Assignment5

Sindhu Vegiraju

25/04/2021

#Import Cereals dataset

Cereals<-read.csv("D:/MACHINE LEARNING/Assignment5/Cereals.csv")

#Q)Data Preprocessing. Remove all cereals with missing values.

#Removing missing values that might be present in the data

sum(is.na(Cereals))

## [1] 4

CS<-na.omit(Cereals)  
sum(is.na(CS)) #To verify if the missing values are removed

## [1] 0

CS1<-CS[,c(-1,-2,-3)]

#Scaling the data

CSF1<-scale(CS1)

#Q1)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.

#Applying hierarchical clustering to the data using Euclidean distance

#Dissimilarity matrix

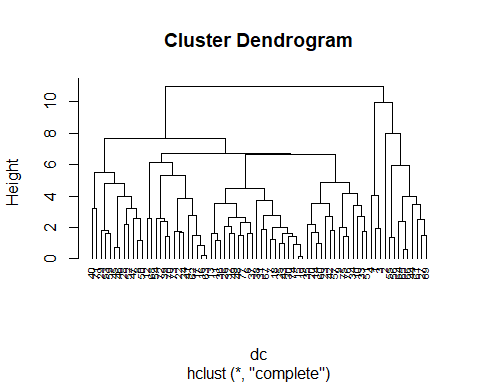
dc <- dist(CSF1, method = "euclidean")

#Hierarchical clustering using Complete Linkage

hc\_complete <- hclust(dc, method = "complete" )

#Plot the obtained dendogram

plot(hc\_complete, cex = 0.6, hang = -1)



#Use Agnes to compare the clustering from single linkage, complete linkage, average linkage, and Ward

c\_single<-agnes(CSF1,method= "single")  
c\_complete<-agnes(CSF1,method = "complete")  
c\_average<-agnes(CSF1,method = "average")  
c\_ward<-agnes(CSF1,method = "ward")

#Comparing the agglomerative coefficients of Single,complete,average and ward methods

c\_single$ac

## [1] 0.6067859

c\_complete$ac

## [1] 0.8353712

c\_average$ac

## [1] 0.7766075

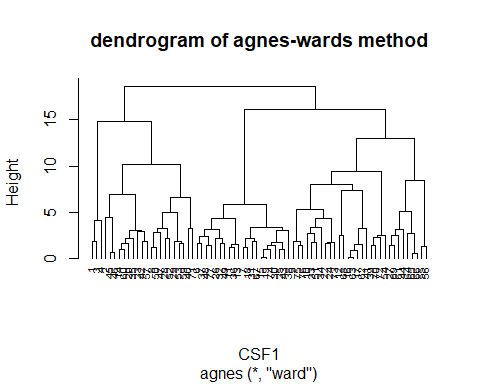
c\_ward$ac

## [1] 0.9046042

#By observing the above values we can say that the best linkage method is ward with agglomerative coefficient of 0.9046042

#visualizing the dendogram using wards method:

pl<-pltree(c\_ward,cex=0.6,hang=-1,main = "dendrogram of agnes-wards method")



#Q2)How many clusters would you choose?

#Create the distance matrix

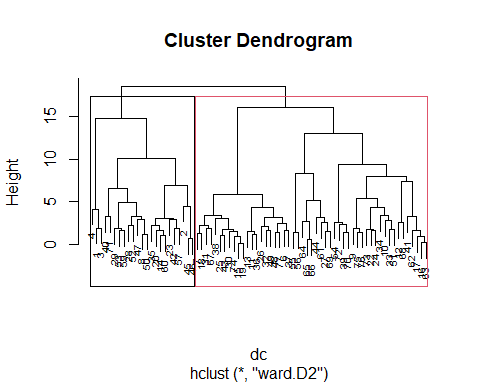
dc<-dist(CSF1,method = "euclidean")

#Wards method for Hierarchical clustering

w\_hc<-hclust(dc,method = "ward.D2")

#plotting dendrogram and taking k=2 by observing the distance

plot(w\_hc,cex=0.6)  
rect.hclust(w\_hc,k=2,border = 1:2)



#For identifying clusters, cut the dendrogram with cutree()

clust1<-cutree(w\_hc, k=2)

#Number of members in each cluster

table(clust1)

## clust1  
## 1 2   
## 23 51

#k=2 is cutting the longest path, so I choose k=2.

#Q3)Comment on the structure of the clusters and on their stability.

set.seed(123)  
Cereals\_New<-Cereals

#Removing any missing values that might be present in the data

nd<-na.omit(Cereals\_New)

#Divide the data and create partitions

p1<-nd[1:55,]  
p2<-nd[56:74,]

#Perform clustering using agnes() with single, complete, average and ward with partitioned data

r1<- agnes(scale(p1[,-c(1:3)]),method ="ward")  
r2<- agnes(scale(p1[,-c(1:3)]),method="average")  
r3<- agnes(scale(p1[,-c(1:3)]),method="complete")  
r4<- agnes(scale(p1[,-c(1:3)]),method="single")  
cbind(ward=r1$ac,average=r2$ac,complete=r3$ac,single=r4$ac)

## ward average complete single  
## [1,] 0.8808195 0.7449303 0.8120228 0.6564842

c2<-cutree(r1, k=2)

#Calculate the centers

cc<-as.data.frame(cbind(scale(p1[,-c(1:3)]),c2))  
center1<-colMeans(cc[cc$c2==1,])  
center2<-colMeans(cc[cc$c2==2,])

#Bind the 2 centers

centers<-rbind(center1,center2)  
centers

## calories protein fat sodium fiber carbo  
## center1 0.2870363 0.7424779 0.7043608 -0.18054158 0.7754394 -0.3921999  
## center2 -0.1514914 -0.3918634 -0.3717460 0.09528584 -0.4092597 0.2069944  
## sugars potass vitamins shelf weight cups  
## center1 0.11389860 0.9717483 -0.2193799 0.7766318 0.5853498 -0.7197837  
## center2 -0.06011315 -0.5128671 0.1157838 -0.4098890 -0.3089346 0.3798858  
## rating c2  
## center1 0.2618963 1  
## center2 -0.1382230 2

#Calculating Distance

x<-as.data.frame(rbind(centers[,-14],scale(p2[,-c(1:3)])))  
y1<-get\_dist(x)  
y2<-as.matrix(y1)  
d1<-data.frame(data=seq(1,nrow(p2),1),clusters=rep(0,nrow(p2)))  
for(i in 1:nrow(p2))  
{  
 d1[i,2]<-which.min(y2[i+2,1:2])  
}  
d1

## data clusters  
## 1 1 1  
## 2 2 1  
## 3 3 1  
## 4 4 2  
## 5 5 2  
## 6 6 2  
## 7 7 1  
## 8 8 2  
## 9 9 2  
## 10 10 2  
## 11 11 2  
## 12 12 2  
## 13 13 1  
## 14 14 1  
## 15 15 2  
## 16 16 2  
## 17 17 1  
## 18 18 2  
## 19 19 2

y3<-as.data.frame(cbind(CSF1,clust1))  
cbind(y3$clust1[56:74],d1$clusters)

## [,1] [,2]  
## [1,] 1 1  
## [2,] 1 1  
## [3,] 2 1  
## [4,] 2 2  
## [5,] 2 2  
## [6,] 2 2  
## [7,] 2 1  
## [8,] 2 2  
## [9,] 2 2  
## [10,] 2 2  
## [11,] 2 2  
## [12,] 2 2  
## [13,] 1 1  
## [14,] 2 1  
## [15,] 2 2  
## [16,] 2 2  
## [17,] 2 1  
## [18,] 2 2  
## [19,] 2 2

table(y3$clust1[56:74]==d1$clusters)

##   
## FALSE TRUE   
## 4 15

#Stability of the model: Accuracy = 78.94% (15/19)

#Q4)Healthy cereals

r<-cbind(nd,clust1)  
r[r$clust1==1,]

## name mfr type calories protein fat sodium  
## 1 100%\_Bran N C 70 4 1 130  
## 2 100%\_Natural\_Bran Q C 120 3 5 15  
## 3 All-Bran K C 70 4 1 260  
## 4 All-Bran\_with\_Extra\_Fiber K C 50 4 0 140  
## 8 Basic\_4 G C 130 3 2 210  
## 14 Clusters G C 110 3 2 140  
## 20 Cracklin'\_Oat\_Bran K C 110 3 3 140  
## 23 Crispy\_Wheat\_&\_Raisins G C 100 2 1 140  
## 28 Fruit\_&\_Fibre\_Dates,\_Walnuts,\_and\_Oats P C 120 3 2 160  
## 29 Fruitful\_Bran K C 120 3 0 240  
## 35 Great\_Grains\_Pecan P C 120 3 3 75  
## 40 Just\_Right\_Fruit\_&\_Nut K C 140 3 1 170  
## 42 Life Q C 100 4 2 150  
## 45 Muesli\_Raisins,\_Dates,\_&\_Almonds R C 150 4 3 95  
## 46 Muesli\_Raisins,\_Peaches,\_&\_Pecans R C 150 4 3 150  
## 47 Mueslix\_Crispy\_Blend K C 160 3 2 150  
## 50 Nutri-Grain\_Almond-Raisin K C 140 3 2 220  
## 52 Oatmeal\_Raisin\_Crisp G C 130 3 2 170  
## 53 Post\_Nat.\_Raisin\_Bran P C 120 3 1 200  
## 57 Quaker\_Oat\_Squares Q C 100 4 1 135  
## 59 Raisin\_Bran K C 120 3 1 210  
## 60 Raisin\_Nut\_Bran G C 100 3 2 140  
## 71 Total\_Raisin\_Bran G C 140 3 1 190  
## fiber carbo sugars potass vitamins shelf weight cups rating clust1  
## 1 10.0 5.0 6 280 25 3 1.00 0.33 68.40297 1  
## 2 2.0 8.0 8 135 0 3 1.00 1.00 33.98368 1  
## 3 9.0 7.0 5 320 25 3 1.00 0.33 59.42551 1  
## 4 14.0 8.0 0 330 25 3 1.00 0.50 93.70491 1  
## 8 2.0 18.0 8 100 25 3 1.33 0.75 37.03856 1  
## 14 2.0 13.0 7 105 25 3 1.00 0.50 40.40021 1  
## 20 4.0 10.0 7 160 25 3 1.00 0.50 40.44877 1  
## 23 2.0 11.0 10 120 25 3 1.00 0.75 36.17620 1  
## 28 5.0 12.0 10 200 25 3 1.25 0.67 40.91705 1  
## 29 5.0 14.0 12 190 25 3 1.33 0.67 41.01549 1  
## 35 3.0 13.0 4 100 25 3 1.00 0.33 45.81172 1  
## 40 2.0 20.0 9 95 100 3 1.30 0.75 36.47151 1  
## 42 2.0 12.0 6 95 25 2 1.00 0.67 45.32807 1  
## 45 3.0 16.0 11 170 25 3 1.00 1.00 37.13686 1  
## 46 3.0 16.0 11 170 25 3 1.00 1.00 34.13976 1  
## 47 3.0 17.0 13 160 25 3 1.50 0.67 30.31335 1  
## 50 3.0 21.0 7 130 25 3 1.33 0.67 40.69232 1  
## 52 1.5 13.5 10 120 25 3 1.25 0.50 30.45084 1  
## 53 6.0 11.0 14 260 25 3 1.33 0.67 37.84059 1  
## 57 2.0 14.0 6 110 25 3 1.00 0.50 49.51187 1  
## 59 5.0 14.0 12 240 25 2 1.33 0.75 39.25920 1  
## 60 2.5 10.5 8 140 25 3 1.00 0.50 39.70340 1  
## 71 4.0 15.0 14 230 100 3 1.50 1.00 28.59278 1

r[r$clust1==2,]

## name mfr type calories protein fat sodium fiber carbo  
## 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  
## 9 Bran\_Chex R C 90 2 1 200 4.0 15.0  
## 10 Bran\_Flakes P C 90 3 0 210 5.0 13.0  
## 11 Cap'n'Crunch Q C 120 1 2 220 0.0 12.0  
## 12 Cheerios G C 110 6 2 290 2.0 17.0  
## 13 Cinnamon\_Toast\_Crunch G C 120 1 3 210 0.0 13.0  
## 15 Cocoa\_Puffs G C 110 1 1 180 0.0 12.0  
## 16 Corn\_Chex R C 110 2 0 280 0.0 22.0  
## 17 Corn\_Flakes K C 100 2 0 290 1.0 21.0  
## 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  
## 22 Crispix K C 110 2 0 220 1.0 21.0  
## 24 Double\_Chex R C 100 2 0 190 1.0 18.0  
## 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  
## 27 Frosted\_Mini-Wheats K C 100 3 0 0 3.0 14.0  
## 30 Fruity\_Pebbles P C 110 1 1 135 0.0 13.0  
## 31 Golden\_Crisp P C 100 2 0 45 0.0 11.0  
## 32 Golden\_Grahams G C 110 1 1 280 0.0 15.0  
## 33 Grape\_Nuts\_Flakes P C 100 3 1 140 3.0 15.0  
## 34 Grape-Nuts P C 110 3 0 170 3.0 17.0  
## 36 Honey\_Graham\_Ohs Q C 120 1 2 220 1.0 12.0  
## 37 Honey\_Nut\_Cheerios G C 110 3 1 250 1.5 11.5  
## 38 Honey-comb P C 110 1 0 180 0.0 14.0  
## 39 Just\_Right\_Crunchy\_\_Nuggets K C 110 2 1 170 1.0 17.0  
## 41 Kix G C 110 2 1 260 0.0 21.0  
## 43 Lucky\_Charms G C 110 2 1 180 0.0 12.0  
## 44 Maypo A H 100 4 1 0 0.0 16.0  
## 48 Multi-Grain\_Cheerios G C 100 2 1 220 2.0 15.0  
## 49 Nut&Honey\_Crunch K C 120 2 1 190 0.0 15.0  
## 51 Nutri-grain\_Wheat K C 90 3 0 170 3.0 18.0  
## 54 Product\_19 K C 100 3 0 320 1.0 20.0  
## 55 Puffed\_Rice Q C 50 1 0 0 0.0 13.0  
## 56 Puffed\_Wheat Q C 50 2 0 0 1.0 10.0  
## 61 Raisin\_Squares K C 90 2 0 0 2.0 15.0  
## 62 Rice\_Chex R C 110 1 0 240 0.0 23.0  
## 63 Rice\_Krispies K C 110 2 0 290 0.0 22.0  
## 64 Shredded\_Wheat N C 80 2 0 0 3.0 16.0  
## 65 Shredded\_Wheat\_'n'Bran N C 90 3 0 0 4.0 19.0  
## 66 Shredded\_Wheat\_spoon\_size N C 90 3 0 0 3.0 20.0  
## 67 Smacks K C 110 2 1 70 1.0 9.0  
## 68 Special\_K K C 110 6 0 230 1.0 16.0  
## 69 Strawberry\_Fruit\_Wheats N C 90 2 0 15 3.0 15.0  
## 70 Total\_Corn\_Flakes G C 110 2 1 200 0.0 21.0  
## 72 Total\_Whole\_Grain G C 100 3 1 200 3.0 16.0  
## 73 Triples G C 110 2 1 250 0.0 21.0  
## 74 Trix G C 110 1 1 140 0.0 13.0  
## 75 Wheat\_Chex R C 100 3 1 230 3.0 17.0  
## 76 Wheaties G C 100 3 1 200 3.0 17.0  
## 77 Wheaties\_Honey\_Gold G C 110 2 1 200 1.0 16.0  
## sugars potass vitamins shelf weight cups rating clust1  
## 6 10 70 25 1 1.00 0.75 29.50954 2  
## 7 14 30 25 2 1.00 1.00 33.17409 2  
## 9 6 125 25 1 1.00 0.67 49.12025 2  
## 10 5 190 25 3 1.00 0.67 53.31381 2  
## 11 12 35 25 2 1.00 0.75 18.04285 2  
## 12 1 105 25 1 1.00 1.25 50.76500 2  
## 13 9 45 25 2 1.00 0.75 19.82357 2  
## 15 13 55 25 2 1.00 1.00 22.73645 2  
## 16 3 25 25 1 1.00 1.00 41.44502 2  
## 17 2 35 25 1 1.00 1.00 45.86332 2  
## 18 12 20 25 2 1.00 1.00 35.78279 2  
## 19 13 65 25 2 1.00 1.00 22.39651 2  
## 22 3 30 25 3 1.00 1.00 46.89564 2  
## 24 5 80 25 3 1.00 0.75 44.33086 2  
## 25 13 30 25 2 1.00 1.00 32.20758 2  
## 26 11 25 25 1 1.00 0.75 31.43597 2  
## 27 7 100 25 2 1.00 0.80 58.34514 2  
## 30 12 25 25 2 1.00 0.75 28.02576 2  
## 31 15 40 25 1 1.00 0.88 35.25244 2  
## 32 9 45 25 2 1.00 0.75 23.80404 2  
## 33 5 85 25 3 1.00 0.88 52.07690 2  
## 34 3 90 25 3 1.00 0.25 53.37101 2  
## 36 11 45 25 2 1.00 1.00 21.87129 2  
## 37 10 90 25 1 1.00 0.75 31.07222 2  
## 38 11 35 25 1 1.00 1.33 28.74241 2  
## 39 6 60 100 3 1.00 1.00 36.52368 2  
## 41 3 40 25 2 1.00 1.50 39.24111 2  
## 43 12 55 25 2 1.00 1.00 26.73451 2  
## 44 3 95 25 2 1.00 1.00 54.85092 2  
## 48 6 90 25 1 1.00 1.00 40.10596 2  
## 49 9 40 25 2 1.00 0.67 29.92429 2  
## 51 2 90 25 3 1.00 1.00 59.64284 2  
## 54 3 45 100 3 1.00 1.00 41.50354 2  
## 55 0 15 0 3 0.50 1.00 60.75611 2  
## 56 0 50 0 3 0.50 1.00 63.00565 2  
## 61 6 110 25 3 1.00 0.50 55.33314 2  
## 62 2 30 25 1 1.00 1.13 41.99893 2  
## 63 3 35 25 1 1.00 1.00 40.56016 2  
## 64 0 95 0 1 0.83 1.00 68.23588 2  
## 65 0 140 0 1 1.00 0.67 74.47295 2  
## 66 0 120 0 1 1.00 0.67 72.80179 2  
## 67 15 40 25 2 1.00 0.75 31.23005 2  
## 68 3 55 25 1 1.00 1.00 53.13132 2  
## 69 5 90 25 2 1.00 1.00 59.36399 2  
## 70 3 35 100 3 1.00 1.00 38.83975 2  
## 72 3 110 100 3 1.00 1.00 46.65884 2  
## 73 3 60 25 3 1.00 0.75 39.10617 2  
## 74 12 25 25 2 1.00 1.00 27.75330 2  
## 75 3 115 25 1 1.00 0.67 49.78744 2  
## 76 3 110 25 1 1.00 1.00 51.59219 2  
## 77 8 60 25 1 1.00 0.75 36.18756 2

#Calculating mean ratings to determine the best cluster.

mean(r[r$clust1==1,"rating"])

## [1] 42.90285

mean(r[r$clust1==2,"rating"])

## [1] 42.13229

#Cluster 1 has high rating values, So we can infer this cluster has more nutrition values. #Since we using distance metric algorithm we essentially need to normalize data, as the features of data are different, hence we need to scale it to similar features. #Also, Normalization is used to eliminate redundant data and ensures that good quality clusters are generated which can improve the clustering algorithm. So we need to do normalization.