R Notebook

Code

Hide

```
##install.packages("tidyverse")
library(tidyverse)
library(cluster)    # clustering algorithms
##install.packages("factoextra")
library(factoextra) # clustering visualization
##install.packages("ggplot2")
library(dendextend) # for comparing two dendrograms
##install.packages("sparcl")
library(sparcl)     #to create colourDendograms
```

Hide

```
Csdata<- read.csv("Cereals.csv")
```

Hide

```
Csdata1<- na.omit(Csdata) ##remove the missing value
```

Hide

```
Csdata2<-Csdata1[,c(-2,-3)] ## excluding catagorical variable
head(Csdata2)
```

	name	calories	protein	f…	sodi…	fiber	ca…	sug…	pota.
	<fctr>	<int>	<int>	<int>	<int>	<dbl>	<dbl>	<int>	<in
1	100%_Bran	70	4	1	130	10.0	5.0	6	28
2	100%_Natural_Bran	120	3	5	15	2.0	8.0	8	13
3	All-Bran	70	4	1	260	9.0	7.0	5	32
4	All-Bran_with_Extra_Fiber	50	4	0	140	14.0	8.0	0	33
6	Apple_Cinnamon_Cheerios	110	2	2	180	1.5	10.5	10	7
7	Apple_Jacks	110	2	0	125	1.0	11.0	14	3

6 rows | 1-10 of 14 columns

Hide

```
str(Csdata2)
```

```
'data.frame':   74 obs. of  14 variables:
 $ name    : Factor w/ 77 levels "100%_Bran","100%_Natural_Bran",..: 1 2 3 4 6 7 8 9 1
0 11 ...
 $ 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 ...
 $ shelf   : int  3 3 3 3 1 2 3 1 3 2 ...
 $ 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 ...
```

Hide

```
summary(Csdata2)
```

```
                      name       calories      protein           fat        sodium
fiber
 100%_Bran               : 1   Min.   : 50   Min.   :1.000   Min.   :0   Min.   :  0.
0   Min.   : 0.000
 100%_Natural_Bran       : 1   1st Qu.:100   1st Qu.:2.000   1st Qu.:0   1st Qu.:135.
0   1st Qu.: 0.250
 All-Bran                : 1   Median :110   Median :2.500   Median :1   Median :180.
0   Median : 2.000
 All-Bran_with_Extra_Fiber: 1   Mean   :107   Mean   :2.514   Mean   :1   Mean   :162.
4   Mean   : 2.176
 Apple_Cinnamon_Cheerios  : 1   3rd Qu.:110   3rd Qu.:3.000   3rd Qu.:1   3rd Qu.:217.
5   3rd Qu.: 3.000
 Apple_Jacks             : 1   Max.   :160   Max.   :6.000   Max.   :5   Max.   :320.
0   Max.   :14.000
 (Other)                 :68
     carbo           sugars           potass          vitamins          shelf
weight
 Min.   : 5.00   Min.   : 0.000   Min.   : 15.00   Min.   :  0.00   Min.   :1.000   Mi
n.   :0.500
 1st Qu.:12.00   1st Qu.: 3.000   1st Qu.: 41.25   1st Qu.: 25.00   1st Qu.:1.250   1s
t Qu.:1.000
 Median :14.50   Median : 7.000   Median : 90.00   Median : 25.00   Median :2.000   Me
dian :1.000
 Mean   :14.73   Mean   : 7.108   Mean   : 98.51   Mean   : 29.05   Mean   :2.216   Me
an   :1.031
 3rd Qu.:17.00   3rd Qu.:11.000   3rd Qu.:120.00   3rd Qu.: 25.00   3rd Qu.:3.000   3r
d Qu.:1.000
 Max.   :23.00   Max.   :15.000   Max.   :330.00   Max.   :100.00   Max.   :3.000   Ma
x.   :1.500

      cups            rating
 Min.   :0.2500   Min.   :18.04
 1st Qu.:0.6700   1st Qu.:32.45
 Median :0.7500   Median :40.25
 Mean   :0.8216   Mean   :42.37
 3rd Qu.:1.0000   3rd Qu.:50.52
 Max.   :1.5000   Max.   :93.70
```

Hide

```
rownames(Csdata2) <- Csdata2$name ##Convert the names of the breakfast cereals to the
row names, as this will later help us in visualising the clusters
```

Hide

```
Csdata2 <- Csdata2[,c(-1)] ##Drop the name column as it is now just redundant informat
ion
```

Hide

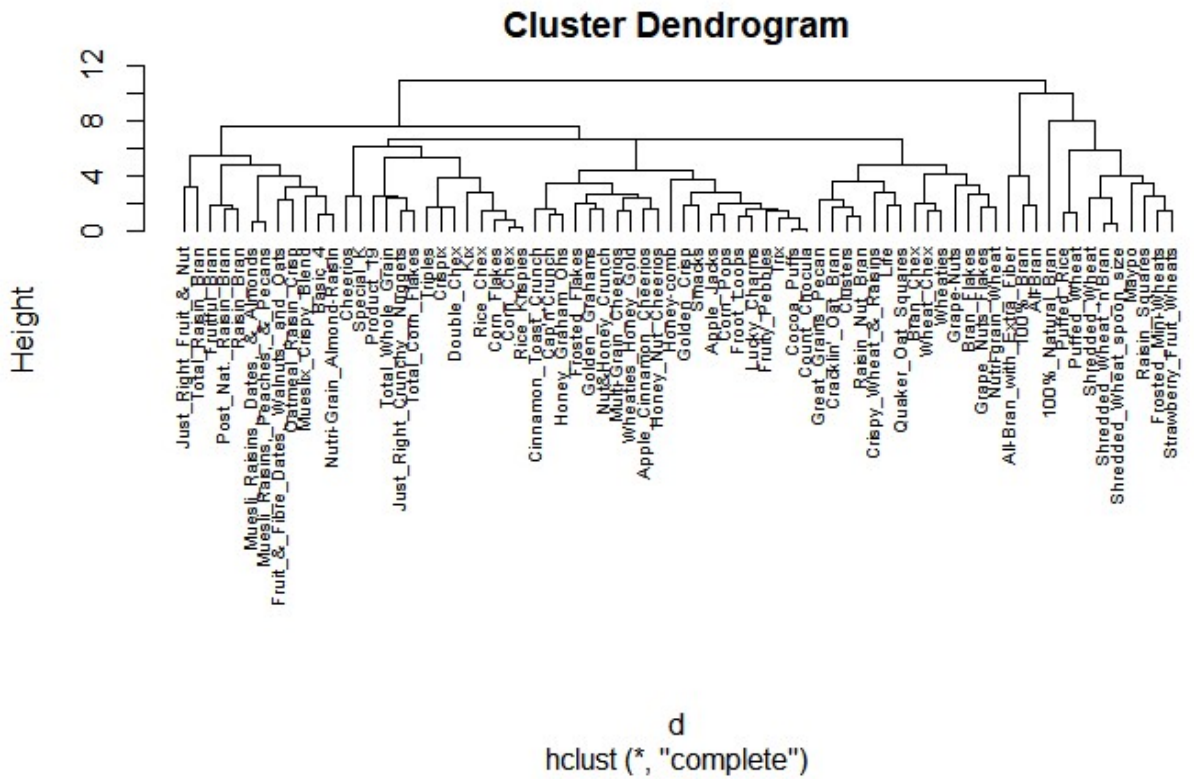
```
##The data must be scaled, before measuring any type of distance metric as the variabl
es with higher ranges will significantly influence the distance
Csdata2 <- scale(Csdata2)
head(Csdata2)
```

```
                           calories    protein        fat     sodium       fiber
carbo     sugars
100%_Bran                 -1.8659155  1.3817478  0.0000000 -0.3910227  3.22866747 -2.5
001396 -0.2542051
100%_Natural_Bran          0.6537514  0.4522084  3.9728810 -1.7804186 -0.07249167 -1.7
292632  0.2046041
All-Bran                  -1.8659155  1.3817478  0.0000000  1.1795987  2.81602258 -1.9
862220 -0.4836096
All-Bran_with_Extra_Fiber -2.8737823  1.3817478 -0.9932203 -0.2702057  4.87924705 -1.7
292632 -1.6306324
Apple_Cinnamon_Cheerios    0.1498180 -0.4773310  0.9932203  0.2130625 -0.27881412 -1.0
868662  0.6634132
Apple_Jacks                0.1498180 -0.4773310 -0.9932203 -0.4514312 -0.48513656 -0.9
583868  1.5810314
                              potass   vitamins      shelf     weight       cups     r
ating
100%_Bran                  2.5605229 -0.1818422  0.9419715 -0.2008324 -2.0856582  1.85
49038
100%_Natural_Bran          0.5147738 -1.3032024  0.9419715 -0.2008324  0.7567534 -0.59
77113
All-Bran                   3.1248675 -0.1818422  0.9419715 -0.2008324 -2.0856582  1.21
51965
All-Bran_with_Extra_Fiber  3.2659536 -0.1818422  0.9419715 -0.2008324 -1.3644493  3.65
78436
Apple_Cinnamon_Cheerios   -0.4022862 -0.1818422 -1.4616799 -0.2008324 -0.3038480 -0.91
65248
Apple_Jacks               -0.9666308 -0.1818422 -0.2598542 -0.2008324  0.7567534 -0.65
53998
```

Qs1 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.

Hide

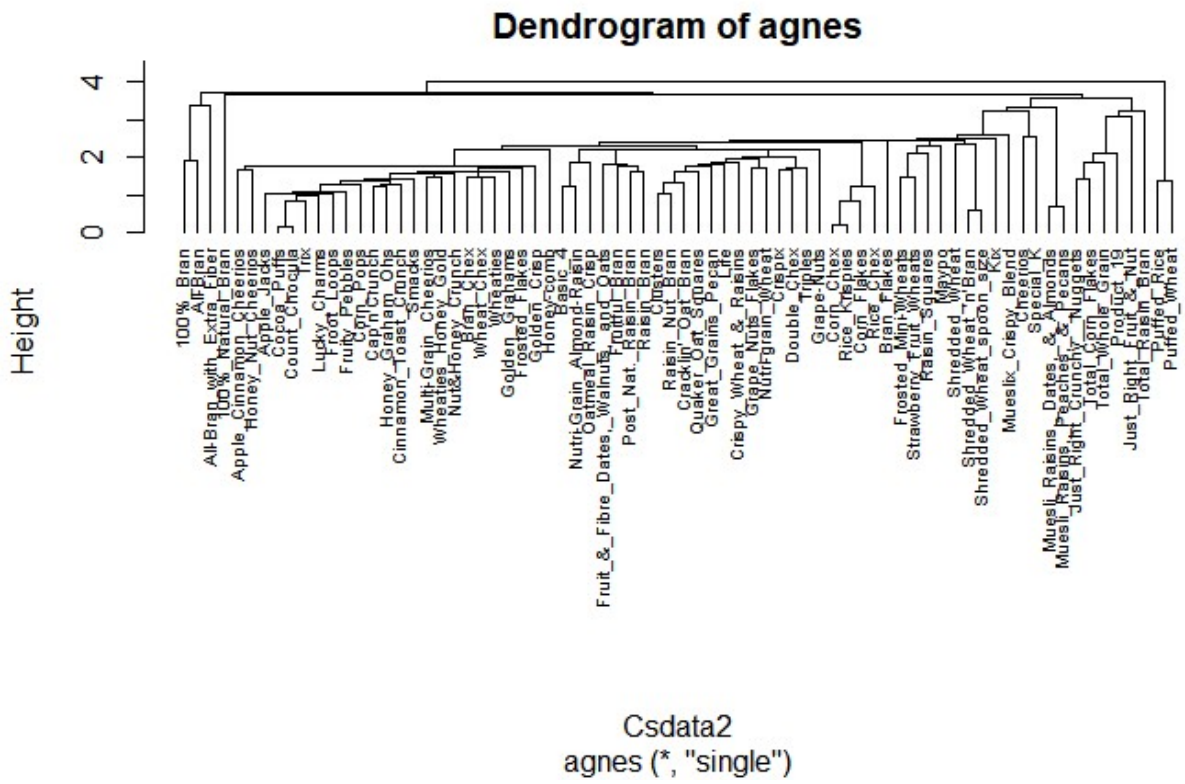
```
# Dissimilarity matrix
d <- dist(Csdata2, method = "euclidean")

# Hierarchical clustering using Complete Linkage
hc_complete <- hclust(d, method = "complete" )

# Plot the obtained dendrogram
plot(hc_complete, cex = 0.6, hang = -1)
```



Using Agnes to compare the clustering from single linkage, complete linkage, average linkage, and Ward and comparing agglomerative coefficients of each method.

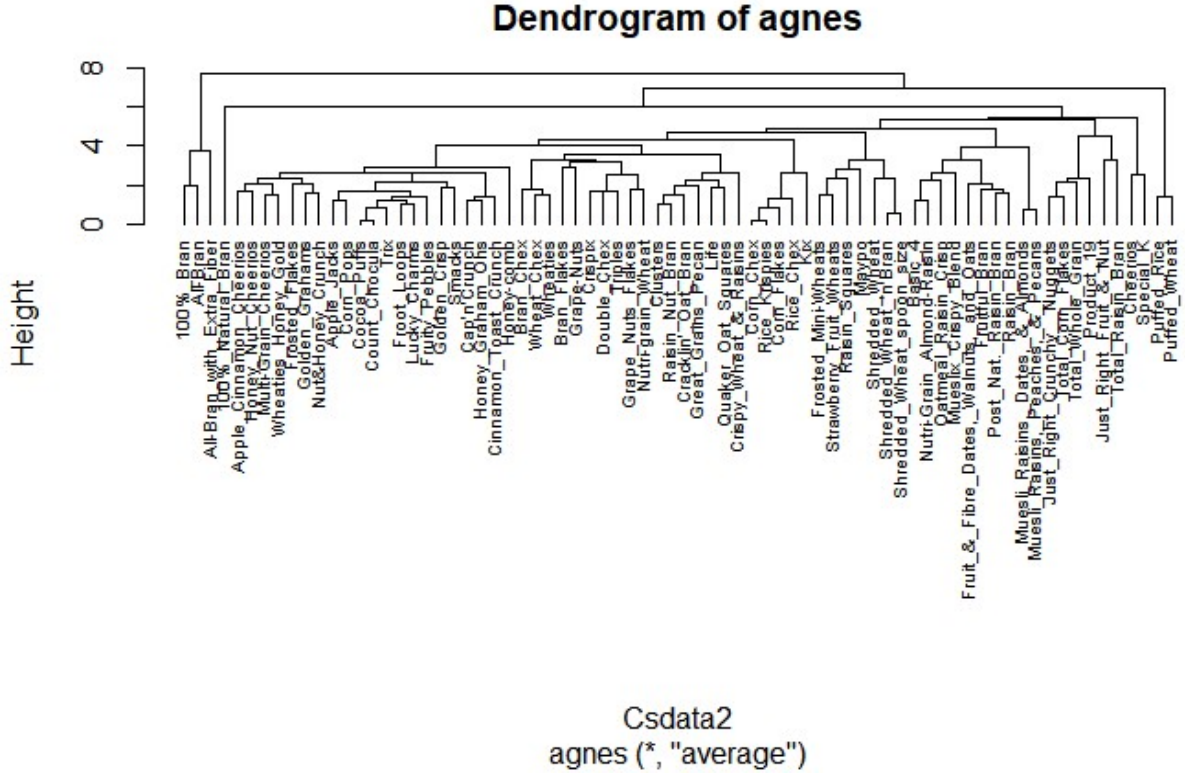
Hide

```
hc_single <- agnes(Csdata2, method = "single")
pltree(hc_single, cex = 0.6, hang = -1, main = "Dendrogram of agnes")
```



Hide

```
hc_average <- agnes(Csdata2, method = "average")
pltree(hc_average, cex = 0.6, hang = -1, main = "Dendrogram of agnes")
```



Hide

```
# Compute with agnes
hc_complete <- agnes(Csdata2, method = "complete")

# Agglomerative coefficient
hc_complete$ac
```

```
[1] 0.8353712
```

Hide

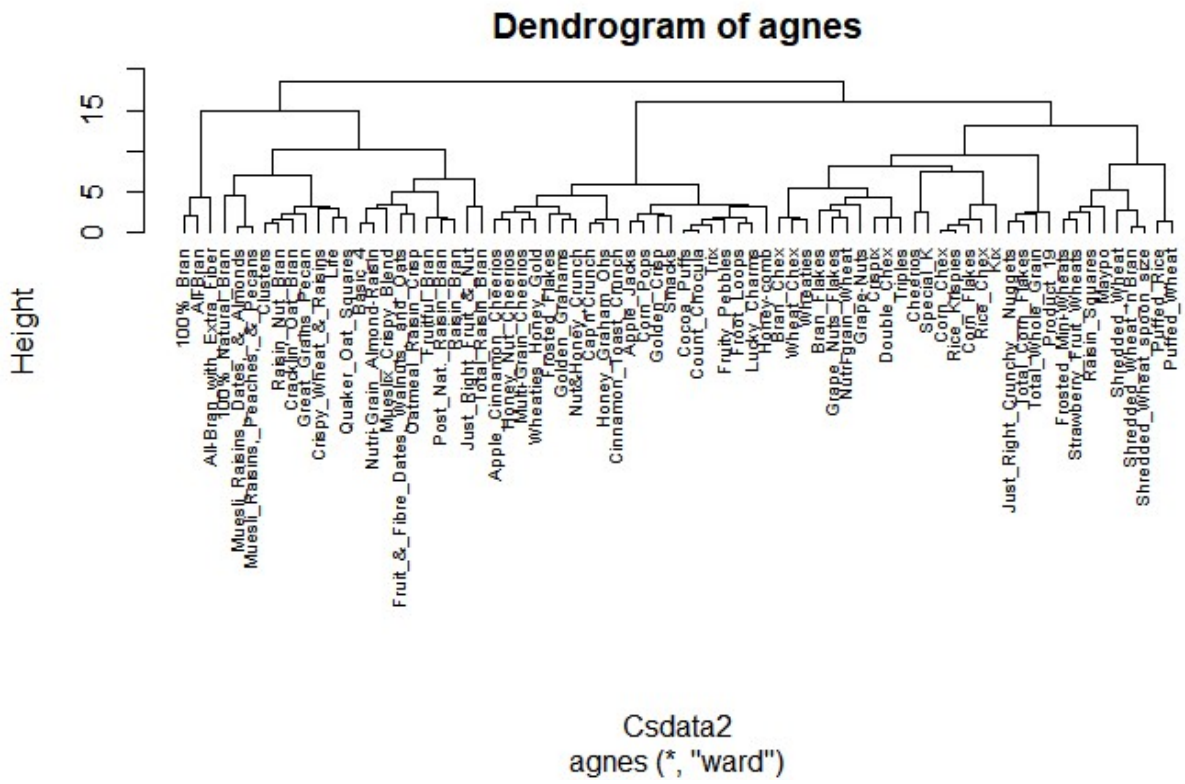
```
##We will find the agnes coefficient of all the methods.

library(tidyverse)
# methods to assess

m <- c( "average", "single", "complete", "ward")
names(m) <- c( "average", "single", "complete", "ward")

# function to compute coefficient
ac <- function(x) {
  agnes(Csdata2, method = x)$ac
}

map_dbl(m, ac)
```

```
  average    single  complete      ward
0.7766075 0.6067859 0.8353712 0.9046042
```

The best linkage method is ward with agglomerative coefficient of 0.9046042.

Hide

```
hc_ward <- agnes(Csdata2, method = "ward")
pltree(hc_ward, cex = 0.6, hang = -1, main = "Dendrogram of agnes") ##visualizing the
dendrogram using wards method:
```

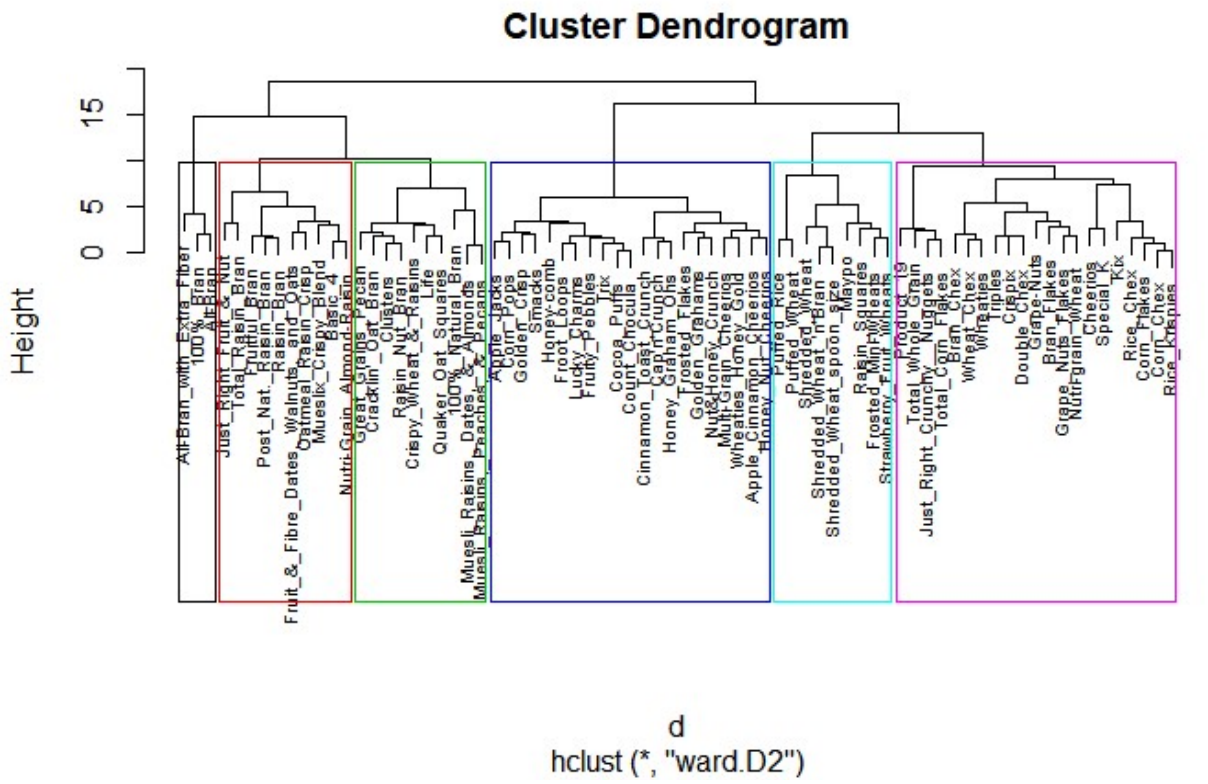


Qs 2) How many clusters would you choose?

Referring the dendrogram, a suitable height shall be chosen where a horizontal line shall be plotted and the number of instances where the horizontal line cut the branches of dendrogram are clusters of our data. So, our hierarchical clustering analysis is producing 6 clusters.

Hide

```
#Create the distance matrix
d <- dist(Csdata2, method = "euclidean")

# Ward's method for Hierarchical clustering
hc_ward_cut <- hclust(d, method = "ward.D2" )

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



Hide

```
##Lets see how many number of records of the data grouped and assigned  to clusters
# Cut tree into 6 groups
sub_grp <- cutree(hc_ward_cut, k = 6)

# Number of members in each cluster
table(sub_grp)
```

```
sub_grp
 1  2  3  4  5  6
 3 10 21 10 21  9
```

Hide

```
cluster <- cutree(hc_ward,k=6)
cluster
```

100%_Bran	100%_Natural_Bran
1	2
All-Bran	All-Bran_with_Extra_Fiber
1	1
Apple_Cinnamon_Cheerios	Apple_Jacks
3	3
Basic_4	Bran_Chex
4	5
Bran_Flakes	Cap'n'Crunch
5	3
Cheerios	Cinnamon_Toast_Crunch
5	3
Clusters	Cocoa_Puffs
2	3
Corn_Chex	Corn_Flakes
5	5
Corn_Pops	Count_Chocula
3	3
Cracklin'_Oat_Bran	Crispix
2	5
Crispy_Wheat_&_Raisins	Double_Chex
2	5
Froot_Loops	Frosted_Flakes
3	3
Frosted_Mini-Wheats	Fruit_&_Fibre_Dates,_Walnuts,_and_Oats
6	4
Fruitful_Bran	Fruity_Pebbles
4	3
Golden_Crisp	Golden_Grahams
3	3
Grape_Nuts_Flakes	Grape-Nuts
5	5
Great_Grains_Pecan	Honey_Graham_Ohs
2	3
Honey_Nut_Cheerios	Honey-comb
3	3
Just_Right_Crunchy__Nuggets	Just_Right_Fruit_&_Nut
5	4
Kix	Life
5	2
Lucky_Charms	Maypo
3	6
Muesli_Raisins,_Dates,_&_Almonds	Muesli_Raisins,_Peaches,_&_Pecans
2	2
Mueslix_Crispy_Blend	Multi-Grain_Cheerios
4	3
Nut&Honey_Crunch	Nutri-Grain_Almond-Raisin
3	4

Nutri-grain_Wheat	Oatmeal_Raisin_Crisp
5	4
Post_Nat._Raisin_Bran	Product_19
4	5
Puffed_Rice	Puffed_Wheat
6	6
Quaker_Oat_Squares	Raisin_Bran
2	4
Raisin_Nut_Bran	Raisin_Squares
2	6
Rice_Chex	Rice_Krispies
5	5
Shredded_Wheat	Shredded_Wheat_'n'Bran
6	6
Shredded_Wheat_spoon_size	Smacks
6	3
Special_K	Strawberry_Fruit_Wheats
5	6
Total_Corn_Flakes	Total_Raisin_Bran
5	4
Total_Whole_Grain	Triples
5	5
Trix	Wheat_Chex
3	5
Wheaties	Wheaties_Honey_Gold
5	3

QS3) 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.

Hide

```
library(knitr)
library(dendextend)
library(ggplot2)
##install.packages("hrbrthemes")
library(hrbrthemes)
##install.packages("viridisLite")
library(viridis)
ce <-data.frame(cbind(Csdata2,cluster))  ## combining cluster lable with subset of the
data to view all information in one table




csdata3 <- na.omit(Csdata) ##creating new subset of data and deleting rows with "NA" v
alue

train.data <- csdata3[1:60,]  ## creating partition of subset  by taking first 60 row
test.data <- csdata3[61:74,]  ## creating partition of subset  by taking next 14 row o
f"newdata"

train <- scale(train.data[,-c(1:3)]) ## Standardizing data for all numeric data for su
bset
test <- scale(test.data[,-c(1:3)]) ## Standardizing data for all numeric data for subs
et

# Computing  agnes with different linkage method to identify highest coefficients



train_ward<- agnes(train,method = "ward")
train_avg<-agnes(train,method="average")
train_com<-agnes(train,method="complete")
train_sin<-agnes(train,method="single")
kable(cbind(ward=train_ward$ac,average=train_avg$ac,complete=train_com$ac,single=train
_sin$ac)) ## this function provide information for Agglomerative coefficients for all
linkage in a tabular format and easily view
```

ward	average	complete	single
0.8898441	0.7567786	0.8207517	0.6672134

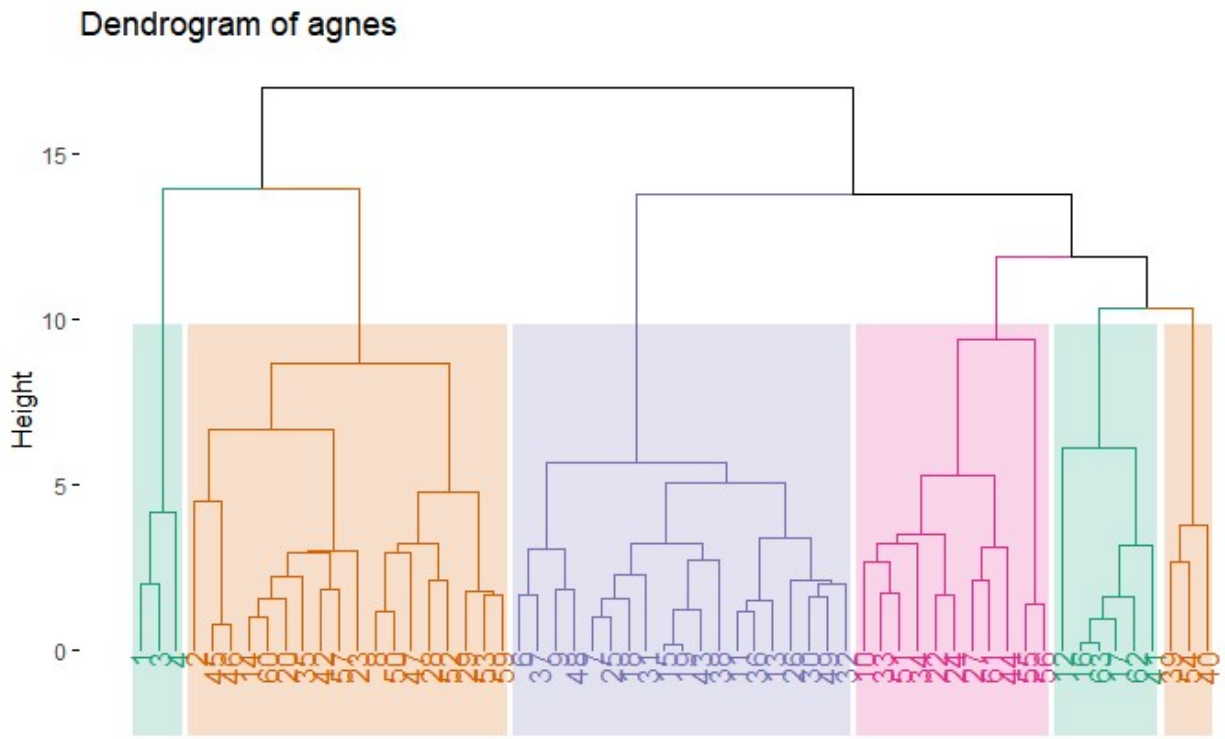
Hide

```
NA
```

Hide

```
fviz_dend(train_ward,k=6,cex=0.9,k_colors = c("#1B9E77", "#D95F02", "#7570B3", "#E7298
A"),rect_border = c("#1B9E77", "#D95F02", "#7570B3", "#E7298A"), rect_fill = TRUE  ,co
lor_labels_by_k = TRUE,rect= TRUE,main="Dendrogram of agnes")
```

```
Length of color vector was shorter than the number of clusters - color vector was recy
cledthe condition has length > 1 and only the first element will be used
```



##dendrogram shows 6 distinct cluster

Hide

```
clust.train <- cutree(hc_ward_cut,k=6) # getting lable for data points

b1 <-data.frame(cbind(train,clust.train))  ## combining cluster lable with subset of t
he data to view all information in one table
```

```
number of rows of result is not a multiple of vector length (arg 2)
```

Hide

```
n1<-data.frame(column=seq(1,13,1),mean=rep(0,13))## to find centroid of variable for c
luster lable with k=1
n2<-data.frame(column=seq(1,13,1),mean=rep(0,13))## to find centroid of variable for c
luster lable with k=2
n3<-data.frame(column=seq(1,13,1),mean=rep(0,13))## to find centroid of variable for c
luster lable with k=3
n4<-data.frame(column=seq(1,13,1),mean=rep(0,13))## to find centroid of variable for c
luster lable with k=4
n5<-data.frame(column=seq(1,13,1),mean=rep(0,13))## to find centroid of variable for c
luster lable with k=5
n6<-data.frame(column=seq(1,13,1),mean=rep(0,13))## to find centroid of variable for c
luster lable with k=6
for(i in 1:13) ## formation of " for loop" to find centroid for all column in subsent
of the data
{
  n1[i,2]<-mean(b1[b1$clust==1,i])
  n2[i,2]<-mean(b1[b1$clust==2,i])
  n3[i,2]<-mean(b1[b1$clust==3,i])
  n4[i,2]<-mean(b1[b1$clust==4,i])
  n5[i,2]<-mean(b1[b1$clust==5,i])
  n6[i,2]<-mean(b1[b1$clust==6,i])
}
centroid.train<-t(cbind(n1$mean,n2$mean,n3$mean,n4$mean,n5$mean,n6$mean)) ## combining
mean of each column and then transpose data from column to row for better view
colnames(centroid.train)<-colnames(Csdata1[,-c(1:3)])  ## assign column name to matrix
from orignal data

centroid.train
```

```
       calories     protein         fat     sodium      fiber      carbo      sugars
potass   vitamins
[1,] -2.1272666  1.42341904 -0.38646082  0.1080686  3.3813952 -1.9394118 -0.95559258
2.8120550 -0.1416617
[2,]  0.3903972  0.76645640  1.31396678 -0.6378166  0.1344873 -0.4985852  0.05717221
0.4074633 -0.2833233
[3,]  0.1566900 -0.92287608 -0.07729216  0.1292585 -0.6318769 -0.4323888  0.89569789 -
0.7155725 -0.1416617
[4,]  1.1127649  0.48490099  0.33493271  0.3058412  0.5422187  0.3281648  0.73234873
0.8845117  0.3305439
[5,] -0.2151168  0.04692590 -0.63379574  0.7776701 -0.2049330  1.1028028 -1.00459733 -
0.3896019  0.4249850
[6,] -1.4261450 -0.07820984 -0.81929694 -2.1380629 -0.3842495 -0.1816878 -1.06993699 -
0.3494137 -0.7083083
          shelf    weight       cups      rating
[1,]  0.8630890 -0.207590 -1.6847616  2.46712909
[2,]  0.7367833 -0.207590 -0.5345494 -0.05241472
[3,] -0.8209871 -0.207590  0.3475015 -0.94490144
[4,]  0.7227493  1.886053 -0.5234684 -0.28898963
[5,] -0.2315605 -0.207590  0.5225819  0.41122801
[6,]  0.3578662 -1.485067  0.2034479  1.31269447
```

Hide

```
c1<-data.frame(data=seq(1,14,1),cluster=rep(0,14))  ##creating a function for a "for l
oop" to find minimum distance form cluster centroid to each observation in test data i
n order to see which cluster will these observation fall in

for(i in 1:14)   ## " for loop " for 14 observation in test data
{
  x1<-as.data.frame(rbind(centroid.train,test[i,]))
  z1<-as.matrix(get_dist(x1))
  c1[i,2]<-which.min(z1[5,-5])
}

c1 ## tabluar form with observation no. and cluster no.
```

data <dbl>	cluster <dbl>
1	3
2	3
3	3
4	3
5	3

data <dbl>	cluster <dbl>
6	6
7	6
8	3
9	6
10	6

1-10 of 14 rows Previous **1** 2 Next

Hide

```
kable(cbind(data_labels=ce[61:74,14],partition_labels=c1$cluster))  ## checking cluste
r label for 14 observation form test.data against orignal cluster lables formed in " S
TEP 1"
```

data_labels	partition_labels
6	3
6	3
6	3
3	3
5	3
6	6
5	6
4	3
5	6
5	6
3	3
5	6
5	6
3	3

Hide

```
### on further investigation we see the stability of cluster is 92% ie. out of 14 obse
rvation it has predicted 13 observation as correct
```

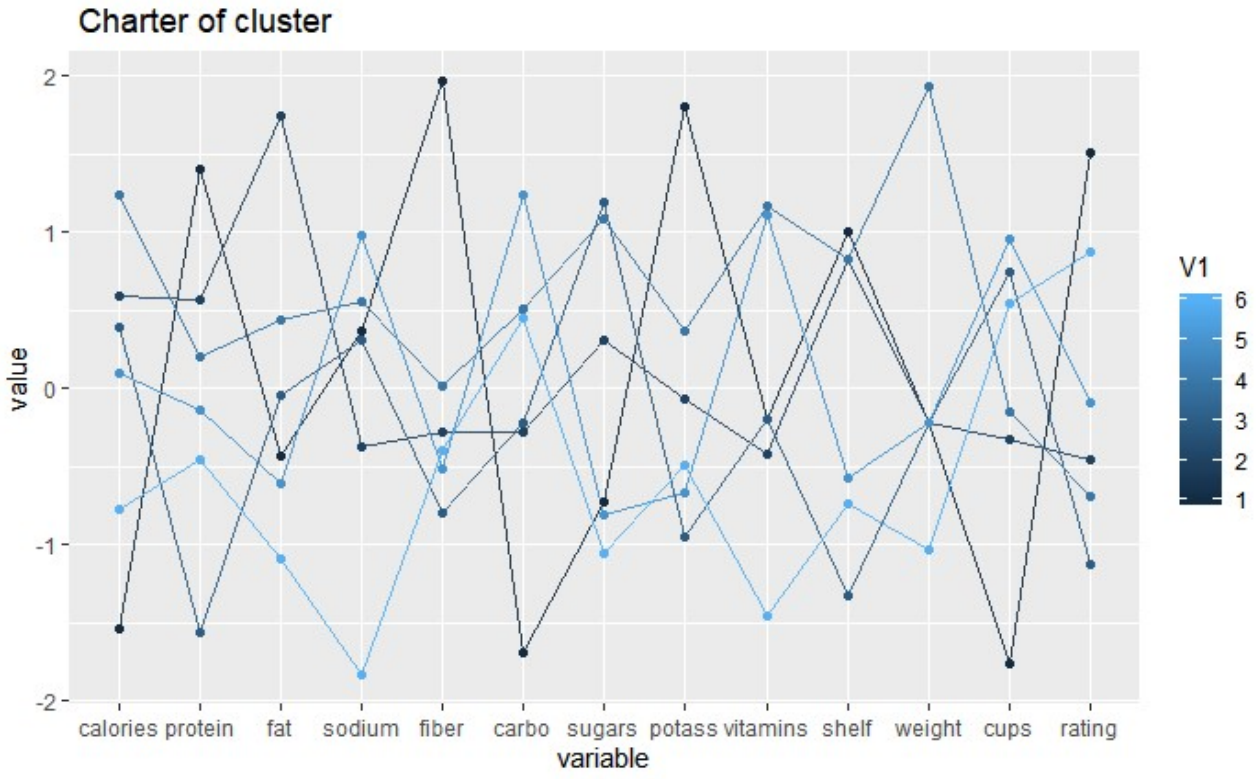
QS 4) The elementary public schools would like to choose a set of cereals to include in their daily cafeterias. Every day a different cereal is offered, but all cereals should support a healthy diet. For this goal, you are requested to find a cluster of “healthy cereals.” Should the data be normalized?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

Hide

```
ce <-data.frame(cbind(Csdata2,cluster))  ## combining cluster lable with subset of the
data to view all information in one table

m1<-data.frame(column=seq(1,13,1),mean=rep(0,13)) ## to find centroid of variable for
cluster lable with k=1
m2<-data.frame(column=seq(1,13,1),mean=rep(0,13)) ## to find centroid of variable for
cluster lable with k=2
m3<-data.frame(column=seq(1,13,1),mean=rep(0,13)) ## to find centroid of variable for
cluster lable with k=3
m4<-data.frame(column=seq(1,13,1),mean=rep(0,13)) ## to find centroid of variable for
cluster lable with k=4
m5<-data.frame(column=seq(1,13,1),mean=rep(0,13)) ## to find centroid of variable for
cluster lable with k=5
m6<-data.frame(column=seq(1,13,1),mean=rep(0,13)) ## to find centroid of variable for
cluster lable with k=6
for(i in 1:13)  ## formation of for loop" to find centroid for all column in subsent o
f the data
{
  m1[i,2]<-mean(ce[ce$cluster==1,i])
  m2[i,2]<-mean(ce[ce$cluster==2,i])
  m3[i,2]<-mean(ce[ce$cluster==3,i])
  m4[i,2]<-mean(ce[ce$cluster==4,i])
  m5[i,2]<-mean(ce[ce$cluster==5,i])
  m6[i,2]<-mean(ce[ce$cluster==6,i])
}
centroid<-t(cbind(m1$mean,m2$mean,m3$mean,m4$mean,m5$mean,m6$mean)) ## combining mean
of each column and then transpose data from column to row for better view
colnames(centroid)<-colnames(Csdata1[,-c(1:3)]) ## assign column name to matrix from o
rignal data

centroid ## view information
```

Hide

```
library(ggplot2)
library(GGally)

#  ploting mean of each variable in cluster on chart to identify  bestpossible combina
tion
ggparcoord(cbind(c(1:6),centroid),columns = 2:14,groupColumn = 1,showPoints = TRUE,tit
le = " Charter of cluster",alphaLines = 0.9)
```



Hide

```
table(cluster)
```

```
cluster
 1  2  3  4  5  6
 3 10 21 10 21  9
```

Hide

```
res1<-cbind(csdata3,cluster)
res1[res1$cluster==1,]
```

name	mfr	t…	calories	protein	f…	sodi…	fiber	ca…
<fctr>	<fctr>	<fctr>	<int>	<int>	<int>	<int>	<dbl>	<dbl>
1 100%_Bran	N	C	70	4	1	130	10	5

	name	mfr	t…	calories	protein	f…	sodi…	fiber	ca…
	<fctr>	<fctr>	<fctr>	<int>	<int>	<int>	<int>	<dbl>	<dbl>
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

3 rows | 1-10 of 17 columns

Hide

```
res1[res1$cluster==2,]
```

	name	mfr	t…	calories	protein	f…	sodi…	fib
	<fctr>	<fctr>	<fctr>	<int>	<int>	<int>	<int>	<db
2	100%_Natural_Bran	Q	C	120	3	5	15	2
14	Clusters	G	C	110	3	2	140	2
20	Cracklin'_Oat_Bran	K	C	110	3	3	140	4
23	Crispy_Wheat_&_Raisins	G	C	100	2	1	140	2
35	Great_Grains_Pecan	P	C	120	3	3	75	3
42	Life	Q	C	100	4	2	150	2
45	Muesli_Raisins,_Dates,_&_Almonds	R	C	150	4	3	95	3
46	Muesli_Raisins,_Peaches,_&_Pecans	R	C	150	4	3	150	3
57	Quaker_Oat_Squares	Q	C	100	4	1	135	2
60	Raisin_Nut_Bran	G	C	100	3	2	140	2

1-10 of 10 rows | 1-10 of 17 columns

Hide

```
res1[res1$cluster==3,]
```

	name	mfr	t…	calories	protein	fat	sodi…	fiber	car…
	<fctr>	<fctr>	<fctr>	<int>	<int>	<int>	<int>	<dbl>	<dbl>
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

	name	mfr	t…	calories	protein	fat	sodi…	fiber	car…
	<fctr>	<fctr>	<fctr>	<int>	<int>	<int>	<int>	<dbl>	<dbl>
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

1-10 of 21 rows | 1-10 of 17 columns Previous **1** 2 3 Next

Hide

```
NA
```

Hide

```
res1[res1$cluster==4,]
```

	name	mfr	t…	calories	protein	f…	sodi.
	<fctr>	<fctr>	<fctr>	<int>	<int>	<int>	<in
8	Basic_4	G	C	130	3	2	21
28	Fruit_&_Fibre_Dates,_Walnuts,_and_Oats	P	C	120	3	2	16
29	Fruitful_Bran	K	C	120	3	0	24
40	Just_Right_Fruit_&_Nut	K	C	140	3	1	17
47	Mueslix_Crispy_Blend	K	C	160	3	2	15
50	Nutri-Grain_Almond-Raisin	K	C	140	3	2	22
52	Oatmeal_Raisin_Crisp	G	C	130	3	2	17
53	Post_Nat._Raisin_Bran	P	C	120	3	1	20
59	Raisin_Bran	K	C	120	3	1	21
71	Total_Raisin_Bran	G	C	140	3	1	19

1-10 of 10 rows | 1-10 of 17 columns

Hide

```
res1[res1$cluster==5,]
```

	name <fctr>	mfr <fctr>	t… <fctr>	calories <int>	protein <int>	f… <int>	sodi… <int>	fiber <dbl>	ca… <dbl>
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

1-10 of 21 rows | 1-10 of 17 columns Previous **1** 2 3 Next

Hide

```
res1[res1$cluster==6,]
```

	name <fctr>	mfr <fctr>	t… <fctr>	calories <int>	protein <int>	f… <int>	sodi… <int>	fiber <dbl>	ca… <dbl>
27	Frosted_Mini-Wheats	K	C	100	3	0	0	3	14
44	Maypo	A	H	100	4	1	0	0	16
55	Puffed_Rice	Q	C	50	1	0	0	0	13
56	Puffed_Wheat	Q	C	50	2	0	0	1	10
61	Raisin_Squares	K	C	90	2	0	0	2	15
64	Shredded_Wheat	N	C	80	2	0	0	3	16
65	Shredded_Wheat_'n'Bran	N	C	90	3	0	0	4	19
66	Shredded_Wheat_spoon_size	N	C	90	3	0	0	3	20
69	Strawberry_Fruit_Wheats	N	C	90	2	0	15	3	15

9 rows | 1-10 of 17 columns

Hide

1) Healthy cereals with much dietary fibers, less calories and less fats : 100% Bran, All-Bran with extra fibers, and All-Bran.
2) Cereals for hungry people in need of energy are Muesli Cereals

The clusters contain nutritionally rich, adequate and poor levels. We grouped all the records in the 6 clusters