R Notebook

Sean Bradford

```
rm(list=ls())

library(ISLR)
library(cluster)
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 4.2.1

## Loading required package: ggplot2

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

```
library(Rfast)
```

```
## Warning: package 'Rfast' was built under R version 4.2.2

## Loading required package: Rcpp

## Warning: package 'Rcpp' was built under R version 4.2.1

## Loading required package: RcppZiggurat

## Warning: package 'RcppZiggurat' was built under R version 4.2.2
```

```
library(analogue)
```

```
## Warning: package 'analogue' was built under R version 4.2.2

## Loading required package: vegan

## Warning: package 'vegan' was built under R version 4.2.2

## Loading required package: permute

## Warning: package 'permute' was built under R version 4.2.2

## Loading required package: lattice

## This is vegan 2.6-4

## analogue version 0.17-6
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.2.1

## Registered S3 methods overwritten by 'pROC':
##   method    from
##   print.roc analogue
##   plot.roc  analogue

##
## Attaching package: 'caret'

## The following object is masked from 'package:vegan':
##
##     tolerance
```

```
cereal=read.csv("C:\\Users\\Sean\\OneDrive\\Desktop\\Grad School\\Machine Learning\\Module 8 - Hierarchi

rownames(cereal)=cereal$name

cereal=cereal[,-1]

head(cereal)
```

```
##                           mfr type calories protein fat sodium fiber carbo
## 100%_Bran                   N    C       70       4   1    130  10.0   5.0
## 100%_Natural_Bran           Q    C      120       3   5     15   2.0   8.0
## All-Bran                    K    C       70       4   1    260   9.0   7.0
## All-Bran_with_Extra_Fiber   K    C       50       4   0    140  14.0   8.0
## Almond_Delight              R    C      110       2   2    200   1.0  14.0
## Apple_Cinnamon_Cheerios     G    C      110       2   2    180   1.5  10.5
##                           sugars potass vitamins shelf weight cups   rating
## 100%_Bran                      6    280       25     3      1 0.33 68.40297
## 100%_Natural_Bran              8    135        0     3      1 1.00 33.98368
## All-Bran                       5    320       25     3      1 0.33 59.42551
## All-Bran_with_Extra_Fiber      0    330       25     3      1 0.50 93.70491
## Almond_Delight                 8     NA       25     3      1 0.75 34.38484
## Apple_Cinnamon_Cheerios       10     70       25     1      1 0.75 29.50954
```

```
# columns 1,2,12 are categorical and need to be removed before normalization
norm_cereal=scale(cereal[,c(-1,-2,-12)])

# Removing N/A values from data
norm_cereal=as.data.frame(na.omit(norm_cereal))
```

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.

```r
single=agnes(norm_cereal,method="single")
complete=agnes(norm_cereal,method="complete")
average=agnes(norm_cereal,method="average")
ward=agnes(norm_cereal,method="ward")

single$ac
```

```
## [1] 0.6091225
```

```r
complete$ac
```

```
## [1] 0.8508357
```

```r
average$ac
```

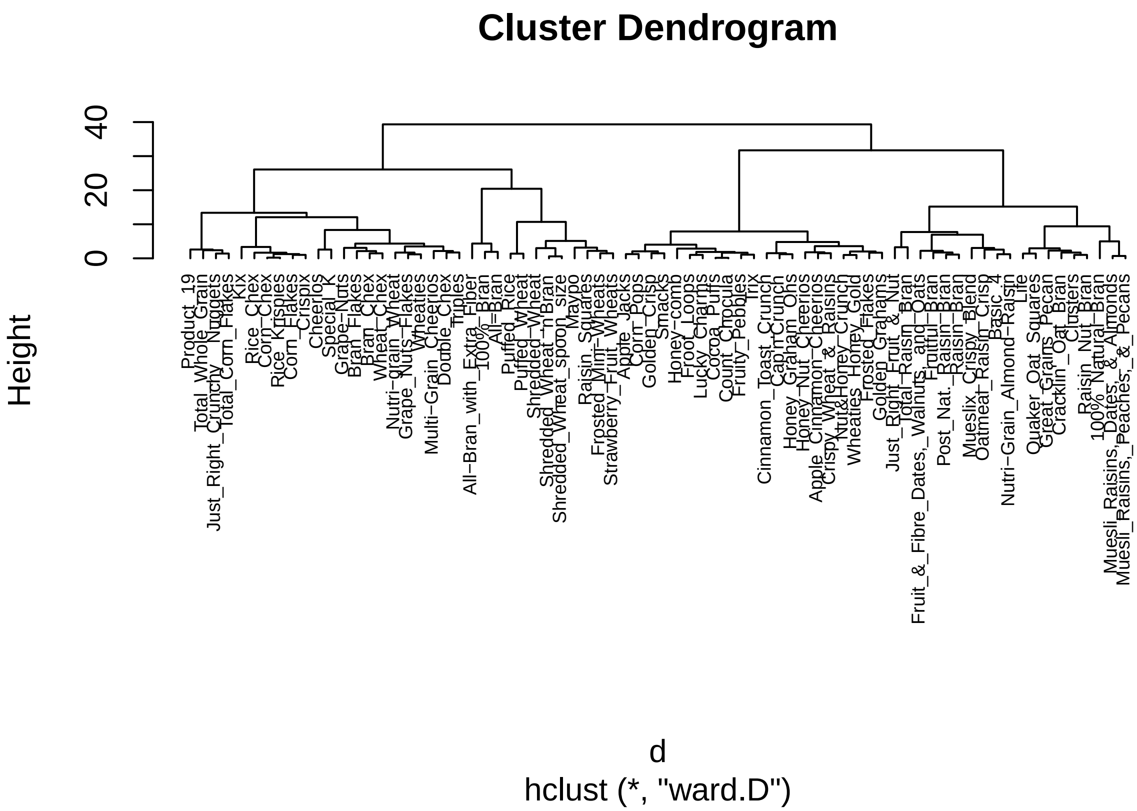
```
## [1] 0.7888569
```

```r
ward$ac
```

```
## [1] 0.9088247
```

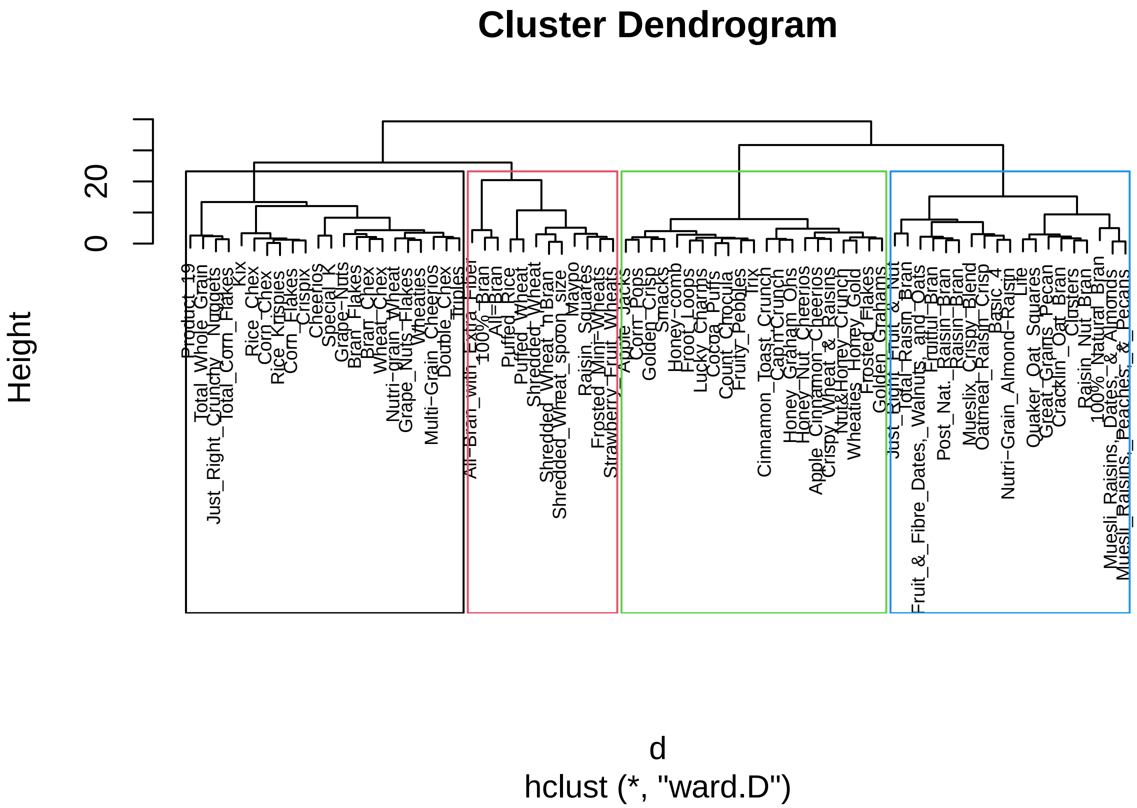
Ward is the best linkage method because it has the highest agglomerative coefficient.

```r
d=dist(norm_cereal,method="euclidean")
d_ward=hclust(d,method="ward.D")
plot(d_ward,cex=0.6,hang=-1)
```

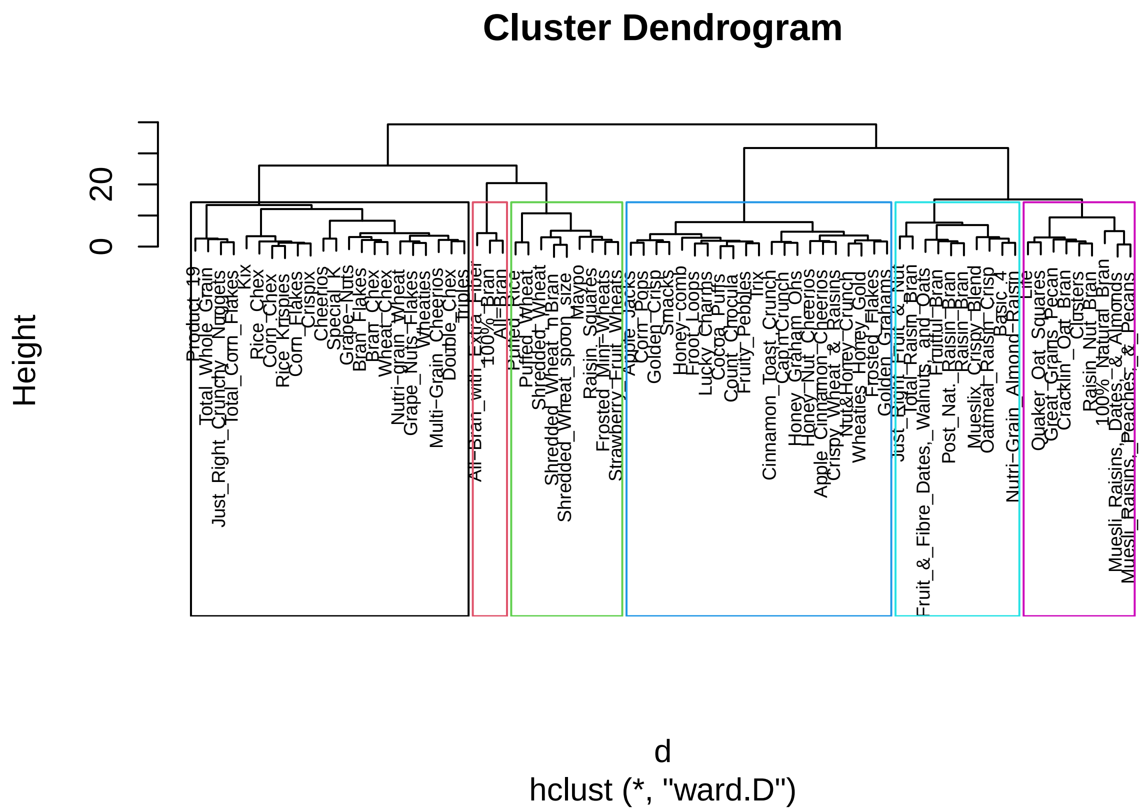


2. How many clusters would you choose?

```r
# Testing for best k value

plot(d_ward,cex=0.6)
rect.hclust(d_ward,k=4,border=1:4)
```




```r
plot(d_ward,cex=0.6)
rect.hclust(d_ward,k=6,border=1:6)
```




```r
k4=cutree(d_ward,k=4)
table(k4)
```

```
## k4
##  1  2  3  4
## 12 19 21 22
```

```r
clustered.data=cbind.data.frame(norm_cereal,k4)
```

K = 4 appears to be the optimal value for clustering

3. Comment on the structure of the clusters and on their stability. Hint: 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.

```r
# Cluster partition A
nrow(norm_cereal)
```

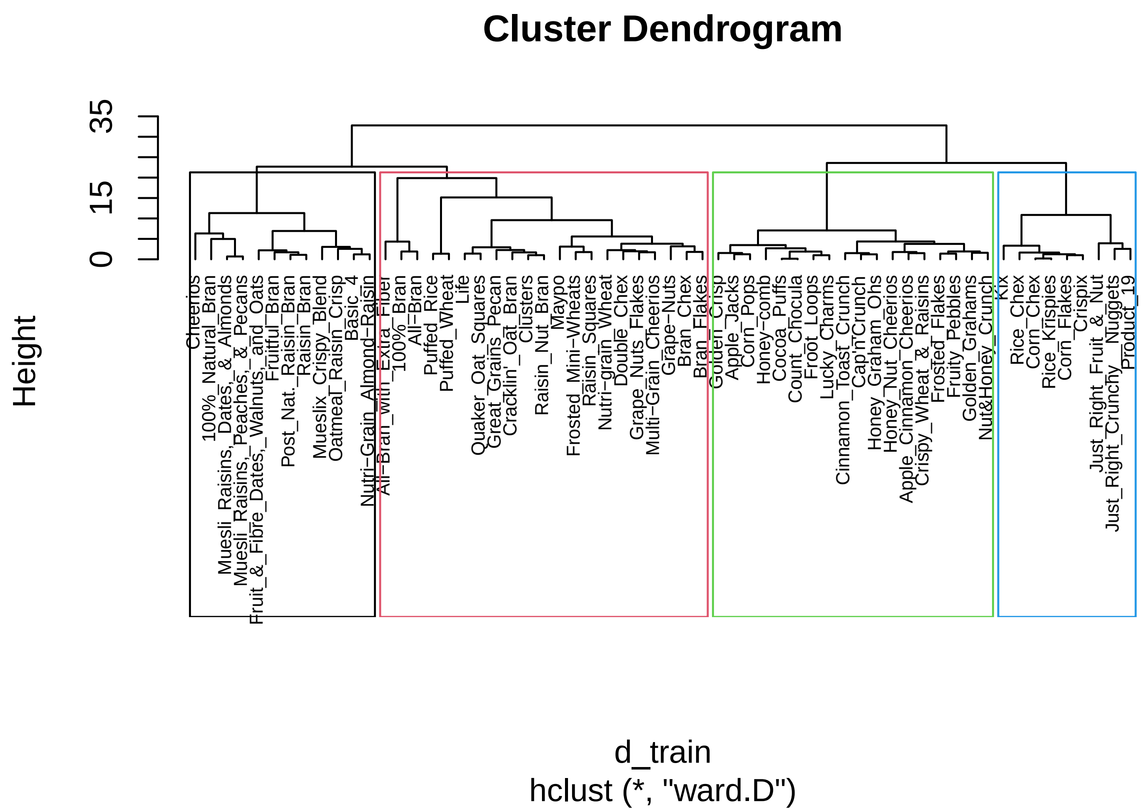
```
## [1] 74
```

```r
# Partitioning 80/20
74*0.8
```

```
## [1] 59.2
```

```r
train=norm_cereal[1:60,]
test=norm_cereal[61:74,]
```

```r
# Use cluster centroids from A to assign each record in partition B
d_train=dist(train,method="euclidean")
d_ward_train=hclust(d_train,method="ward.D")

plot(d_ward_train,cex=0.6,hang=-1)
rect.hclust(d_ward_train,k=4,border=1:4)
```




```r
k4.train=cutree(d_ward_train,k=4)
table(k4.train)
```

```
## k4.train
##  1  2  3  4
## 21 12 18  9
```

```r
train2=cbind.data.frame(train,k4.train)
```

```r
c.1=colMeans(train2[train2$k4.train == "1",])
c.2=colMeans(train2[train2$k4.train=="2",])
c.3=colMeans(train2[train2$k4.train=="3",])
c.4=colMeans(train2[train2$k4.train=="4",])

centroid=rbind(c.1,c.2,c.3,c.4)
test.data.centroid=rowMins(distance(test,centroid[,-13]))
partition.centoid=c(train2$k4.train,test.data.centroid)
clustered.data=cbind(clustered.data,partition.centoid)
```

```r
# Assess how consistent the cluster assignments are compared to the assignments based on all the data.
table(clustered.data$k4==clustered.data$partition.centoid)
```

```
##
## FALSE  TRUE
##    17    57
```

```r
table(clustered.data$k4[61:74]==clustered.data$partition.centoid[61:74])
```

```
##
## FALSE  TRUE
##     2    12
```

```r
(57/74)*100
```

```
## [1] 77.02703
```

```r
(12/14)*100
```

```
## [1] 85.71429
```

Cluster assignments based on test data are 85.71% consistent, and the cluster assignments based on all data are 77.03% consistent.

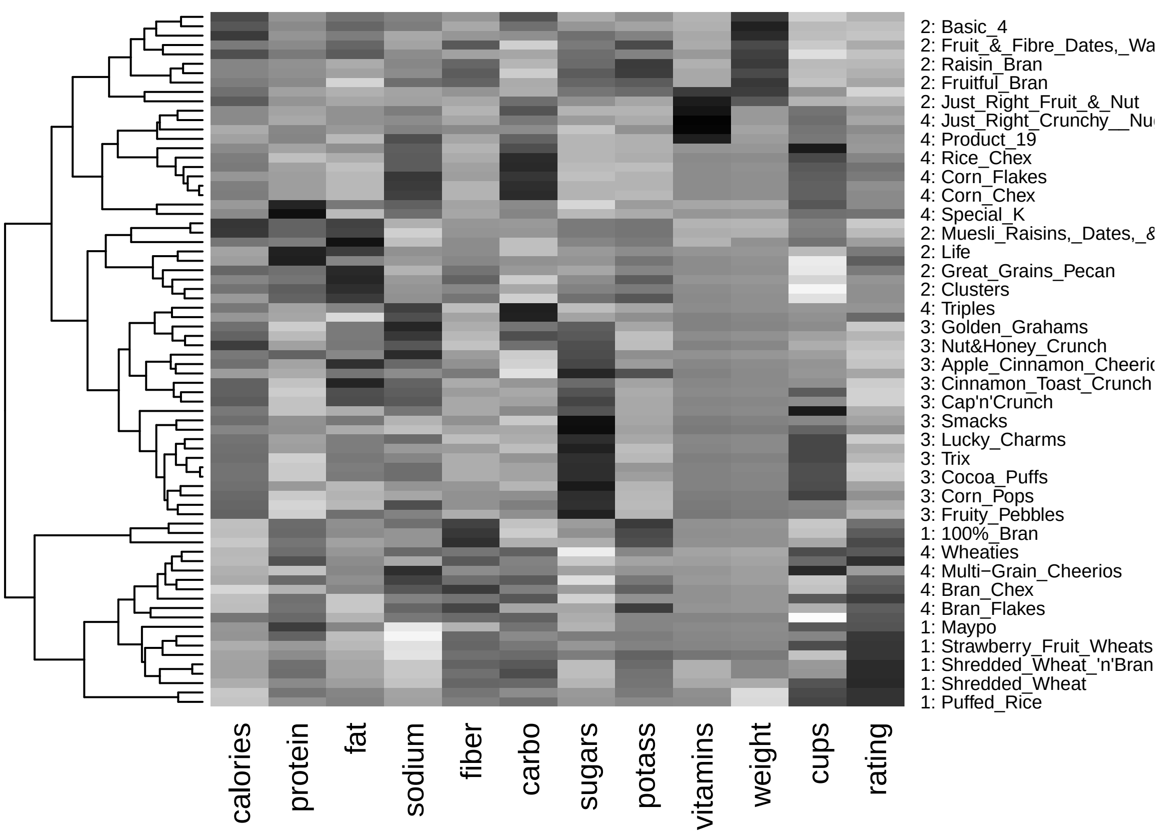
4. The elementary public schools would like to choose a set of cereals to include in their daily cafeterias. Every day a different cereal is offered, but all cereals should support a healthy diet. For this goal, you are requested to find a cluster of "healthy cereals." Should the data be normalized?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

```r
# Calculate all centroids

ctroid1=colMeans(clustered.data[clustered.data$k4 == "1",])
ctroid2=colMeans(clustered.data[clustered.data$k4 == "2",])
ctroid3=colMeans(clustered.data[clustered.data$k4 == "3",])
ctroid4=colMeans(clustered.data[clustered.data$k4 == "4",])
ctroid.bind=rbind(ctroid1, ctroid2, ctroid3, ctroid4)
```

```r
# View avg nutrient values across clusters

head(ctroid.bind)
```

```
##          calories   protein         fat      sodium      fiber      carbo
## ctroid1 -1.5080547  0.2629535 -0.75808029 -1.36294322  0.9152548 -0.4186917
## ctroid2  0.9433345  0.6074881  0.98066557 -0.04635267  0.4220701 -0.1784772
## ctroid3  0.2088503 -0.9331883 -0.01290349  0.10611786 -0.6631465 -0.6075921
## ctroid4 -0.1666359  0.1245569 -0.46452580  0.79007459 -0.1972548  0.8997331
##            sugars     potass   vitamins     weight       cups     rating k4
## ctroid1 -0.9956591  0.6639622 -0.6115433 -0.8447175 -0.3768781  1.6415416  1
## ctroid2  0.5228713  0.7739062  0.1491414  1.0099135 -0.5653466 -0.3057055  2
## ctroid3  1.0162649 -0.7283802 -0.1453172 -0.1967771  0.2328980 -0.9969602  3
## ctroid4 -0.8157229 -0.3425791  0.4650151 -0.1967771  0.4799337  0.2498970  4
##         partition.centoid
## ctroid1          1.000000
## ctroid2          1.789474
## ctroid3          3.000000
## ctroid4          2.681818
```

```r
# Create heatmap to further compare cluster values

row.names(norm_cereal)=paste(k4,": ",row.names(norm_cereal),sep="")

heatmap(as.matrix(norm_cereal),Colv=NA,hclustfun=hclust,col=rev(paste("gray",1:99,sep="")))
```



The heatmap and the table "ctroid.bind" indicate that cluster 1 would be the best choice for elementary public schools. Compared to the other clusters, cluster 1 has the highest rating & fiber, second-highest protein & potassium, and the lowest calories, fat, sodium and sugar. These factors make it the healthiest choice.

The 12 cereals shown below makeup cluster 1 and would become part of the school's breakfast offering.

```
clustered.data[clustered.data$k4 == '1',]
```

```
##                            calories   protein         fat     sodium
## 100%_Bran                 -1.8929836  1.3286071 -0.01290349 -0.3539844
## All-Bran                  -1.8929836  1.3286071 -0.01290349  1.1967306
## All-Bran_with_Extra_Fiber -2.9194605  1.3286071 -1.00647256 -0.2346986
## Frosted_Mini-Wheats       -0.3532681  0.4151897 -1.00647256 -1.9046994
## Maypo                     -0.3532681  1.3286071 -0.01290349 -1.9046994
## Puffed_Rice               -2.9194605 -1.4116451 -1.00647256 -1.9046994
## Puffed_Wheat              -2.9194605 -0.4982277 -1.00647256 -1.9046994
## Raisin_Squares            -0.8665066 -0.4982277 -1.00647256 -1.9046994
## Shredded_Wheat            -1.3797451 -0.4982277 -1.00647256 -1.9046994
## Shredded_Wheat_'n'Bran    -0.8665066  0.4151897 -1.00647256 -1.9046994
## Shredded_Wheat_spoon_size -0.8665066  0.4151897 -1.00647256 -1.9046994
## Strawberry_Fruit_Wheats   -0.8665066 -0.4982277 -1.00647256 -1.7257708
##                                 fiber       carbo       sugars      potass
## 100%_Bran                  3.29284661 -2.50878291 -0.234390576  2.57536849
## All-Bran                   2.87327158 -1.99692385 -0.462771138  3.14346448
## All-Bran_with_Extra_Fiber  4.97114672 -1.74099432 -1.604673946  3.28548848
## Frosted_Mini-Wheats        0.35582142 -0.20541712 -0.006010015  0.01893653
## Maypo                     -0.90290366  0.30644194 -0.919532261 -0.05207547
## Puffed_Rice               -0.90290366 -0.46134666 -1.604673946 -1.18826745
## Puffed_Wheat              -0.48332864 -1.22913525 -1.604673946 -0.69118346
## Raisin_Squares            -0.06375361  0.05051241 -0.234390576  0.16096053
## Shredded_Wheat             0.35582142  0.30644194 -1.604673946 -0.05207547
## Shredded_Wheat_'n'Bran     0.77539645  1.07423054 -1.604673946  0.58703252
## Shredded_Wheat_spoon_size  0.35582142  1.33016007 -1.604673946  0.30298453
## Strawberry_Fruit_Wheats    0.35582142  0.05051241 -0.462771138 -0.12308746
##                            vitamins     weight        cups    rating k4
## 100%_Bran                 -0.1453172 -0.1967771 -2.11003399 1.8321876  1
## All-Bran                  -0.1453172 -0.1967771 -2.11003399 1.1930986  1
## All-Bran_with_Extra_Fiber -0.1453172 -0.1967771 -1.37953029 3.6333849  1
## Frosted_Mini-Wheats       -0.1453172 -0.1967771 -0.09040611 1.1161895  1
## Maypo                     -0.1453172 -0.1967771  0.76901001 0.8674423  1
## Puffed_Rice               -1.2642598 -3.5195485  0.76901001 1.2878220  1
## Puffed_Wheat              -1.2642598 -3.5195485  0.76901001 1.4479620  1
## Raisin_Squares            -0.1453172 -0.1967771 -1.37953029 0.9017710  1
## Shredded_Wheat            -1.2642598 -1.3265194  0.76901001 1.8202929  1
## Shredded_Wheat_'n'Bran    -1.2642598 -0.1967771 -0.64902659 2.2642977  1
## Shredded_Wheat_spoon_size -1.2642598 -0.1967771 -0.64902659 2.1453309  1
## Strawberry_Fruit_Wheats   -0.1453172 -0.1967771  0.76901001 1.1887196  1
##                           partition.centoid
## 100%_Bran                                 1
## All-Bran                                  1
## All-Bran_with_Extra_Fiber                 1
## Frosted_Mini-Wheats                       1
## Maypo                                     1
```

```
## Puffed_Rice                                  1
## Puffed_Wheat                                 1
## Raisin_Squares                               1
## Shredded_Wheat                               1
## Shredded_Wheat_'n'Bran                       1
## Shredded_Wheat_spoon_size                    1
## Strawberry_Fruit_Wheats                      1
```