

Assignment-5 Hierarchical Clustering

Tanmoy

12/5/2020

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

Hierarchical clustering

```
rm(list = ls())
library(caret)
```

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

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

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

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

```
library(cluster)
library(NbClust)

#Loadinng the data
DFCereals<-read.csv("Cereals.csv")

#EDA of the data set
summary(DFCereals)
```

```
##      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
##
```

```
colMeans(is.na(DFCereals))
```

```
## 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
```

```
#Median imputation of missing data
preProcess_1<-preProcess(DFCereals, method = c("medianImpute"))
ImputedDF<-predict(preProcess_1, DFCereals)

#No more NULL values presnt
colMeans(is.na(ImputedDF))
```

```
## 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
```

```
#Scaling the DF
ImputedDF<-subset(ImputedDF, select= -c(1,2,3))
ImputedDF<-scale(ImputedDF)
```

```
# Compute with agnes and with different linkage methods
hc_single <- agnes(ImputedDF, method = "single")
```

```
hc_complete<-agnes(ImputedDF, method="complete")
hc_average <- agnes(ImputedDF, method = "average")
hc_ward <- agnes(ImputedDF, method = "ward")
```

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

```
## [1] 0.6029274
```

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

```
## [1] 0.8353216
```

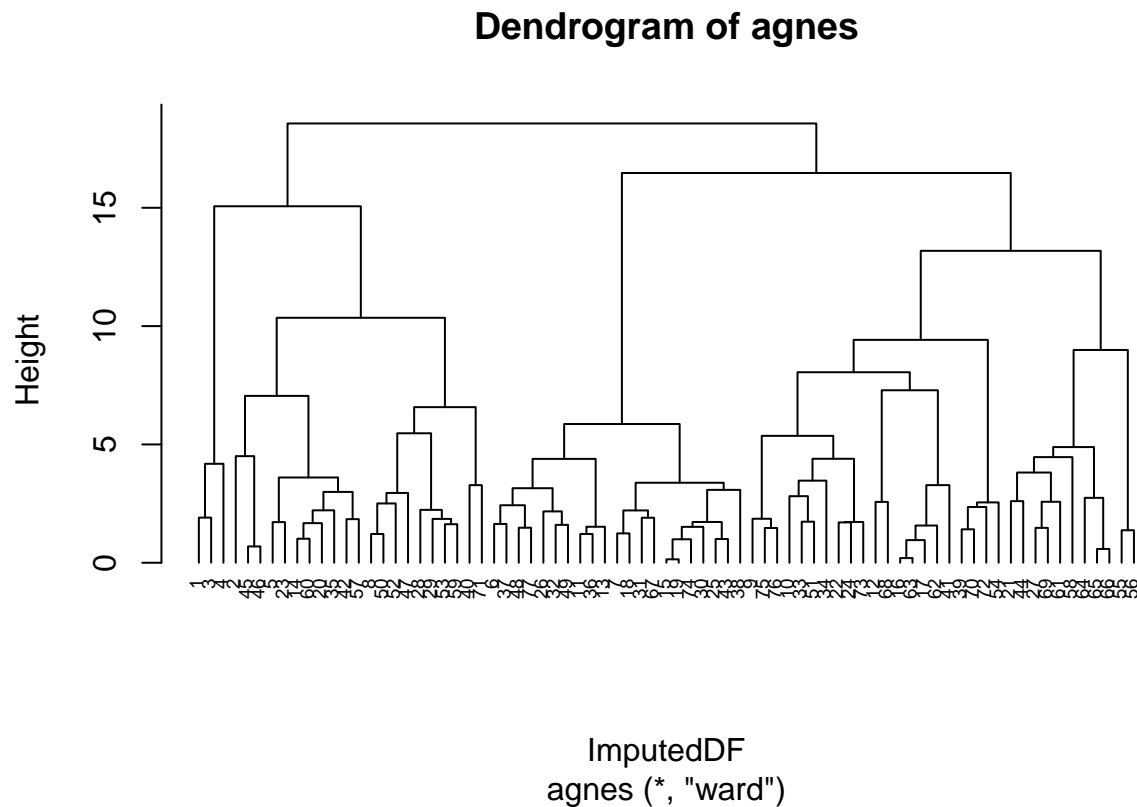
```
print(hc_average$ac)
```

```
## [1] 0.7777555
```

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

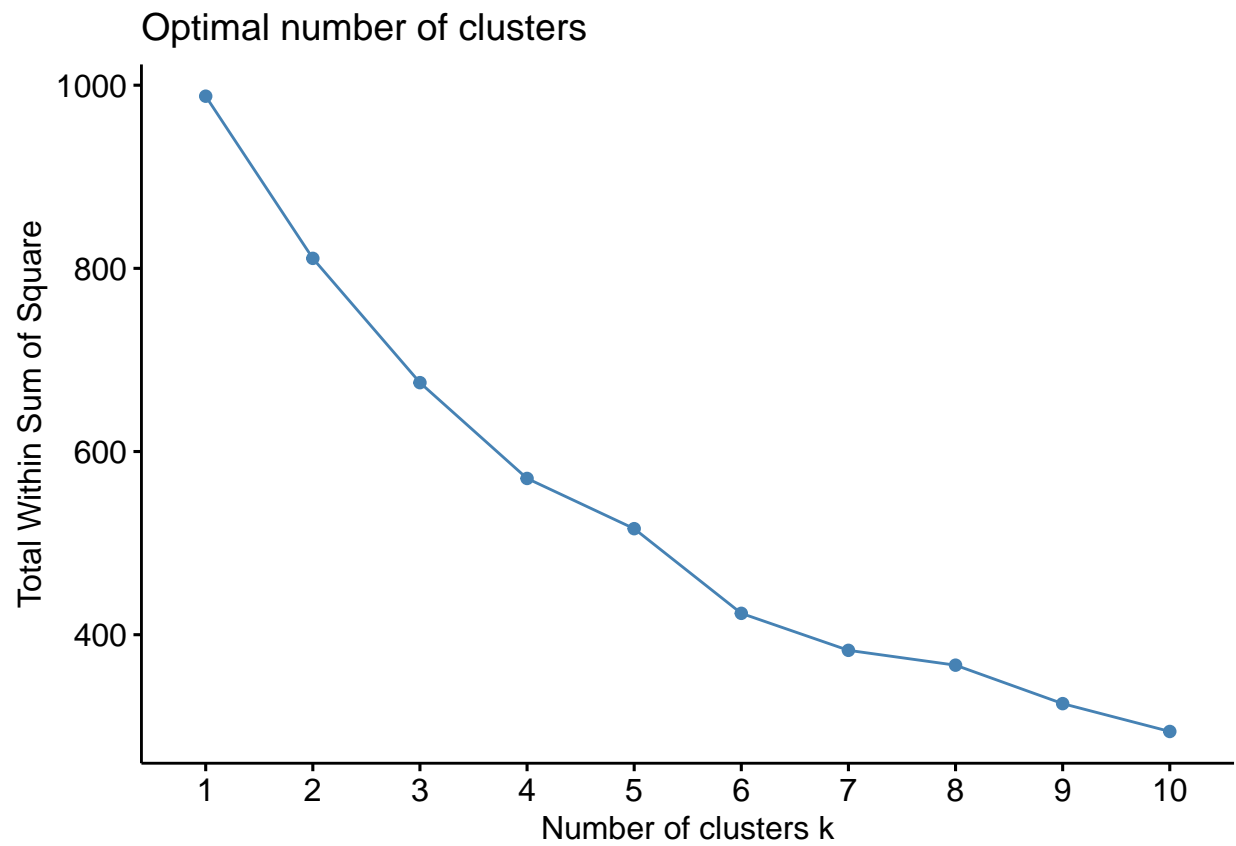
```
## [1] 0.9027089
```

```
# The approach used by Ward describes the best clustering mechanism of the four approaches tested
#visualize the dendrogram
pltree(hc_ward, cex = 0.6, hang = -1, main = "Dendrogram of agnes")
```

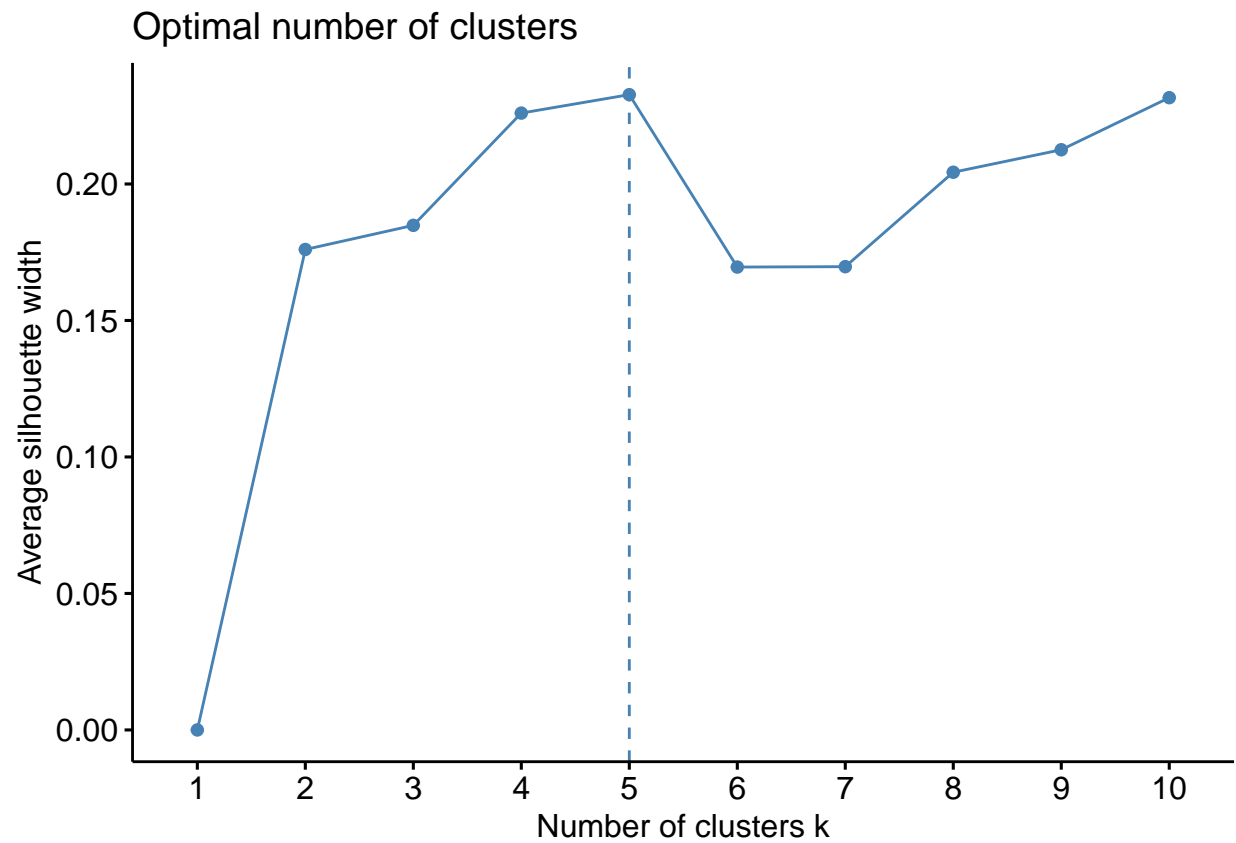


Q> Comment on differences between hierarchical Clustering and K-means

```
set.seed(123)
#Finding optimal number of clusters - Elbow Method
fviz_nbclust(ImputedDF, kmeans, method = "wss")
```

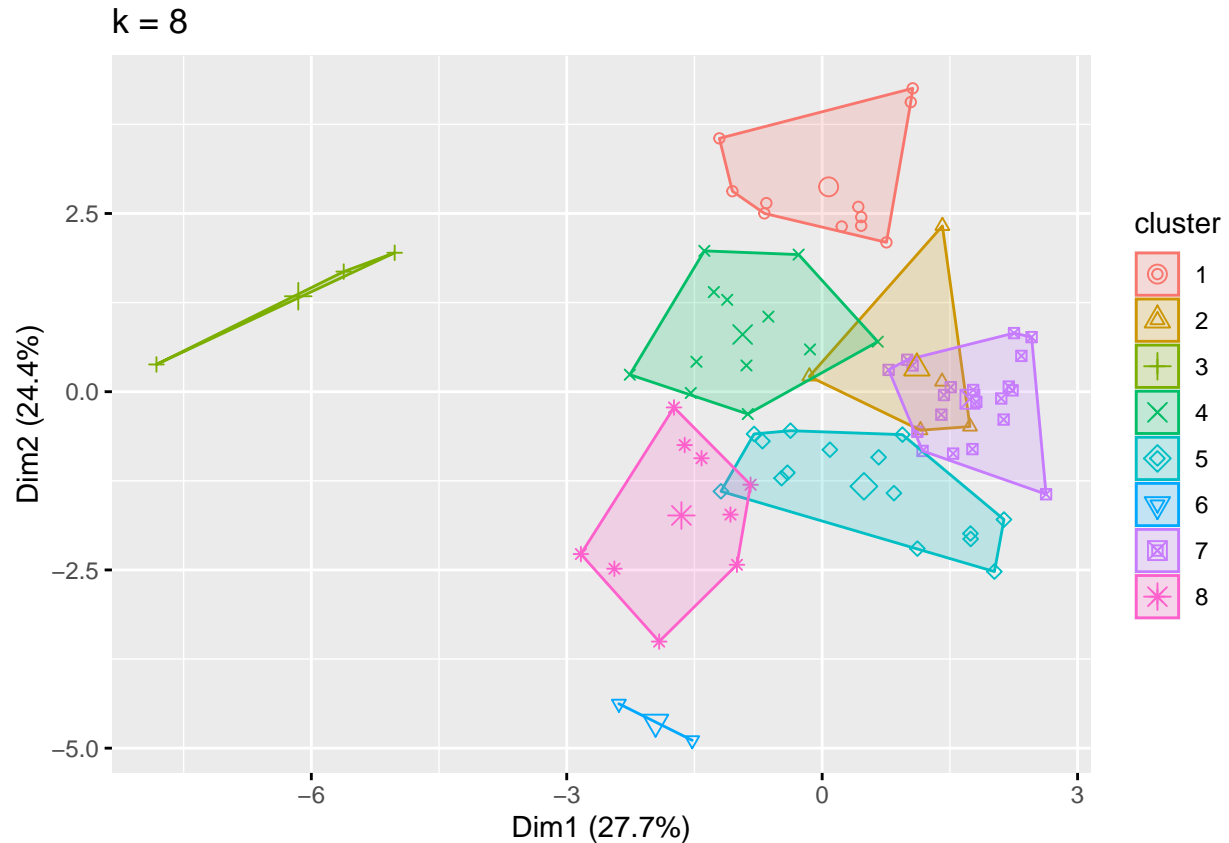


```
#Determining Optimal Cluster by Average Silhouette Method
fviz_nbclust(ImputedDF, kmeans, method = "silhouette")
```



#Silhouette method shows that 8 numbers of clusters would be optimum.

```
k8 <- kmeans(ImputedDF, centers = 8, nstart = 25)
fviz_cluster(k8, geom = "point", data = ImputedDF) + ggtitle("k = 8")
```



```
# slicing the dendrogram on the longest path, 8 is the optimal level of clusters.
# Cut tree into 8 groups
sub_grp <- cutree(hc_ward, k = 8)
C2 <- as.data.frame(cbind(ImputedDF, sub_grp))
head(C2)
```

	calories	protein	fat	sodium	fiber	carbo
## 1	-1.8929836	1.3286071	-0.01290349	-0.3539844	3.29284661	-2.5243405
## 2	0.6732089	0.4151897	3.96137277	-1.7257708	-0.06375361	-1.7514808
## 3	-1.8929836	1.3286071	-0.01290349	1.1967306	2.87327158	-2.0091007
## 4	-2.9194605	1.3286071	-1.00647256	-0.2346986	4.97114672	-1.7514808
## 5	0.1599704	-0.4982277	0.98066557	0.4810160	-0.48332864	-0.2057614
## 6	0.1599704	-0.4982277	0.98066557	0.2424445	-0.27354112	-1.1074310

	sugars	potass	vitamins	shelf	weight	cups	rating
## 1	-0.2358694	2.6126578	-0.1453172	0.9515734	-0.1967771	-2.1100340	1.8321876
## 2	0.2239266	0.5260824	-1.2642598	0.9515734	-0.1967771	0.7690100	-0.6180571
## 3	-0.4657674	3.1882648	-0.1453172	0.9515734	-0.1967771	-2.1100340	1.1930986
## 4	-1.6152574	3.3321665	-0.1453172	0.9515734	-0.1967771	-1.3795303	3.6333849
## 5	0.2239266	-0.1214755	-0.1453172	0.9515734	-0.1967771	-0.3052601	-0.5894990
## 6	0.6837226	-0.4092790	-0.1453172	-1.4507595	-0.1967771	-0.3052601	-0.9365625

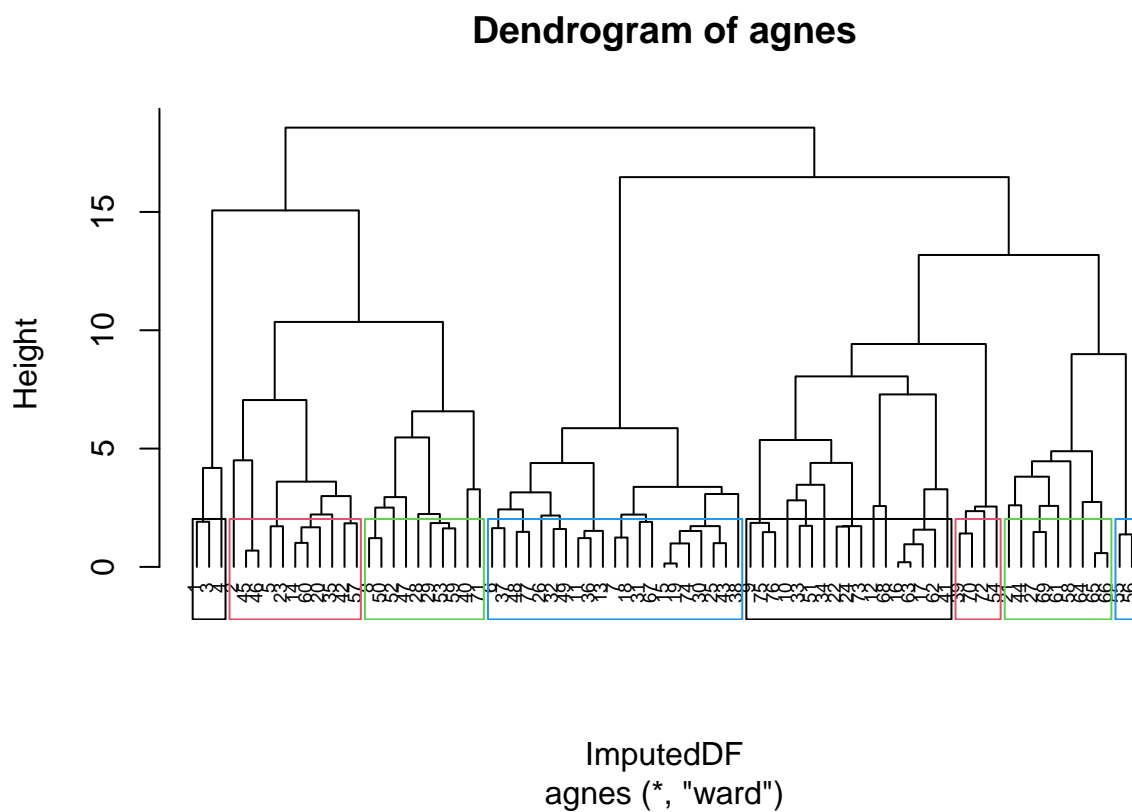
	sub_grp
## 1	1
## 2	2
## 3	1
## 4	1
## 5	2

```
## 6      3
```

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

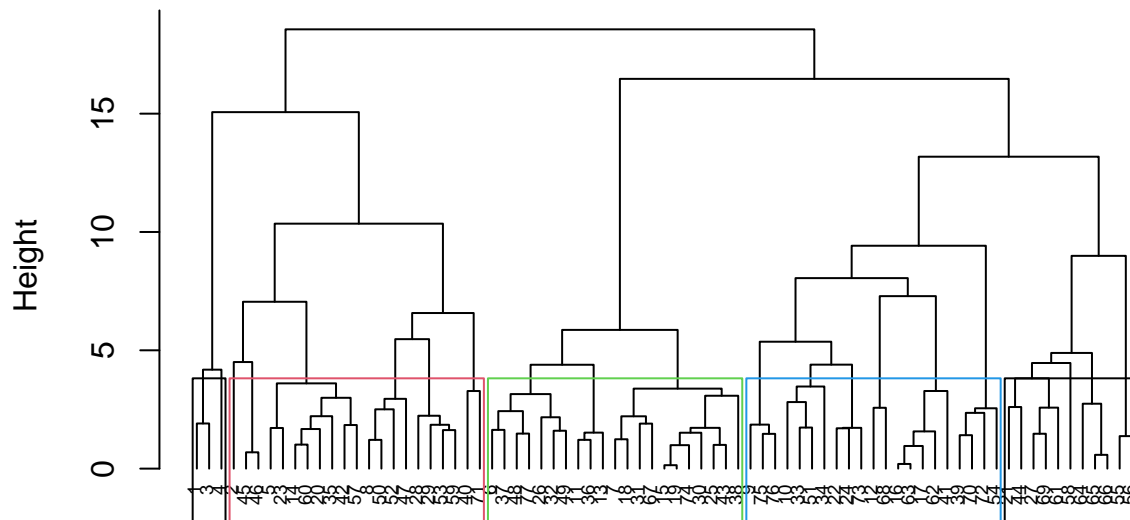
```
## sub_grp  
##  1  2  3  4  5  6  7  8  
##  3 11 21 10 17  9  4  2
```

```
#plot dendrogram  
pltree(hc_ward, cex = 0.6, hang = -1, main = "Dendrogram of agnes")  
rect.hclust(hc_ward, k = 8, border = 1:4)
```



```
#From the Dendrogram it seems that clustering at level of 5 would have been optimum.  
pltree(hc_ward, cex = 0.6, hang = -1, main = "Dendrogram of agnes")  
rect.hclust(hc_ward, k = 5, border = 1:4)
```

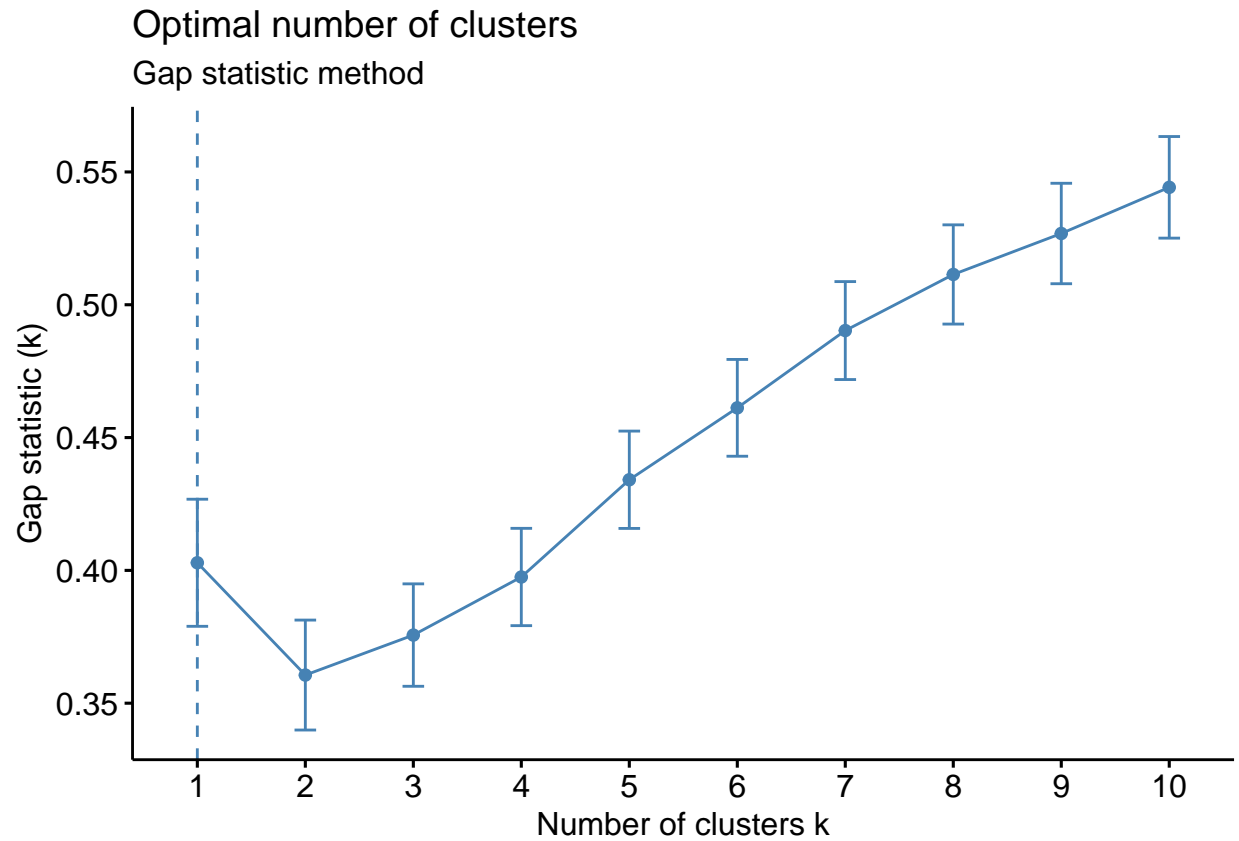
Dendrogram of agnes



ImputedDF
agnes (*, "ward")

#Thus, Hierarchical clustering would be better approach than kmeans to find the suitable no. of cluster.

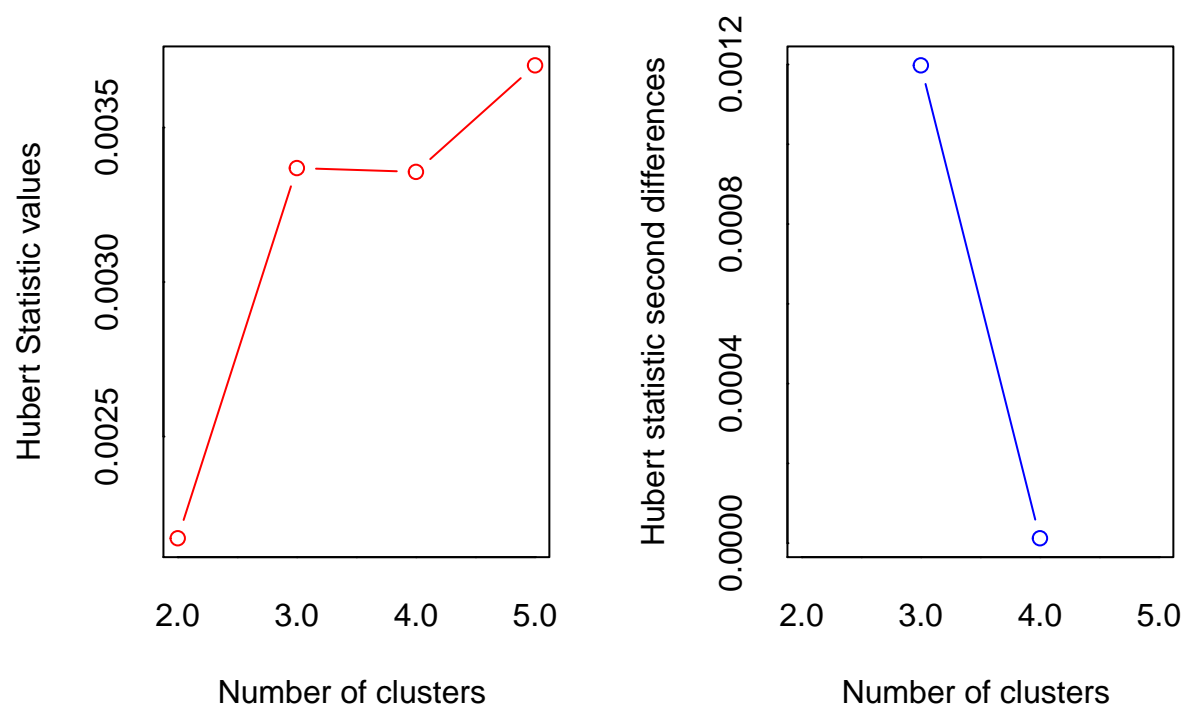
```
# Gap statistic
set.seed(42)
fviz_nbclust(ImputedDF, kmeans,
  nstart = 25,
  method = "gap_stat",
  nboot = 500
) + # reduce it for lower computation time (but less precise results)
labs(subtitle = "Gap statistic method")
```

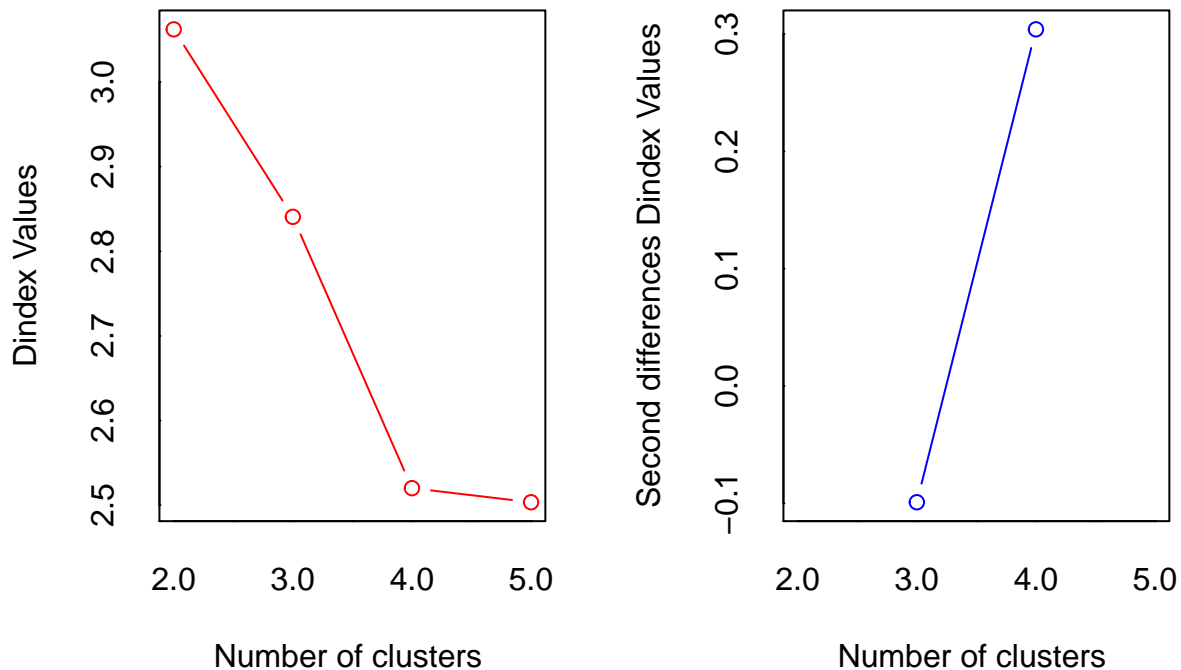
```
#The optimal number of clusters is the one that maximizes the gap statistic. This method suggests only 1 cluster.

#Three methods do not necessarily lead to the same result. Here, all 3 approaches suggest a different number of clusters.

#A fourth alternative is to use the NbClust() function, which provides 30 indices for choosing the best number of clusters.
nbclust_out <- NbClust(
  data = ImputedDF,
  distance = "euclidean",
  min.nc = 2, # minimum number of clusters
  max.nc = 5, # maximum number of clusters
  method = "kmeans" # one of: "ward.D", "ward.D2", "single", "complete", "average", "mcquitty", "median", "silhouette"
)
```



```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##       In the plot of Hubert index, we seek a significant knee that corresponds to a
##       significant increase of the value of the measure i.e the significant peak in Hubert
##       index second differences plot.
##
```

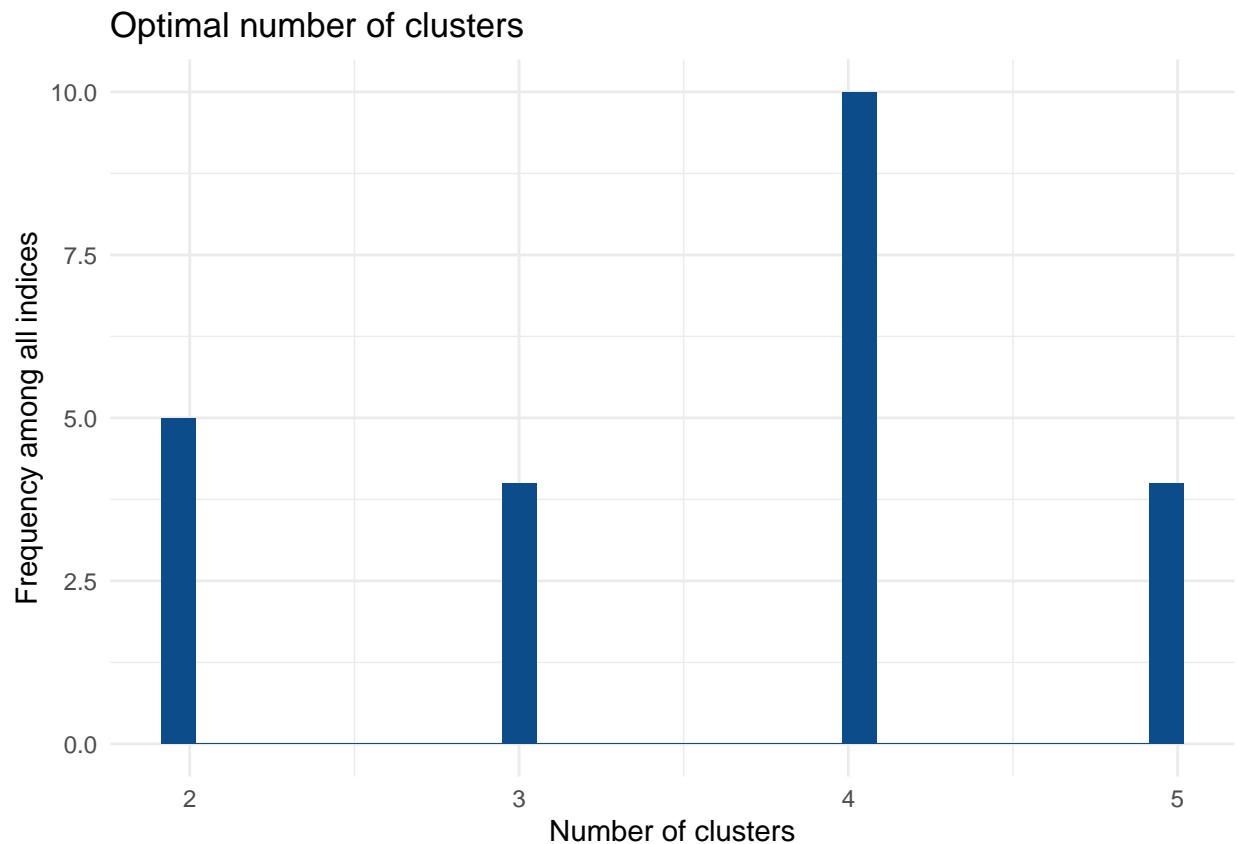


```
## *** : The D index is a graphical method of determining the number of clusters.
##           In the plot of D index, we seek a significant knee (the significant peak in Dindex
##           second differences plot) that corresponds to a significant increase of the value of
##           the measure.
##
## *****
## * Among all indices:
## * 5 proposed 2 as the best number of clusters
## * 4 proposed 3 as the best number of clusters
## * 10 proposed 4 as the best number of clusters
## * 4 proposed 5 as the best number of clusters
##
##           ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is 4
##
## *****
```

```
# create a dataframe of the optimal number of clusters
nbclust_plot <- data.frame(clusters = nbclust_out$Best.nc[1, ])
# select only indices which select between 2 and 5 clusters
nbclust_plot <- subset(nbclust_plot, clusters >= 2 & clusters <= 5)

# create plot
```

```
ggplot(nbclust_plot) +
  aes(x = clusters) +
  geom_histogram(bins = 30L, fill = "#0c4c8a") +
  labs(x = "Number of clusters", y = "Frequency among all indices", title = "Optimal number of clusters") +
  theme_minimal()
```



```
#Based on all 30 indices, the best number of clusters is 5 clusters.
# I would choose 5 clusters.
```

Q> Comment on the structure of the clusters and on their stability.

```
library(caret)
DF<-ImputedDF

# Cluster partition
trainDF<-DF[1:45,] # Partition A
testDF<-DF[46:77,] # Partition B
trainDF <- scale(trainDF)
testDF<-scale(testDF)

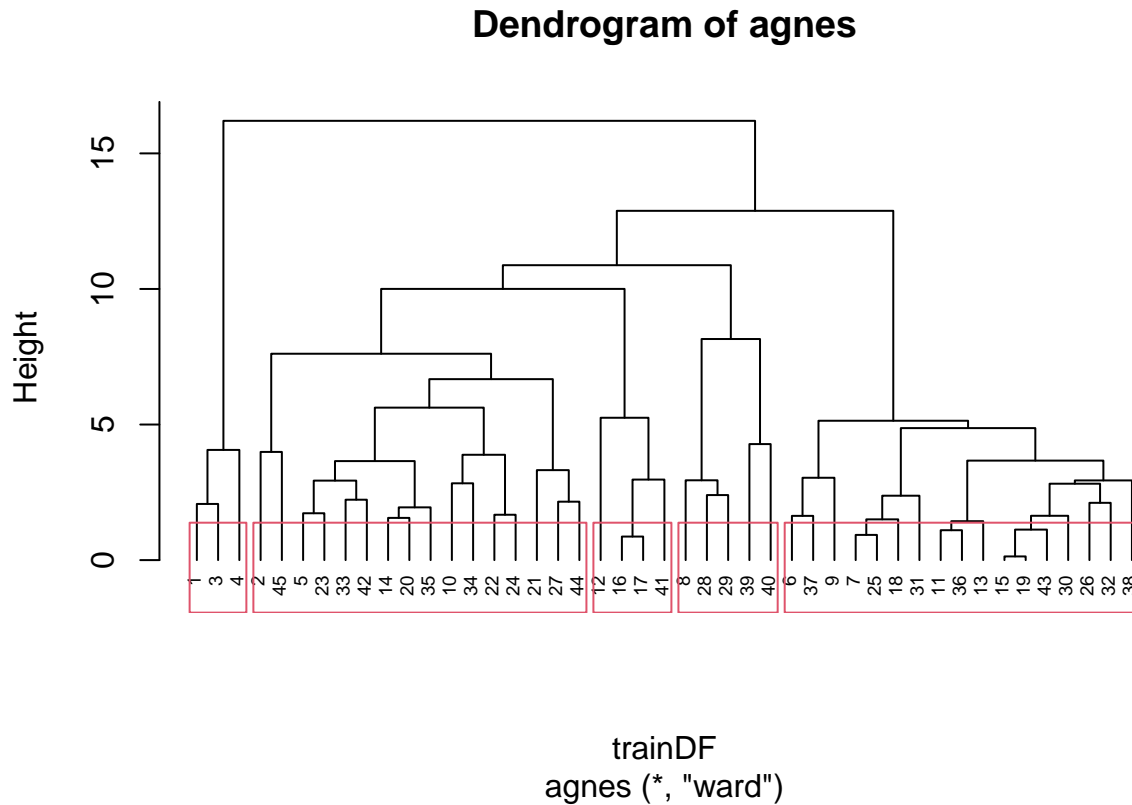
# The approach used earlier describes that ward is the best clustering mechanism
hc_train <- agnes(trainDF, method = "ward")

#visualize the dendrogram
```

```

pltree(hc_train, cex = 0.6, hang = -1, main = "Dendrogram of agnes")
rect.hclust(hc_train, k = 5)

```



```

CWcut <- cutree(hc_train, k = 5)
CWtotal <- as.data.frame( cbind(trainDF,CWcut))
head(CWtotal)

```

##	calories	protein	fat	sodium	fiber	carbo
## 1	-2.2902757	1.3899279	-0.1190407	-0.5218048	2.74720044	-2.27995823
## 2	0.7589079	0.4964028	3.4521808	-2.0963468	-0.07044104	-1.51239124
## 3	-2.2902757	1.3899279	-0.1190407	1.2581124	2.39499525	-1.76824691
## 4	-3.5099491	1.3899279	-1.0118461	-0.3848881	4.15602117	-1.51239124
## 5	0.1490712	-0.3971223	0.7737647	0.4366121	-0.42264622	0.02274273
## 6	0.1490712	-0.3971223	0.7737647	0.1627787	-0.24654363	-0.87275209

##	sugars	potass	vitamins	shelf	weight	cups
## 1	-0.44514757	2.45188569	-0.1330313	0.9770643	-0.3068967	-1.8861150
## 2	0.05564345	0.51266701	-1.6296338	0.9770643	-0.3068967	0.7128963
## 3	-0.69554308	2.98684257	-0.1330313	0.9770643	-0.3068967	-1.8861150
## 4	-1.94752061	3.12058179	-0.1330313	0.9770643	-0.3068967	-1.2266644
## 5	0.05564345	-0.08915948	-0.1330313	0.9770643	-0.3068967	-0.2568840
## 6	0.55643446	-0.35663792	-0.1330313	-1.6092824	-0.3068967	-0.2568840

##	rating	CWcut
## 1	1.9353757	1
## 2	-0.4509040	2
## 3	1.3129704	1

```
## 4 3.6895518 1
## 5 -0.4230914 2
## 6 -0.7610947 3
```

```
CWclust1 <- CWtotal[CWtotal$CWcut==1,]
colMeans(CWclust1)
```

```
## calories protein fat sodium fiber carbo sugars
## -2.6968335 1.3899279 -0.4166425 0.1171398 3.0994056 -1.8535321 -1.0294038
## potass vitamins shelf weight cups rating CWcut
## 2.8531034 -0.1330313 0.9770643 -0.3068967 -1.6662981 2.3126327 1.0000000
```

```
CWclust2 <- CWtotal[CWtotal$CWcut==2,]
colMeans(CWclust2)
```

```
## calories protein fat sodium fiber carbo
## 0.03472681 0.44055751 0.27156164 -0.62021364 0.01761026 0.15067056
## sugars potass vitamins shelf weight cups
## -0.50774645 0.15742221 -0.32010664 0.65377097 -0.30689671 -0.29325080
## rating CWcut
## 0.40553275 2.00000000
```

```
CWclust3 <- CWtotal[CWtotal$CWcut==3,]
colMeans(CWclust3)
```

```
## calories protein fat sodium fiber carbo sugars
## 0.1490712 -0.8701650 -0.1190407 0.1144552 -0.5262360 -0.3685659 0.8952049
## potass vitamins shelf weight cups rating CWcut
## -0.6359169 -0.1330313 -0.7725232 -0.3068967 0.2861930 -0.8097474 3.0000000
```

```
CWclust4 <- CWtotal[CWtotal$CWcut==4,]
colMeans(CWclust4)
```

```
## calories protein fat sodium fiber carbo
## 1.00284261 0.31769781 0.05952036 0.29969545 0.28176415 0.58562518
## sugars potass vitamins shelf weight cups
## 0.30603895 0.43242348 1.66289165 0.97706430 2.45517366 -0.18705985
## rating CWcut
## -0.14518909 4.00000000
```

```
CWclust5 <- CWtotal[CWtotal$CWcut==5,]
colMeans(CWclust5)
```

```
## calories protein fat sodium fiber carbo
## -0.003387982 0.496402831 -0.342242063 1.531945766 -0.510697517 1.621840611
## sugars potass vitamins shelf weight cups
## -1.384130721 -0.607398956 -0.133031332 -1.285989048 -0.306896708 1.440231546
## rating CWcut
## 0.266307344 5.000000000
```

```
CWmeans1 <- rbind(colMeans(CWclust1),colMeans(CWclust2),colMeans(CWclust3),colMeans(CWclust4) ,colMeans
head(CWmeans1)
```

```
##          calories      protein        fat      sodium      fiber      carbo
## [1,] -2.696833500  1.3899279 -0.41664251  0.1171398  3.09940562 -1.8535321
## [2,]  0.034726813  0.4405575  0.27156164 -0.6202136  0.01761026  0.1506706
## [3,]  0.149071198 -0.8701650 -0.11904072  0.1144552 -0.52623598 -0.3685659
## [4,]  1.002842608  0.3176978  0.05952036  0.2996954  0.28176415  0.5856252
## [5,] -0.003387982  0.4964028 -0.34224206  1.5319458 -0.51069752  1.6218406
##          sugars      potass      vitamins      shelf      weight      cups
## [1,] -1.0294038  2.8531034 -0.1330313  0.9770643 -0.3068967 -1.6662981
## [2,] -0.5077464  0.1574222 -0.3201066  0.6537710 -0.3068967 -0.2932508
## [3,]  0.8952049 -0.6359169 -0.1330313 -0.7725232 -0.3068967  0.2861930
## [4,]  0.3060390  0.4324235  1.6628917  0.9770643  2.4551737 -0.1870599
## [5,] -1.3841307 -0.6073990 -0.1330313 -1.2859890 -0.3068967  1.4402315
##          rating CWcut
## [1,]  2.3126327    1
## [2,]  0.4055327    2
## [3,] -0.8097474    3
## [4,] -0.1451891    4
## [5,]  0.2663073    5
```

Q> 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?

```
#install.packages("hrbrthemes")
library(GGally)
```

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

```
library(ggplot2)
library(hrbrthemes)
```

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

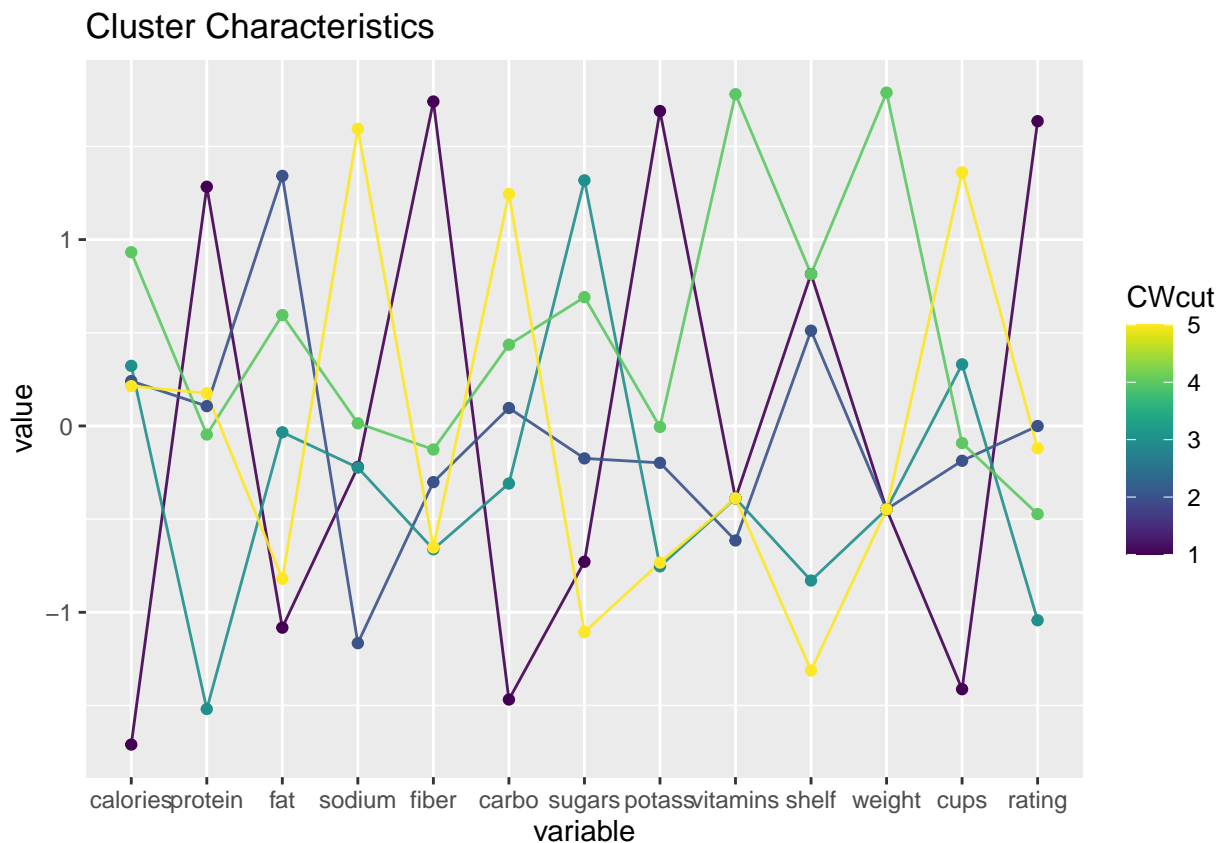
```
##           Please use hrbrthemes::import_roboto_condensed() to install Roboto Condensed and
```

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

```
library(viridis)
```

```
## Loading required package: viridisLite
```

```
#ggparcoord(cbind(c(1:4),CWmeans),columns = 2:14,groupColumn = 1,showPoints = TRUE,title = " Charter of
ggparcoord(CWmeans1,
  columns = 1:13, groupColumn = 14,
  showPoints = TRUE,
  title = "Cluster Characteristics",
  alphaLines = 0.9
) +
  scale_color_viridis(discrete=FALSE)
```



*# Based on the characteristics of the cluster, it is clear that Cluster 1 is the strongest with low cal
 ###In general,when we use the distance metric algorithm the data should be normalized,because the data*

Q> How do you compare hierarchical clustering and k-means? What are they main advantages of hierarchical clustering compared to k-means?

Ans: Clustering is a subjective statistical analysis and there can be more than one appropriate algorithm, depending on the dataset at hand or the type of problem to be solved. So choosing between k-means and hierarchical clustering is not always easy. If the cluster size is known or if we know that there is a specific number of clusters in our dataset (for example if we would like to distinguish diseased and healthy patients depending on some characteristics but we do not know in which group patients belong to), we should probably opt for the k-means clustering as this technique is used when the number of groups is specified in advance. If the number of groups or clusters in the dataset is unknown (for instance in marketing when trying to distinguish clients without any prior belief on the number of different types of customers), then we should probably opt for the hierarchical clustering to determine in how many clusters the data should be divided.