Cluster Analysis on Breakfast Cereal

Problem Description

- In the following Unsupervised Learning activity, we try to cluster various types of breakfast cereal based on their nutritional content.

```
setwd("~/R_KSU/ML/Assignment5")
cereals_data <- read.csv("Cereals.csv", header=T)
data <- cereals_data
str(cereals_data)
```

```
## 'data.frame':    77 obs. of  16 variables:
##  $ name    : chr  "100%_Bran" "100%_Natural_Bran" "All-Bran" "All-Bran_with_Extra_Fiber" ...
##  $ mfr     : chr  "N" "Q" "K" "K" ...
##  $ type    : chr  "C" "C" "C" "C" ...
##  $ calories: int  70 120 70 50 110 110 110 130 90 90 ...
##  $ protein : int  4 3 4 4 2 2 2 3 2 3 ...
##  $ fat     : int  1 5 1 0 2 2 0 2 1 0 ...
##  $ sodium  : int  130 15 260 140 200 180 125 210 200 210 ...
##  $ fiber   : num  10 2 9 14 1 1.5 1 2 4 5 ...
##  $ carbo   : num  5 8 7 8 14 10.5 11 18 15 13 ...
##  $ sugars  : int  6 8 5 0 8 10 14 8 6 5 ...
##  $ potass  : int  280 135 320 330 NA 70 30 100 125 190 ...
##  $ vitamins: int  25 0 25 25 25 25 25 25 25 25 ...
##  $ shelf   : int  3 3 3 3 3 1 2 3 1 3 ...
##  $ weight  : num  1 1 1 1 1 1 1 1.33 1 1 ...
##  $ cups    : num  0.33 1 0.33 0.5 0.75 0.75 1 0.75 0.67 0.67 ...
##  $ rating  : num  68.4 34 59.4 93.7 34.4 ...
```

```
summary(cereals_data)
```

```
##      name               mfr                type              calories    
##  Length:77          Length:77          Length:77          Min.   : 50.0  
##  Class :character   Class :character   Class :character   1st Qu.:100.0  
##  Mode  :character   Mode  :character   Mode  :character   Median :110.0  
##                                                           Mean   :106.9  
##                                                           3rd Qu.:110.0  
##                                                           Max.   :160.0  
##                                                                          
##     protein           fat            sodium          fiber       
##  Min.   :1.000   Min.   :0.000   Min.   :  0.0   Min.   : 0.000  
##  1st Qu.:2.000   1st Qu.:0.000   1st Qu.:130.0   1st Qu.: 1.000  
##  Median :3.000   Median :1.000   Median :180.0   Median : 2.000  
##  Mean   :2.545   Mean   :1.013   Mean   :159.7   Mean   : 2.152  
##  3rd Qu.:3.000   3rd Qu.:2.000   3rd Qu.:210.0   3rd Qu.: 3.000  
##  Max.   :6.000   Max.   :5.000   Max.   :320.0   Max.   :14.000  
##                                                                  
##      carbo          sugars           potass          vitamins     
##  Min.   : 5.0   Min.   : 0.000   Min.   : 15.00   Min.   :  0.00  
##  1st Qu.:12.0   1st Qu.: 3.000   1st Qu.: 42.50   1st Qu.: 25.00  
##  Median :14.5   Median : 7.000   Median : 90.00   Median : 25.00  
##  Mean   :14.8   Mean   : 7.026   Mean   : 98.67   Mean   : 28.25  
##  3rd Qu.:17.0   3rd Qu.:11.000   3rd Qu.:120.00   3rd Qu.: 25.00  
##  Max.   :23.0   Max.   :15.000   Max.   :330.00   Max.   :100.00  
##  NA's   :1      NA's   :1        NA's   :2                        
##      shelf           weight          cups           rating     
##  Min.   :1.000   Min.   :0.50   Min.   :0.250   Min.   :18.04  
##  1st Qu.:1.000   1st Qu.:1.00   1st Qu.:0.670   1st Qu.:33.17  
##  Median :2.000   Median :1.00   Median :0.750   Median :40.40  
##  Mean   :2.208   Mean   :1.03   Mean   :0.821   Mean   :42.67  
##  3rd Qu.:3.000   3rd Qu.:1.00   3rd Qu.:1.000   3rd Qu.:50.83  
##  Max.   :3.000   Max.   :1.50   Max.   :1.500   Max.   :93.70  
##                                                                
```

```
head(cereals_data)
```

	name	...	type	calories	protein	fat	sodium	fiber	carbo
	<chr>	<chr>	<chr>	<int>	<int>	<int>	<int>	<dbl>	<dbl>
1	100%_Bran	N	C	70	4	1	130	10.0	5.0
2	100%_Natural_Bran	Q	C	120	3	5	15	2.0	8.0
3	All-Bran	K	C	70	4	1	260	9.0	7.0
4	All-Bran_with_Extra_Fiber	K	C	50	4	0	140	14.0	8.0
5	Almond_Delight	R	C	110	2	2	200	1.0	14.0
6	Apple_Cinnamon_Cheerios	G	C	110	2	2	180	1.5	10.5

6 rows | 1-10 of 17 columns

```
tail(cereals_data)
```

	name	…	type	calories	protein	fat	sodium	fiber	carbo
	<chr>	<chr>	<chr>	<int>	<int>	<int>	<int>	<dbl>	<dbl>
72	Total_Whole_Grain	G	C	100	3	1	200	3	16
73	Triples	G	C	110	2	1	250	0	21
74	Trix	G	C	110	1	1	140	0	13
75	Wheat_Chex	R	C	100	3	1	230	3	17
76	Wheaties	G	C	100	3	1	200	3	17
77	Wheaties_Honey_Gold	G	C	110	2	1	200	1	16

6 rows | 1-10 of 17 columns

Data Pre-Processing

```
# Total number of NA values in the data set
colSums(is.na(cereals_data))
```

```
##     name      mfr     type calories  protein      fat   sodium    fiber
##        0        0        0        0        0        0        0        0
##    carbo   sugars   potass vitamins    shelf   weight     cups   rating
##        1        1        2        0        0        0        0        0
```

```
# comment: There are 4 NA values in dataset we shall remove those.
cereals_data <- na.omit(cereals_data)

#check for NA values again
colSums(is.na(cereals_data))
```

```
##     name      mfr     type calories  protein      fat   sodium    fiber
##        0        0        0        0        0        0        0        0
##    carbo   sugars   potass vitamins    shelf   weight     cups   rating
##        0        0        0        0        0        0        0        0
```

```
# Setting the rownames of the breakfast cereals to the row names, as this will later help us in
 visualizing the clusters.
data <- cereals_data
rownames(cereals_data) <- cereals_data$name
cereals_data$name = NULL
head(cereals_data)
```

	…	ty…	calories	protein	fat	sodi…	fiber	car…	sugars
	<chr>	<chr>	<int>	<int>	<int>	<int>	<dbl>	<dbl>	<int>
100%_Bran	N	C	70	4	1	130	10.0	5.0	6

	…	ty…	calories	protein	fat	sodi…	fiber	car…	sugars
	<chr>	<chr>	<int>	<int>	<int>	<int>	<dbl>	<dbl>	<int>
100%_Natural_Bran	Q	C	120	3	5	15	2.0	8.0	8
All-Bran	K	C	70	4	1	260	9.0	7.0	5
All-Bran_with_Extra_Fiber	K	C	50	4	0	140	14.0	8.0	0
Apple_Cinnamon_Cheerios	G	C	110	2	2	180	1.5	10.5	10
Apple_Jacks	K	C	110	2	0	125	1.0	11.0	14

6 rows | 1-10 of 16 columns

```
## Converting categorical variables into dummy variables
library(fastDummies)
cereals_data <- fastDummies::dummy_cols(cereals_data, select_columns = "mfr")[,-1]
cereals_data <- fastDummies::dummy_cols(cereals_data, select_columns = "type")[,-1]
cereals_data <- fastDummies::dummy_cols(cereals_data, select_columns = "shelf")[,-10]
str(cereals_data)
```

```
## 'data.frame':    74 obs. of  24 variables:
##  $ calories: int  70 120 70 50 110 110 130 90 90 120 ...
##  $ protein : int  4 3 4 4 2 2 3 2 3 1 ...
##  $ fat     : int  1 5 1 0 2 0 2 1 0 2 ...
##  $ sodium  : int  130 15 260 140 180 125 210 200 210 220 ...
##  $ fiber   : num  10 2 9 14 1.5 1 2 4 5 0 ...
##  $ carbo   : num  5 8 7 8 10.5 11 18 15 13 12 ...
##  $ sugars  : int  6 8 5 0 10 14 8 6 5 12 ...
##  $ potass  : int  280 135 320 330 70 30 100 125 190 35 ...
##  $ vitamins: int  25 0 25 25 25 25 25 25 25 25 ...
##  $ weight  : num  1 1 1 1 1 1 1.33 1 1 1 ...
##  $ cups    : num  0.33 1 0.33 0.5 0.75 1 0.75 0.67 0.67 0.75 ...
##  $ rating  : num  68.4 34 59.4 93.7 29.5 ...
##  $ mfr_A   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ mfr_G   : int  0 0 0 0 1 0 1 0 0 0 ...
##  $ mfr_K   : int  0 0 1 1 0 1 0 0 0 0 ...
##  $ mfr_N   : int  1 0 0 0 0 0 0 0 0 0 ...
##  $ mfr_P   : int  0 0 0 0 0 0 0 0 1 0 ...
##  $ mfr_Q   : int  0 1 0 0 0 0 0 0 0 1 ...
##  $ mfr_R   : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ type_C  : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ type_H  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ shelf_1 : int  0 0 0 0 1 0 0 1 0 0 ...
##  $ shelf_2 : int  0 0 0 0 0 1 0 0 0 1 ...
##  $ shelf_3 : int  1 1 1 1 0 0 1 0 1 0 ...
```

```
# Assigning cereal lables as row names of the data frame.
rownames(cereals_data) <- data$name
head(cereals_data)
```

	calories <int>	protein <int>	... <int>	sodi... <int>	fiber <dbl>	ca... <dbl>	sug... <int>	pota... <int>	vitamin <int
100%_Bran	70	4	1	130	10.0	5.0	6	280	2
100%_Natural_Bran	120	3	5	15	2.0	8.0	8	135	(
All-Bran	70	4	1	260	9.0	7.0	5	320	2
All-Bran_with_Extra_Fiber	50	4	0	140	14.0	8.0	0	330	2
Apple_Cinnamon_Cheerios	110	2	2	180	1.5	10.5	10	70	2
Apple_Jacks	110	2	0	125	1.0	11.0	14	30	2

6 rows | 1-10 of 25 columns

Data Normalization

```r
## Data Scaling
mean_norm_minmax <- function(x){
                                (x- mean(x)) /(max(x)-min(x))
}

cereals_data <- as.data.frame(lapply(cereals_data, mean_norm_minmax))
rownames(cereals_data) <- data$name
#cereals_data_norm <- scale(cereals_data_dum, center = T, scale = T)
head(cereals_data)
```

	calories <dbl>	protein <dbl>	fat <dbl>	sodium <dbl>	fiber <dbl>	ca <(
100%_Bran	-0.33660934	0.2972973	0.0	-0.1011402	0.55888031	-0.5405
100%_Natural_Bran	0.11793612	0.0972973	0.8	-0.4605152	-0.01254826	-0.3738
All-Bran	-0.33660934	0.2972973	0.0	0.3051098	0.48745174	-0.4294
All-Bran_with_Extra_Fiber	-0.51842752	0.2972973	-0.2	-0.0698902	0.84459459	-0.3738
Apple_Cinnamon_Cheerios	0.02702703	-0.1027027	0.2	0.0551098	-0.04826255	-0.2349
Apple_Jacks	0.02702703	-0.1027027	-0.2	-0.1167652	-0.08397683	-0.2072

6 rows | 1-7 of 25 columns

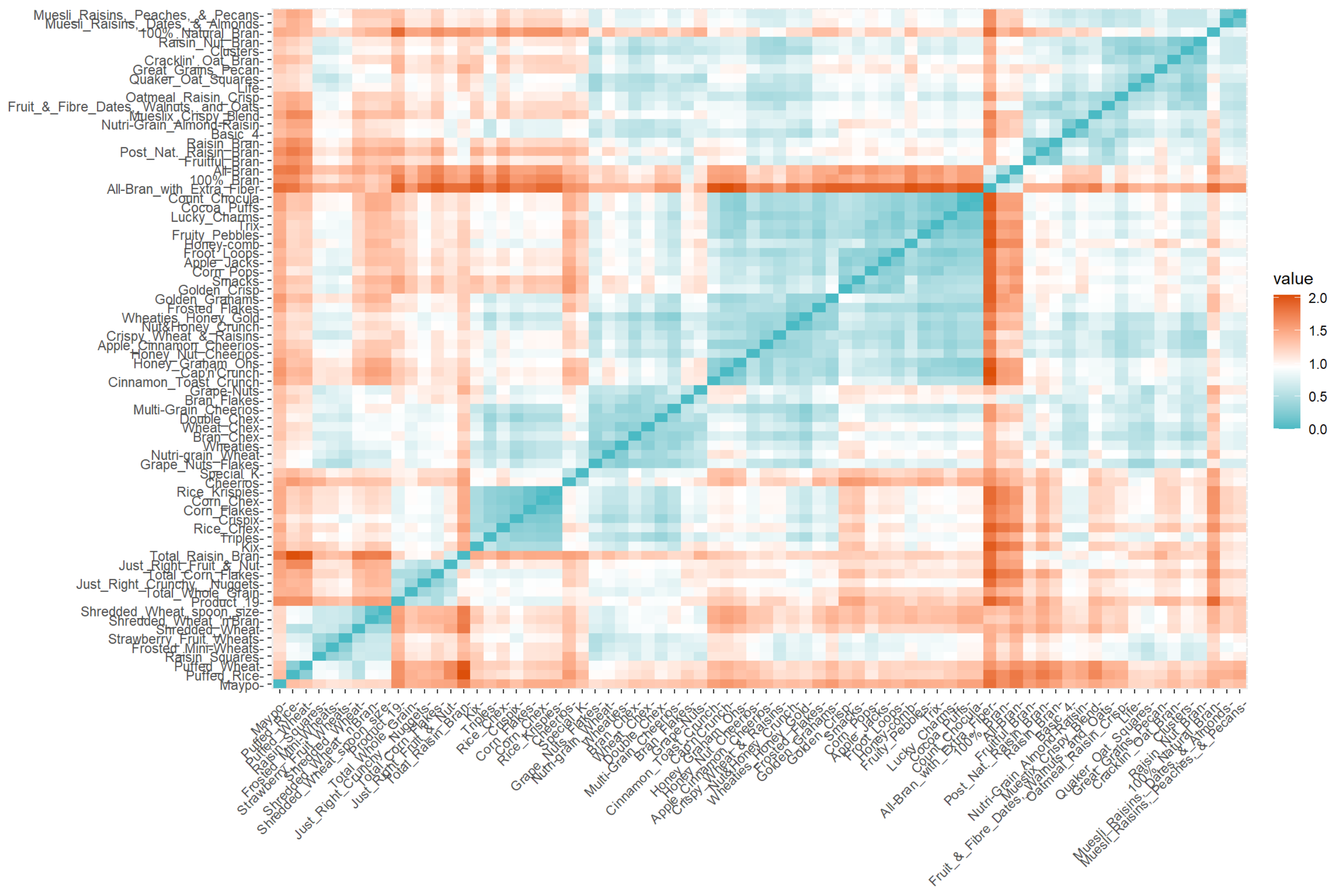
DATA EXPLORATION

```r
# Correlation chart avoiding the dummified variables
library(factoextra)
```

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

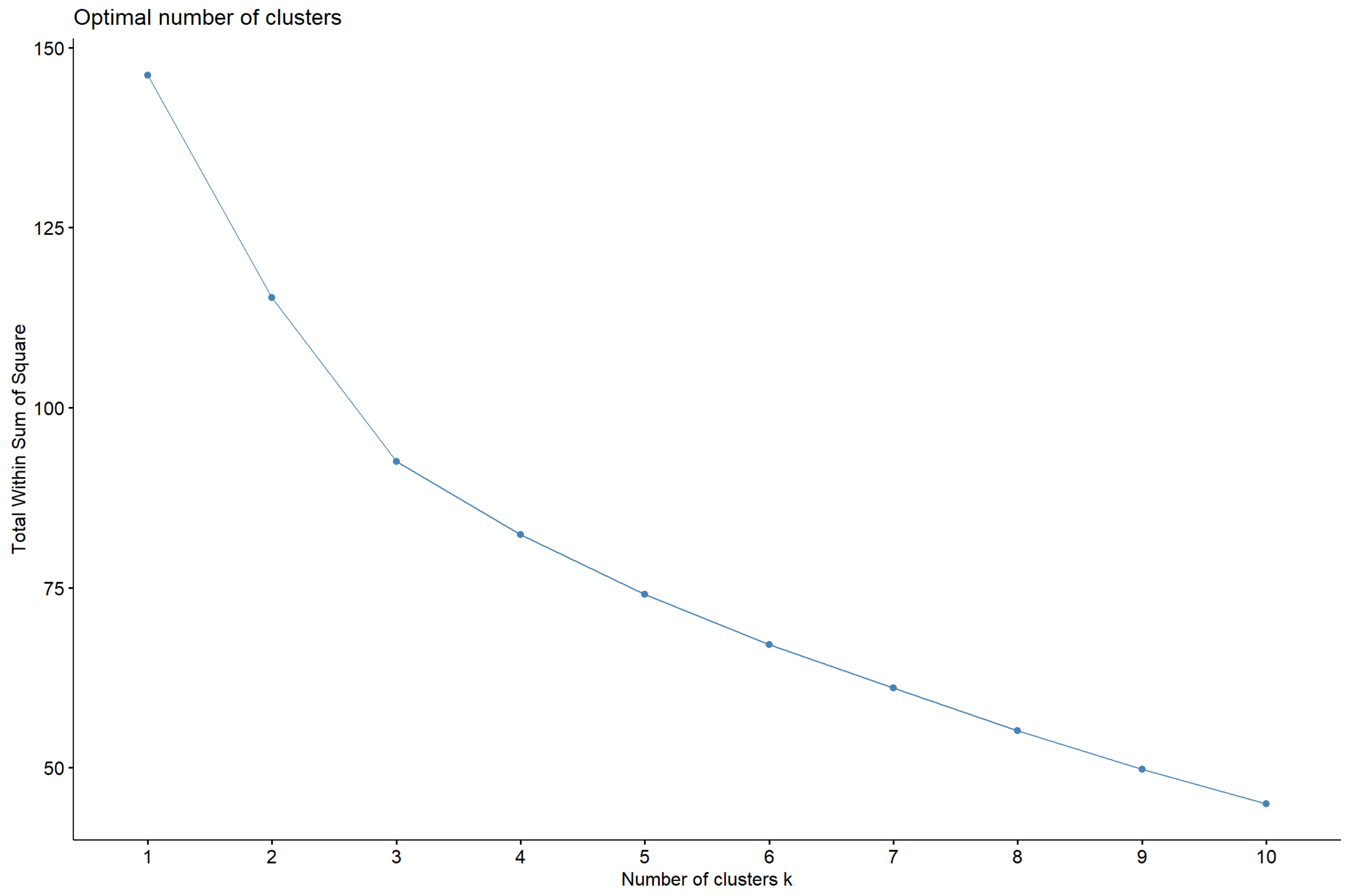
```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
distance <- get_dist(cereals_data[,2:13])
fviz_dist(distance, gradient = list(low= "#00AFBB", mid = "white", high = "#DC4E07"))
```



Determining Optimal Clusters

```
fviz_nbclust(cereals_data, FUN = hcut, method = "wss")
```



- From these estimators, lets assume the optimum K would be 3 We shall evaluate its stability later.

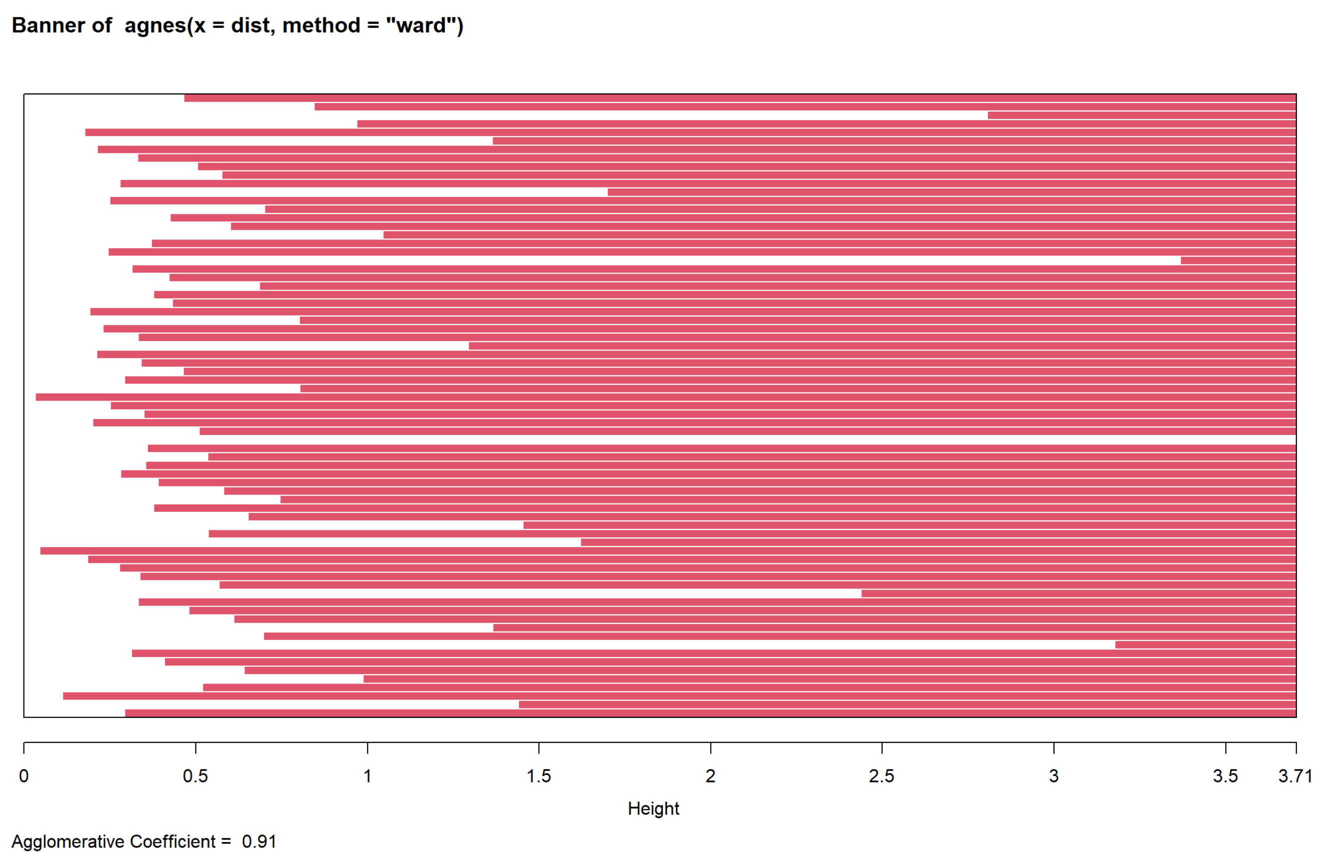
Hierarchical Clustering

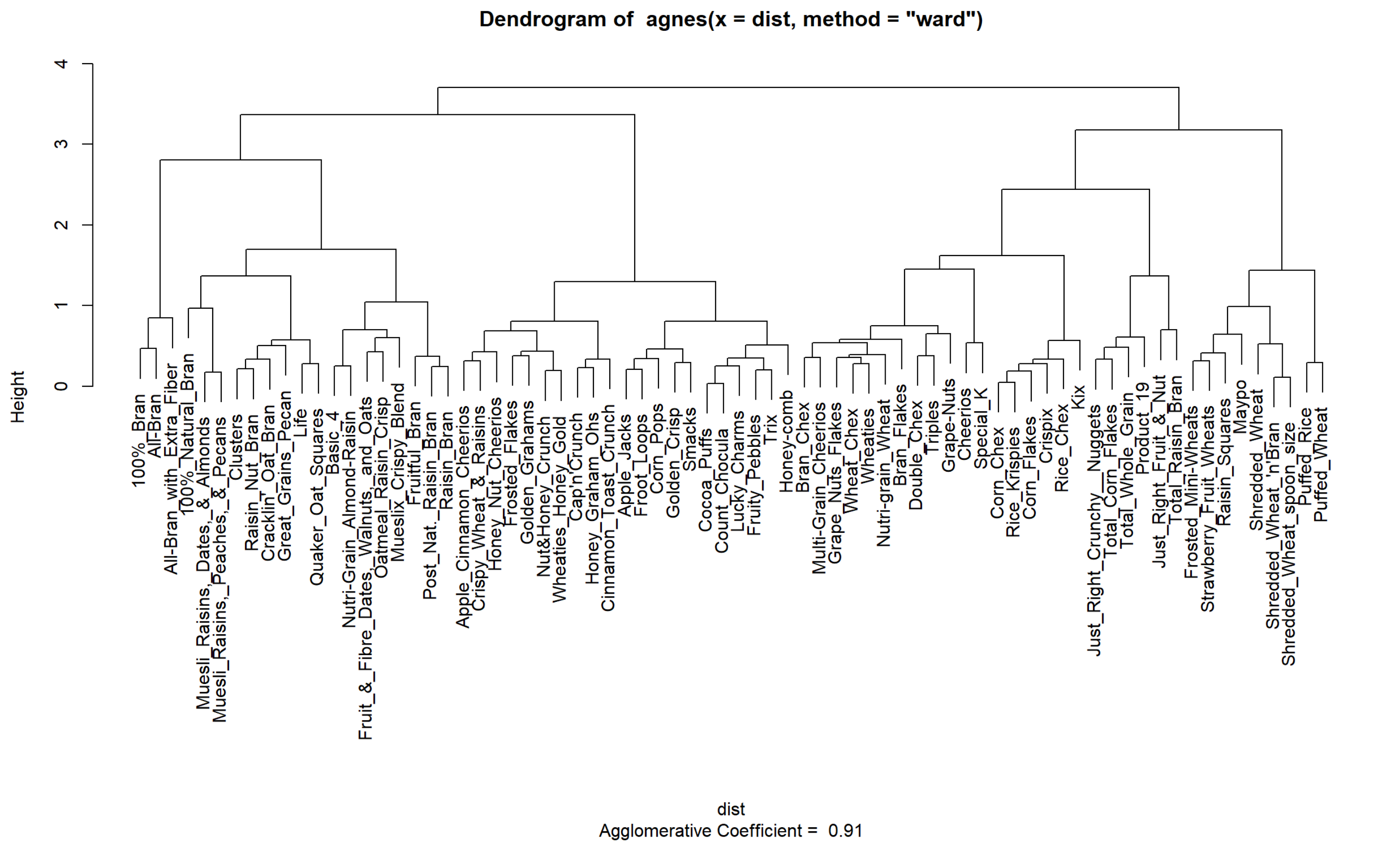
I will use the euclidean distance measure distance.

```
dist <- dist(cereals_data[,1:12], method="euclidean")
```

- hierarchical clustering using ward linkage method.

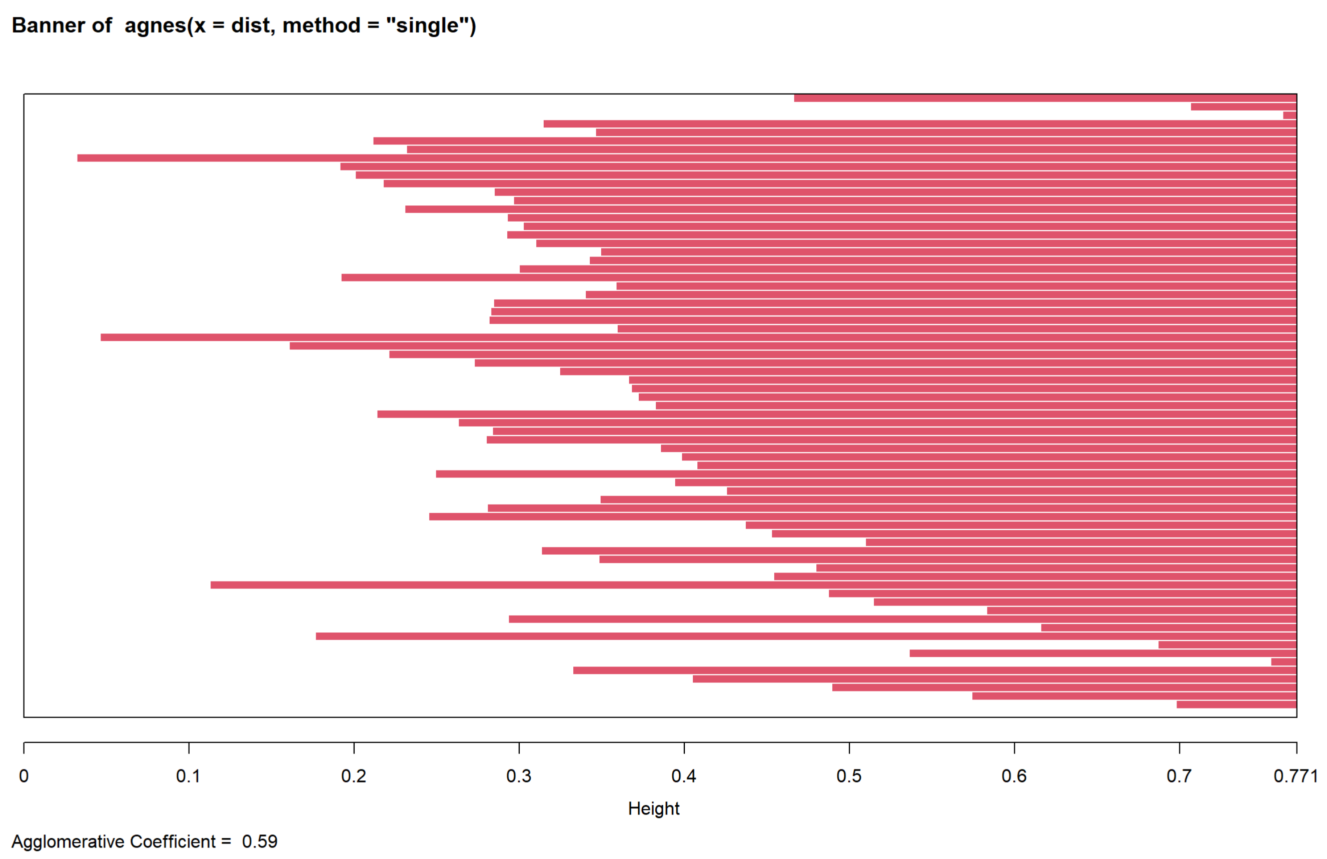
```
library(cluster)
hc_fit_wd <- agnes(dist, method="ward")
plot(hc_fit_wd)
```

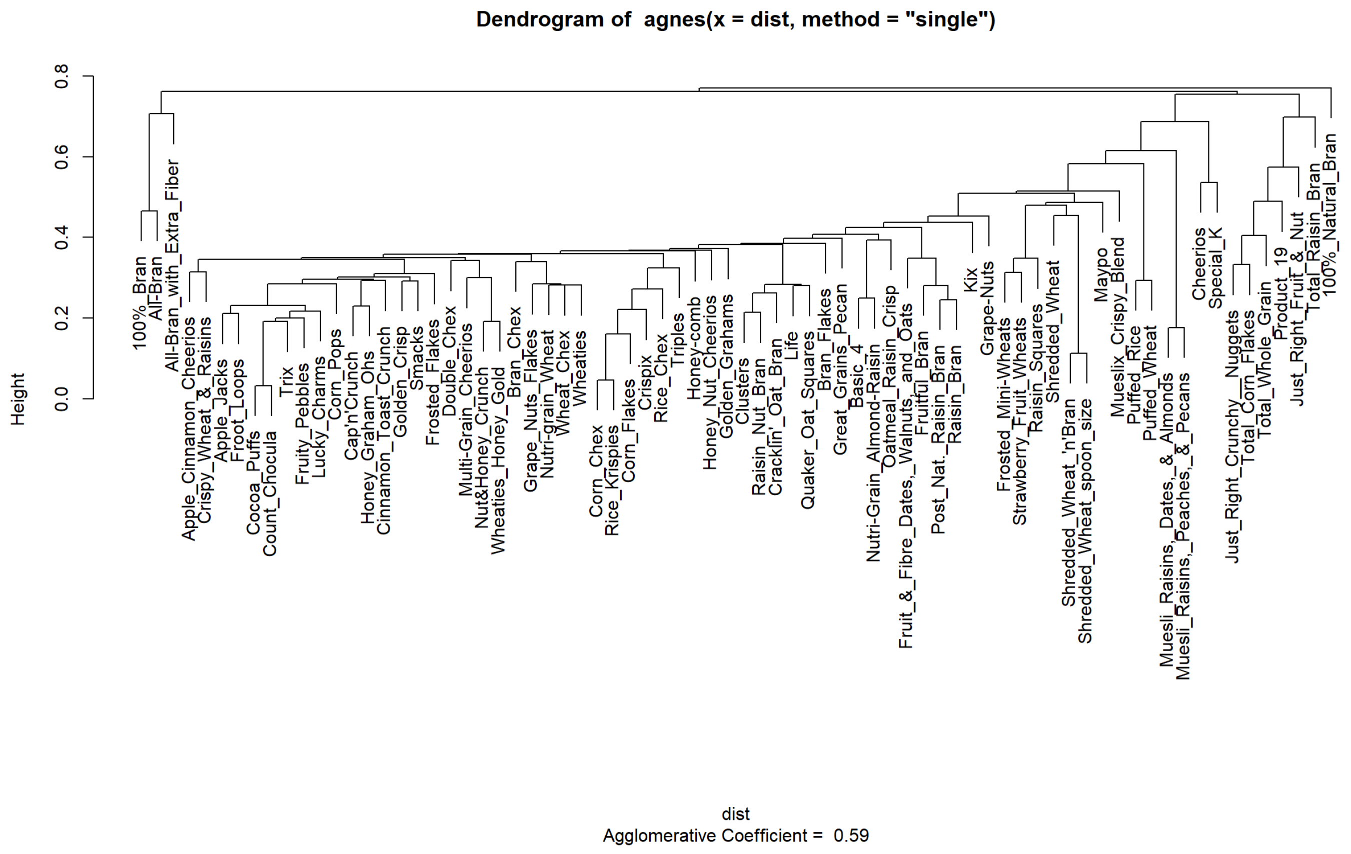




- hierarchical clustering using single linkage method.

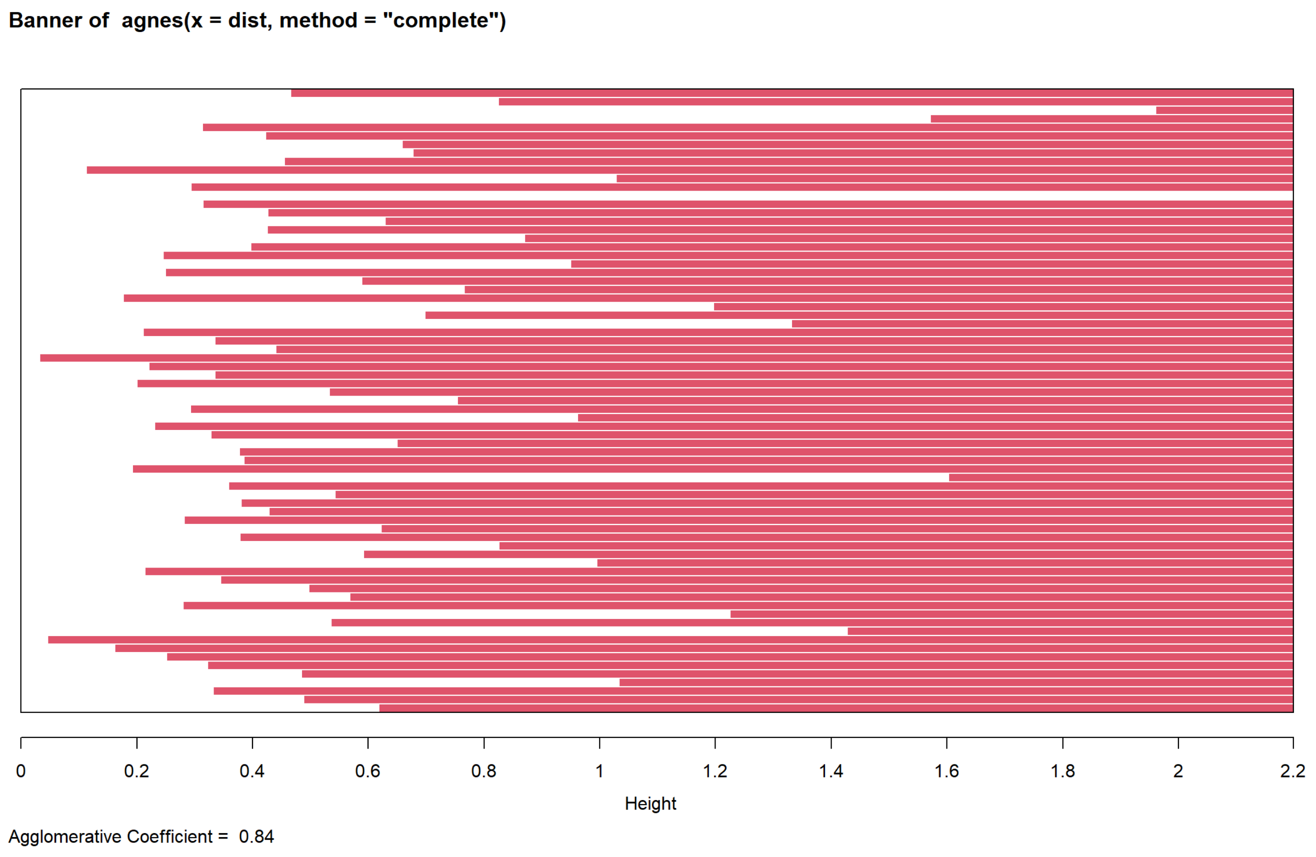
```
hc_fit_sg <- agnes(dist, method="single")
plot(hc_fit_sg)
```

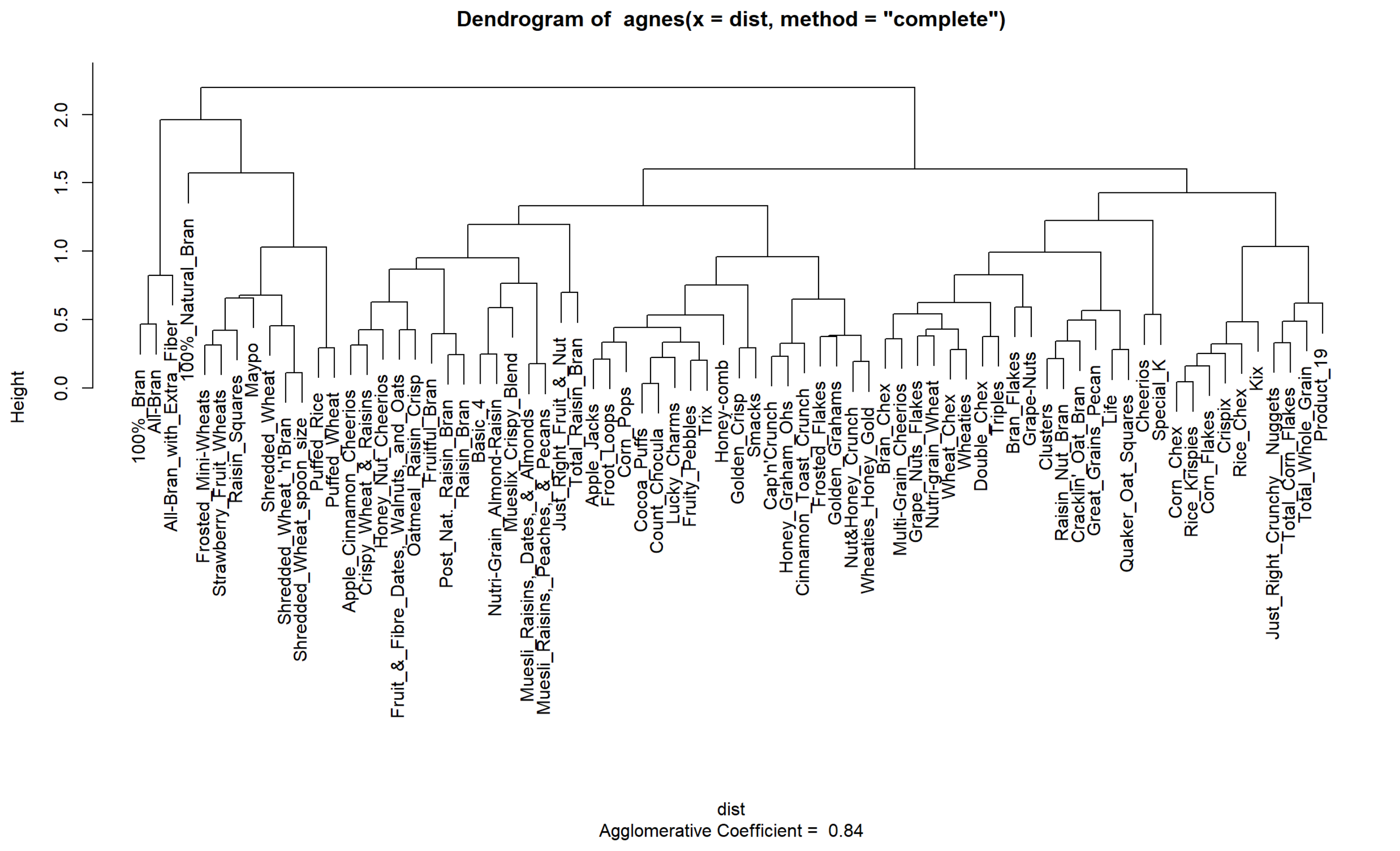




- hierarchical clustering using complete linkage method.

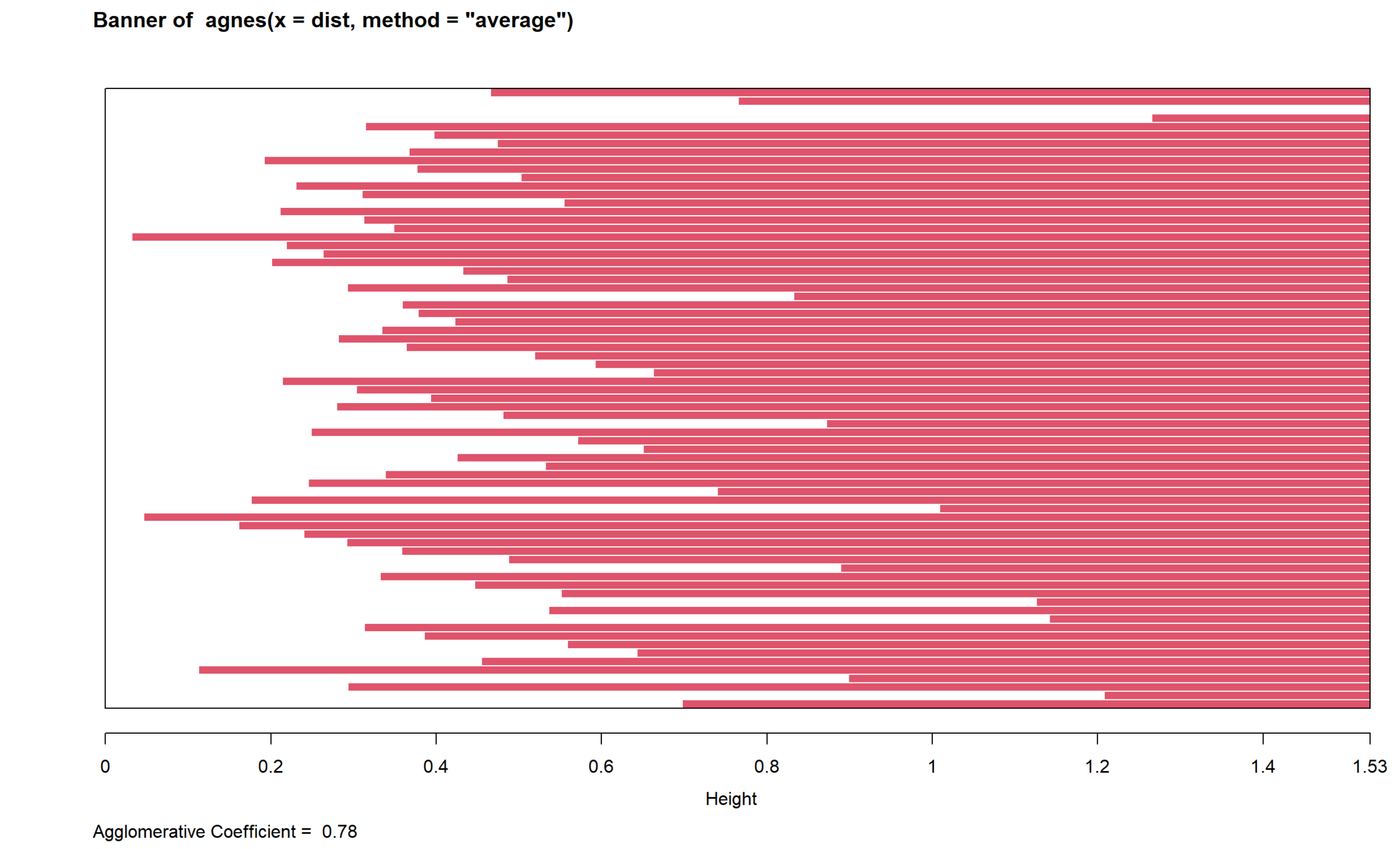
```
hc_fit_cmp <- agnes(dist, method="complete")
plot(hc_fit_cmp)
```

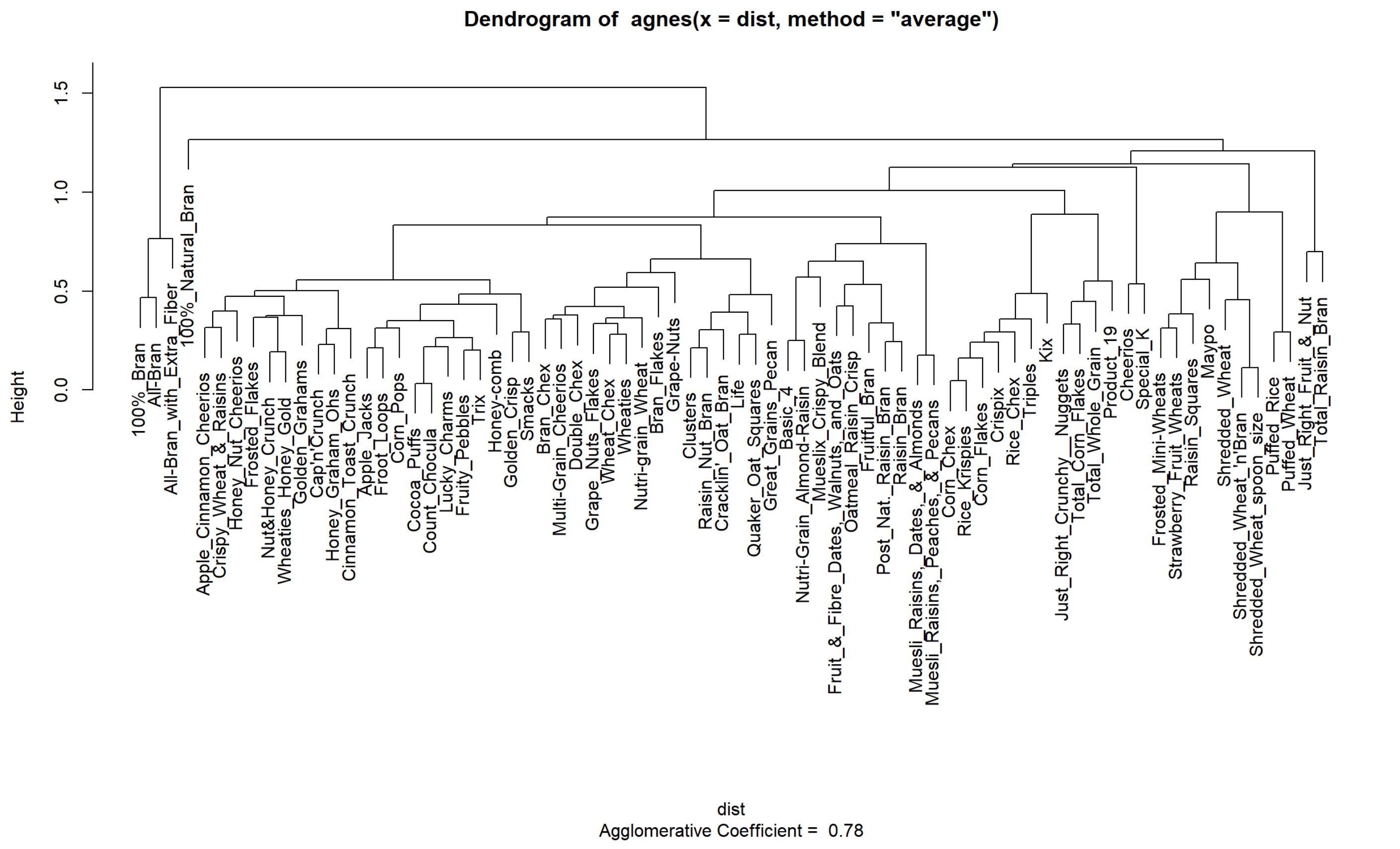




- hierarchical clustering using ward linkage method.

```
hc_fit_avg <- agnes(dist, method="average")
plot(hc_fit_avg)
```





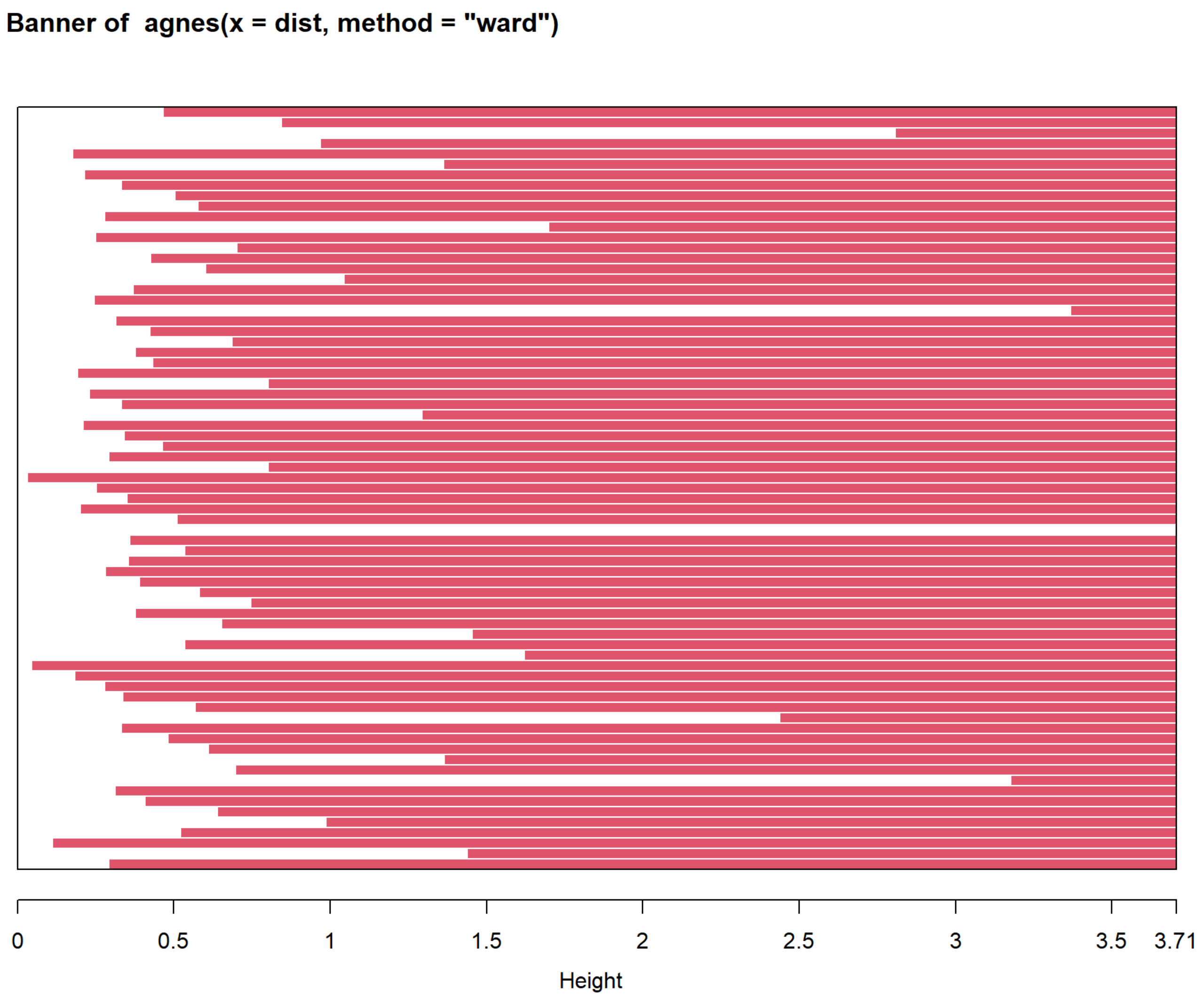
Based on the agglomerative coefficients, “WARD” is the most efficient method to proceed further.

```
points_hc <- cutree(hc_fit_wd, k=3)
cereals_clusts_hc <- cbind(points_hc, cereals_data)

colnames(cereals_clusts_hc)[1] <- "cluster_hc"
head(cereals_clusts_hc)
```

	cluster_hc <int>	**calories** <dbl>	**protein** <dbl>	**fat** <dbl>	**sodium** <dbl>	**fi** <(
100%_Bran	1	-0.33660934	0.2972973	0.0	-0.1011402	0.55888
100%_Natural_Bran	1	0.11793612	0.0972973	0.8	-0.4605152	-0.01254
All-Bran	1	-0.33660934	0.2972973	0.0	0.3051098	0.48745
All-Bran_with_Extra_Fiber	1	-0.51842752	0.2972973	-0.2	-0.0698902	0.84459
Apple_Cinnamon_Cheerios	2	0.02702703	-0.1027027	0.2	0.0551098	-0.04826
Apple_Jacks	2	0.02702703	-0.1027027	-0.2	-0.1167652	-0.08397

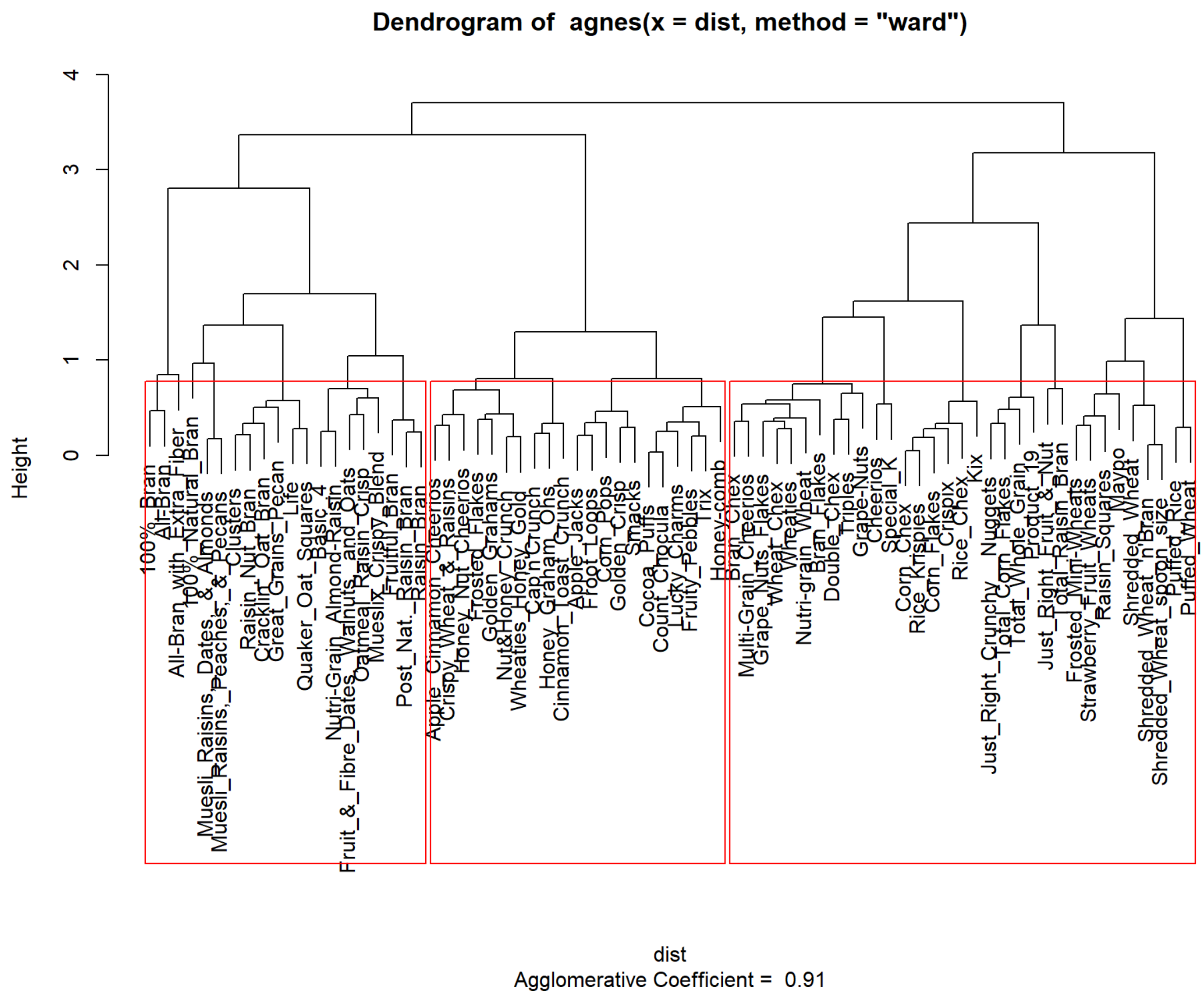
6 rows | 1-7 of 26 columns

```
library(cluster)
plot(hc_fit_wd)
```



Agglomerative Coefficient = 0.91

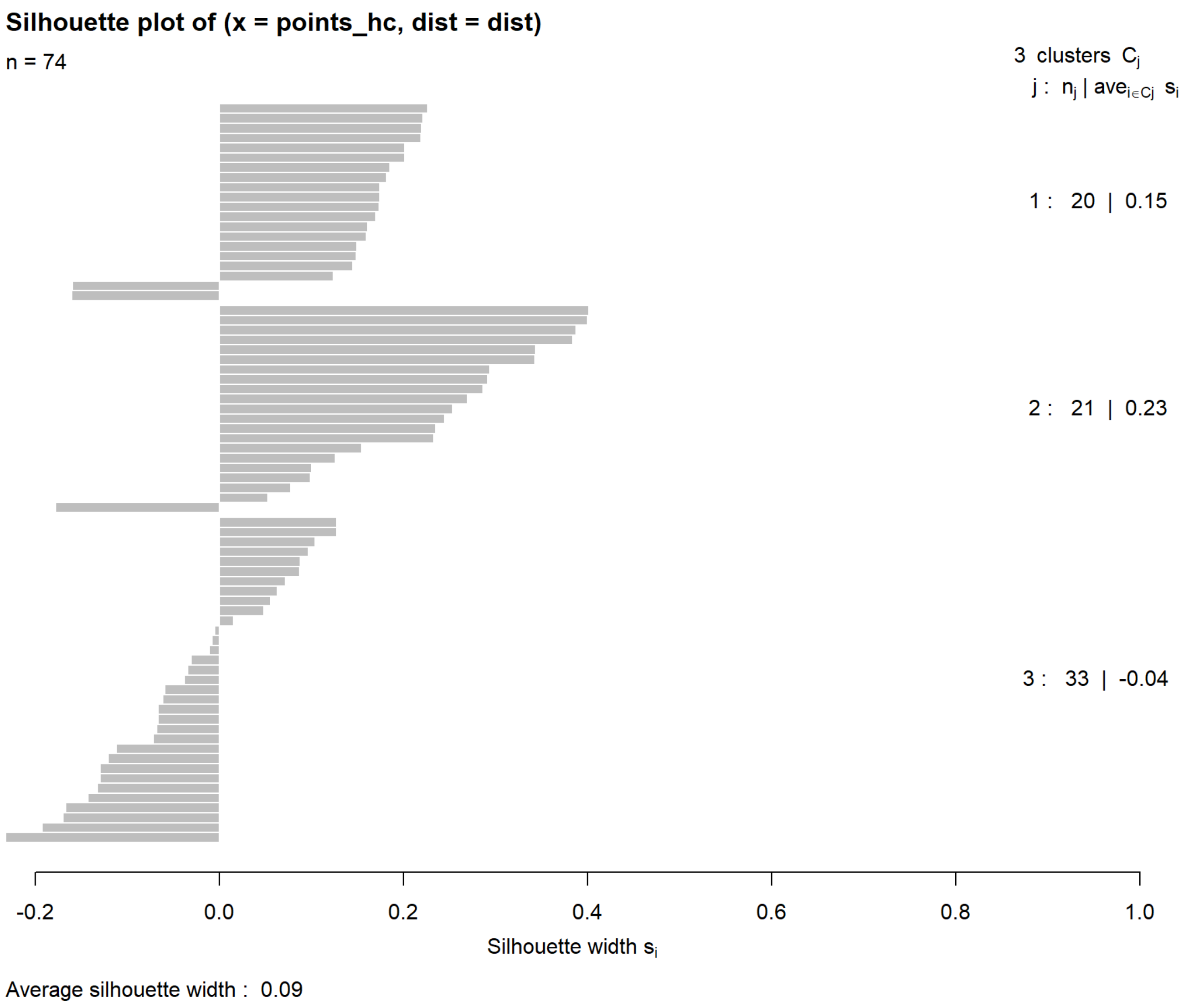
```
rect.hclust(hc_fit_wd, k = 3, border = "red")
```



Checking Quality of clusters Created

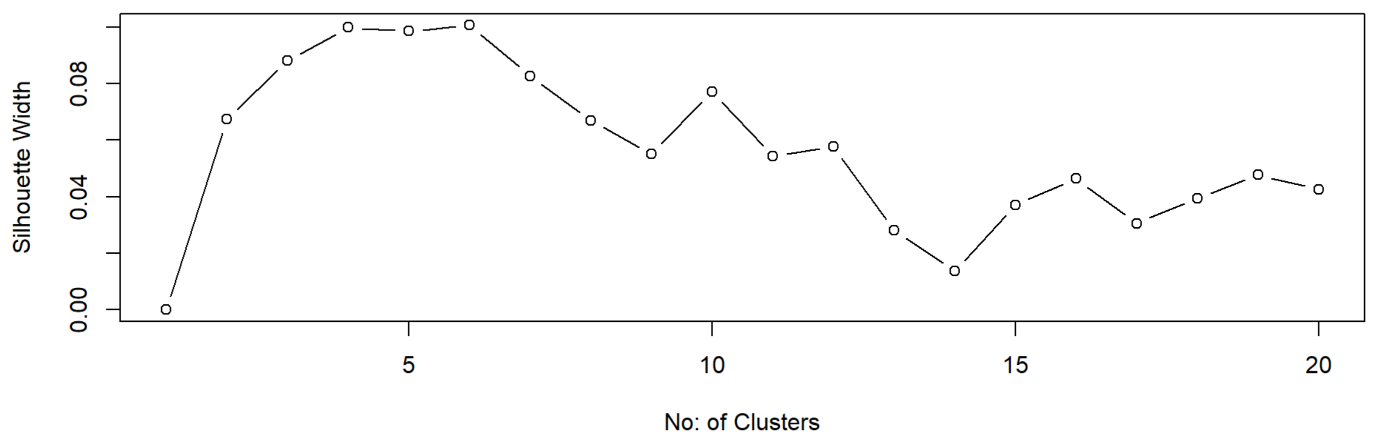
```
- The silhouette width/value is a measure of how similar an object is to its own cluster (cohesi
on) compared to other clusters (separation)  [i.e., intra-cluster cohesion and inter-cluster sep
aration]
- Ranges from -1 to +1
- Values closer to 1 means higher quality of the cluster created
```

```
library(cluster)
dist = daisy(x = cereals_data, metric = "euclidean")
sil_value = silhouette(points_hc, dist = dist)
plot(sil_value)
```



- finding the optimal number of clusters where silhouette width would be maximum

```
sil_value_hc = 0
for (i in 2:20) {
  points_hc <- cutree(hc_fit_wd, k = i)
  sil_value_hc[i] = mean(silhouette(points_hc, dist = dist)[,3])
}
plot(1:20, sil_value_hc, type = "b", xlab = "No: of Clusters", ylab = "Silhouette Width")
```



- According to the Silhoutte, the optimized cluster value is 4 and 6. Lets check the stability of both 4 and 6 clusters with clusterboot method now.

Cluster Stability

Checking for Stability of k=4.

- clusterboot is an integrated function that computes the clustering as well, using interface functions for various clustering methods implemented in R (several interface functions are provided, but you can implement further ones for your favourite clustering method)

- Clusterboot function using library(fpc)

```
library(fpc)
```

```
## Warning: package 'fpc' was built under R version 4.0.5
```

```
#Input the scaled cereals_data
hclust_stability = clusterboot(cereals_data, clustermethod=hclustCBI, method="ward.D2", k=4, cou
nt = FALSE)
```

- What are the cluster stability values? Values > 0.85 denote very stable clusters. 0.6 - 0.75 means the clusters show some patterns but needs to be investigated further

```
#Cluster stability values
hclust_stability$bootmean
```

```
## [1] 0.6541825 0.6004057 0.6193093 0.6900000
```

- How many times the different clusters were dissolved

```
#Cluster dissolution rate. If maximum Jaccard coefficient < 0.5, that cluster is assumed to be d
issolved. Below code shows the number of times each cluster was dissolved. The lower the value,
 the better.
hclust_stability$bootbrd
```

```
## [1] 35 41 36 31
```

Checking for Stability of k=6.

```
hclust_stability = clusterboot(cereals_data, clustermethod=hclustCBI, method="ward.D2", k=6, cou
nt = FALSE)
```

- What are the cluster stability values? Values > 0.85 denote very stable clusters. 0.6 - 0.75 means the clusters show some patterns but needs to be investigated further

```
#Cluster stability values
hclust_stability$bootmean
```

```
## [1] 0.6783651 0.4780906 0.6388820 0.7087018 0.6867587 0.5800000
```

- How many times the different clusters were dissolved

```
#Cluster dissolution rate. If maximum Jaccard coefficient < 0.5, that cluster is assumed to be d
issolved. Below code shows the number of times each cluster was dissolved. The lower the value,
 the better.
hclust_stability$bootbrd
```

```
## [1] 31 71 24 11 19 42
```

Hence, after checking the stability of both posible values of K; the best choice is 4.

- Implementing the the hierarchical clustering with k=4

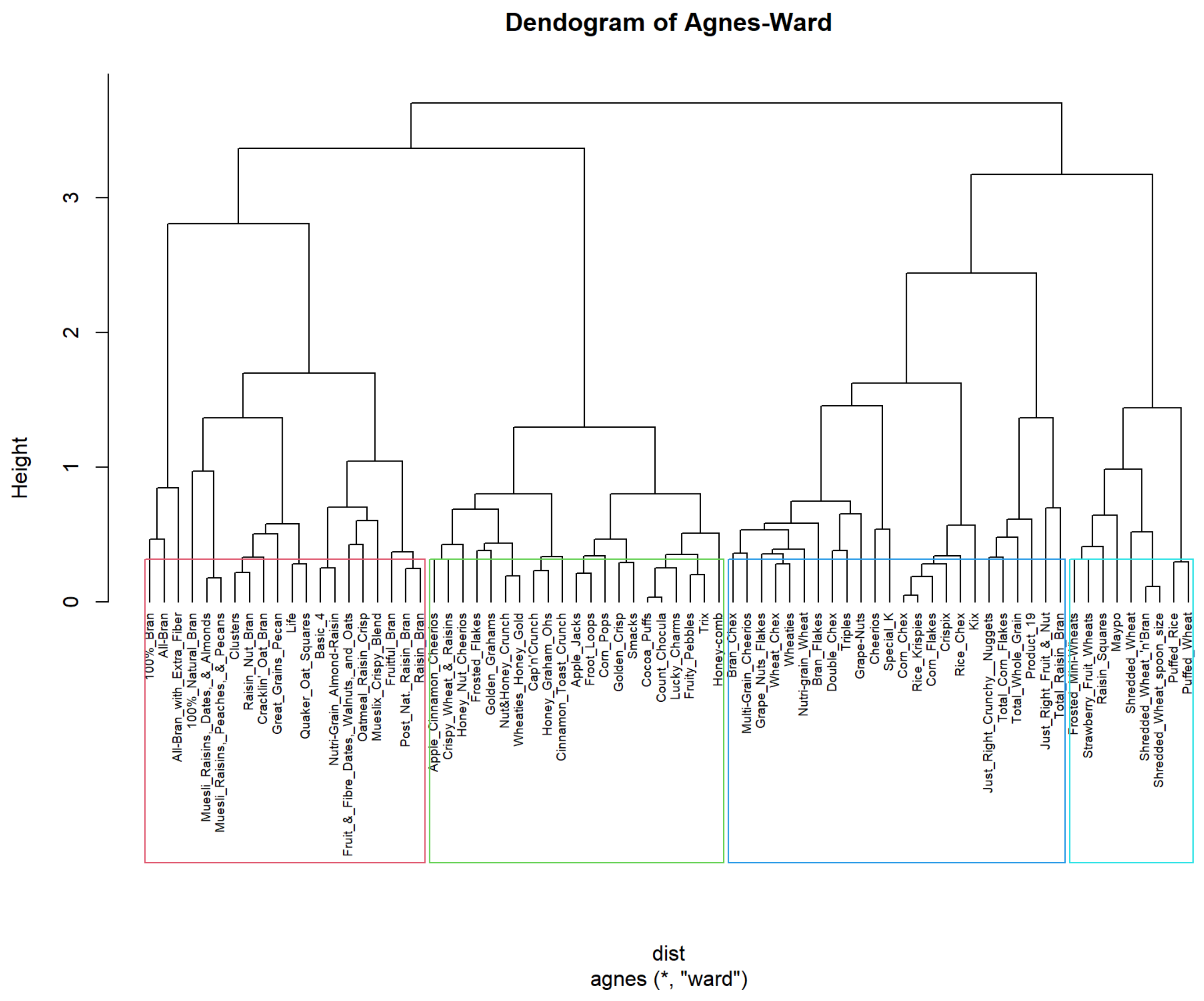
```
points_hc_4 <- cutree(hc_fit_wd, k=4)
cereals_clusts_hc_4 <- cbind(points_hc_4, cereals_data)

colnames(cereals_clusts_hc_4)[1] <- "cluster_hc"
head(cereals_clusts_hc_4)
```

	cluster_hc	**calories**	**protein**	**fat**	**sodium**	**fi**
	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<
100%_Bran	1	-0.33660934	0.2972973	0.0	-0.1011402	0.55888
100%_Natural_Bran	1	0.11793612	0.0972973	0.8	-0.4605152	-0.01254
All-Bran	1	-0.33660934	0.2972973	0.0	0.3051098	0.48745
All-Bran_with_Extra_Fiber	1	-0.51842752	0.2972973	-0.2	-0.0698902	0.84459

	cluster_hc	calories	protein	fat	sodium	fi
	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<
Apple_Cinnamon_Cheerios	2	0.02702703	-0.1027027	0.2	0.0551098	-0.04826
Apple_Jacks	2	0.02702703	-0.1027027	-0.2	-0.1167652	-0.08397

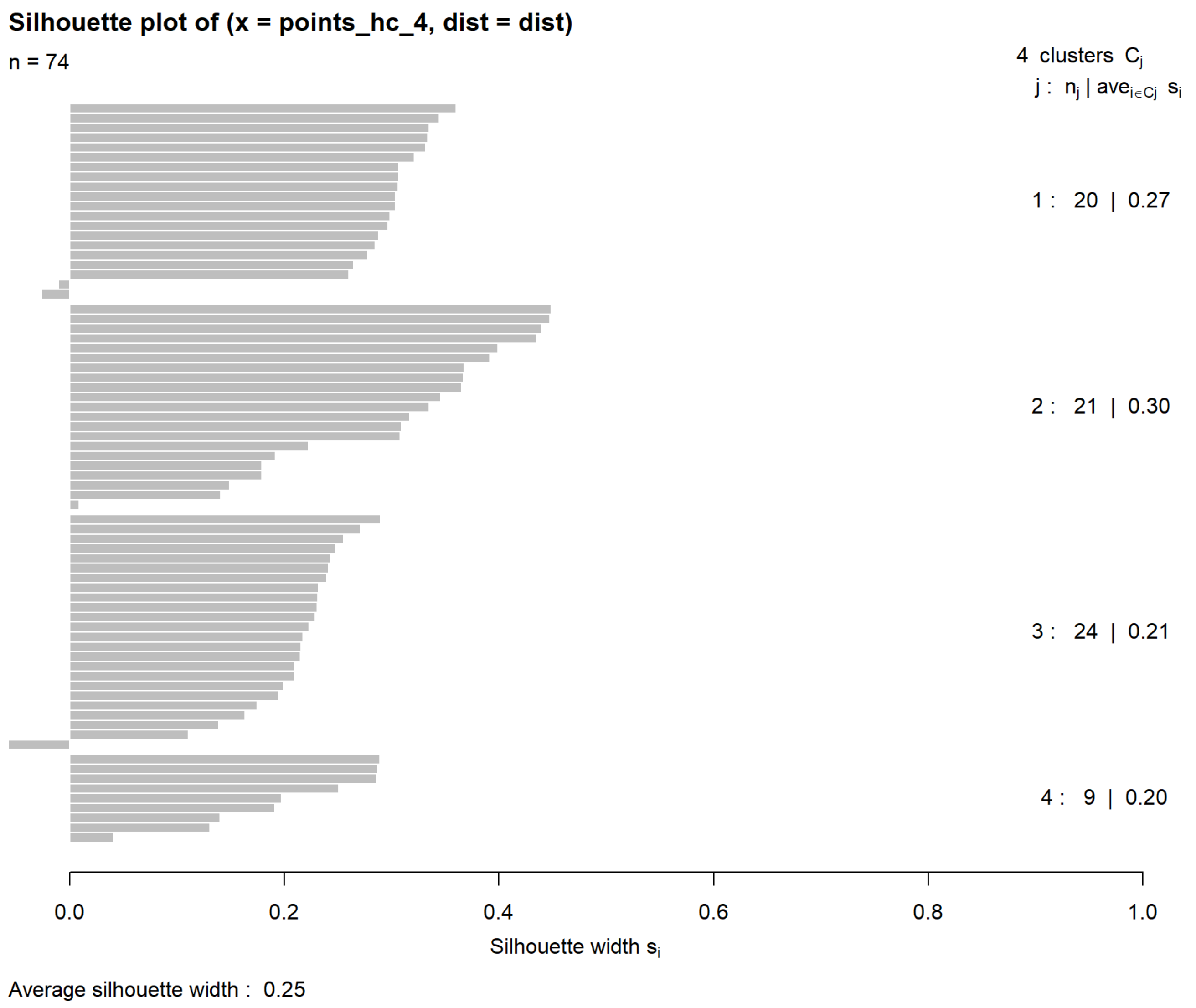
6 rows | 1-7 of 26 columns

```
pltree(hc_fit_wd, cex = 0.6, hang = -1, main = "Dendogram of Agnes-Ward")
rect.hclust(hc_fit_wd, k = 4, border = 2:5)
```



checking Quality of clusters Created

```
library(cluster)
dist = daisy(x = cereals_clusts_hc_4, metric = "euclidean")
sil_value = silhouette(points_hc_4, dist = dist)
plot(sil_value)
```



- A significant improvement in the silhouette width from the case when k was 3.

Selection of the cluster that would be the best cereal for breakfast are based on:

```
- Sodium and Sugar content should be minimal
```

- Cluster 3 has the best options of healthy cereals that students can be served all 5 workdays - so that everyday they can be served with different cereals.

```
library(hrbrthemes)
```

```
## NOTE: Either Arial Narrow or Roboto Condensed fonts are required to use these themes.
```

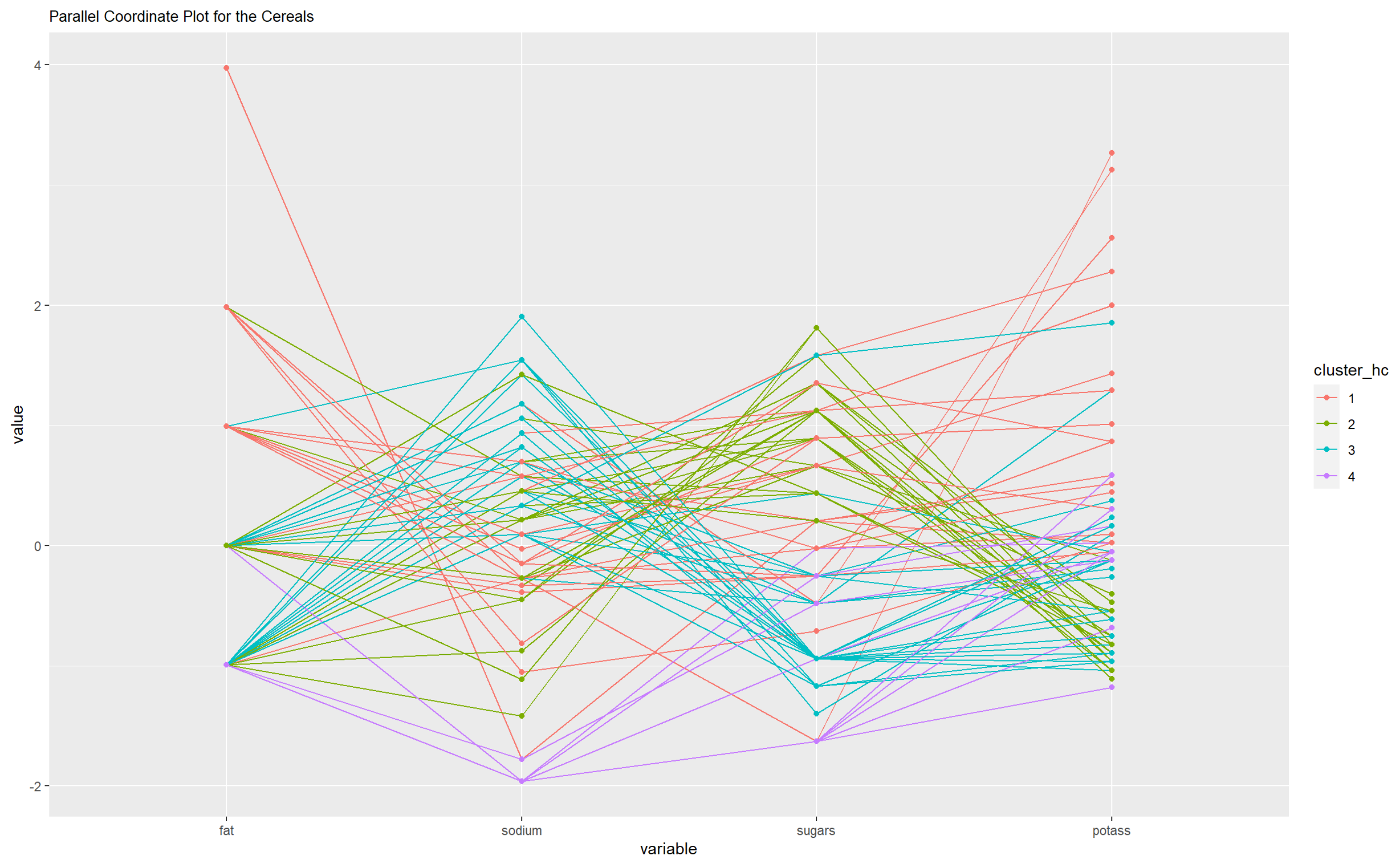
```
##       Please use hrbrthemes::import_roboto_condensed() to install Roboto Condensed and
```

```
##       if Arial Narrow is not on your system, please see https://bit.ly/arialnarrow
```

```
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
cereals_clusts_hc_4$cluster_hc <- as.factor(cereals_clusts_hc_4$cluster_hc)

ggparcoord(cereals_clusts_hc_4,
    columns = c(4,5,8,9), groupColumn = 1,
    showPoints = TRUE,
    title = "Parallel Coordinate Plot for the Cereals",
    alphaLines = 1
    ) + theme(plot.title = element_text(size=10))
```



- Approaching forward with an elimination technique, we will cancel a few parameters to be considered while choosing the best cluster for healthy breakfast.
    - We will consider fat, sodium, sugars and potash for choosing; the cereal having the least nutrition value in these criteria should be eliminated from selection.
    - Cluster 4 seems to be the best choice.