell, we can propose that Clusters 1 and 5 seem to be the best choices for providing a healthy food solution to children.