hw5

library(readr)  
library(cluster)  
library(dplyr)

##   
## 载入程辑包：'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(caret)

## 载入需要的程辑包：lattice

## 载入需要的程辑包：ggplot2

library(factoextra)

## Warning: 程辑包'factoextra'是用R版本4.1.1 来建造的

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

Cereals <- read\_csv("C:/Users/Administrator/Desktop/Cereals.csv")

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## name = col\_character(),  
## mfr = col\_character(),  
## type = col\_character(),  
## calories = col\_double(),  
## protein = col\_double(),  
## fat = col\_double(),  
## sodium = col\_double(),  
## fiber = col\_double(),  
## carbo = col\_double(),  
## sugars = col\_double(),  
## potass = col\_double(),  
## vitamins = col\_double(),  
## shelf = col\_double(),  
## weight = col\_double(),  
## cups = col\_double(),  
## rating = col\_double()  
## )

# Remove the missing values.  
data <- na.omit(Cereals)  
#check the missing values.  
head(data)

## # A tibble: 6 x 16  
## name mfr type calories protein fat sodium fiber carbo sugars potass  
## <chr> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 100%\_Bran N C 70 4 1 130 10 5 6 280  
## 2 100%\_Natu~ Q C 120 3 5 15 2 8 8 135  
## 3 All-Bran K C 70 4 1 260 9 7 5 320  
## 4 All-Bran\_~ K C 50 4 0 140 14 8 0 330  
## 5 Apple\_Cin~ G C 110 2 2 180 1.5 10.5 10 70  
## 6 Apple\_Jac~ K C 110 2 0 125 1 11 14 30  
## # ... with 5 more variables: vitamins <dbl>, shelf <dbl>, weight <dbl>,  
## # cups <dbl>, rating <dbl>

##1  
#use preprocess to scale the data for clustering later.  
norm\_data<- preProcess(data[4:16], method=c("center", "scale"))  
norm\_data<-predict(norm\_data,data[4:16])  
# get the Euclidean distance   
distance <- dist(norm\_data[1:13], method = "euclidean")  
#single linkage, complete linkage, average linkage, and Ward.  
hc\_single <- agnes(norm\_data, method = "single")  
hc\_complete <- agnes(norm\_data, method = "complete")  
hc\_average <- agnes(norm\_data, method = "average")  
hc\_ward <- agnes(norm\_data, method = "ward")  
#use plot to see the results of the coefficients,highest coefficients means best methods.  
print(hc\_single) #0.6067

## Call: agnes(x = norm\_data, method = "single")   
## Agglomerative coefficient: 0.6067859   
## Order of objects:  
## [1] 1 3 4 2 5 35 6 14 18 71 41 23 28 17 10 34 12 64 46 74 47 8 72 73 30  
## [26] 24 29 36 7 48 50 26 27 51 56 13 57 19 55 33 40 21 31 49 20 22 70 32 15 60  
## [51] 16 59 9 25 66 58 42 61 62 63 39 45 11 65 43 44 37 67 69 52 38 68 53 54  
## Height (summary):  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.1431 1.3777 1.7695 1.8668 2.2787 4.0361   
##   
## Available components:  
## [1] "order" "height" "ac" "merge" "diss" "call" "method" "data"

print(hc\_complete)#0.8353

## Call: agnes(x = norm\_data, method = "complete")   
## Agglomerative coefficient: 0.8353712   
## Order of objects:  
## [1] 1 3 4 2 25 66 58 42 61 62 63 53 54 5 35 46 74 24 30 47 10 34 12 6 17  
## [26] 14 18 71 28 23 41 29 64 36 8 72 73 9 31 49 32 13 57 19 33 21 40 55 11 65  
## [51] 15 60 16 59 39 20 22 70 37 67 69 52 7 48 45 26 50 43 44 27 51 56 38 68  
## Height (summary):  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.1431 1.6076 2.3389 2.9321 3.7169 10.9839   
##   
## Available components:  
## [1] "order" "height" "ac" "merge" "diss" "call" "method" "data"

print(hc\_average) #0.7766

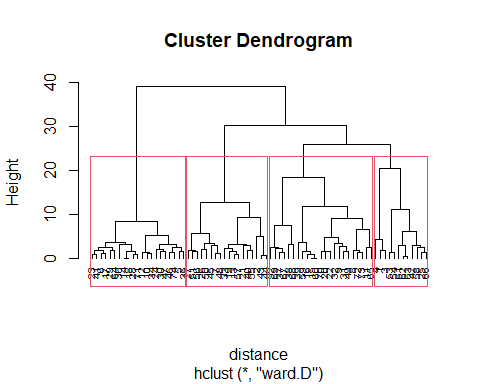
## Call: agnes(x = norm\_data, method = "average")   
## Agglomerative coefficient: 0.7766075   
## Order of objects:  
## [1] 1 3 4 2 5 35 46 74 24 30 47 6 17 14 18 71 23 41 28 29 64 10 34 12 36  
## [26] 8 72 73 9 32 20 22 70 31 49 13 57 19 33 40 55 21 15 60 16 59 39 25 66 58  
## [51] 42 61 62 63 7 48 50 45 26 27 51 56 43 44 37 67 69 52 38 68 11 65 53 54  
## Height (summary):  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.1431 1.4633 2.0666 2.4461 2.9445 7.7243   
##   
## Available components:  
## [1] "order" "height" "ac" "merge" "diss" "call" "method" "data"

print(hc\_ward) #0.9046

## Call: agnes(x = norm\_data, method = "ward")   
## Agglomerative coefficient: 0.9046042   
## Order of objects:  
## [1] 1 3 4 2 43 44 13 57 19 33 21 40 55 7 48 45 26 50 27 51 56 38 68 5 35  
## [26] 46 74 24 30 47 10 34 12 6 17 29 64 14 18 71 28 23 41 36 8 72 73 9 31 49  
## [51] 32 20 22 70 11 65 15 60 16 59 39 37 67 69 52 25 66 58 42 61 62 63 53 54  
## Height (summary):  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.1431 1.5858 2.3422 3.6092 4.1559 18.5749   
##   
## Available components:  
## [1] "order" "height" "ac" "merge" "diss" "call" "method" "data"

# ward agglomerative coefficient is the highest, so this is the best method.

##2   
#use hclust to visualize the ward linkage.  
best\_cluster<-hclust(distance,method = "ward.D")  
plot(best\_cluster, cex = 0.6, hang=-1)  
rect.hclust(best\_cluster,k=4)



label<-cutree(hc\_ward, 6)  
label

## [1] 1 2 1 1 3 3 4 5 5 3 5 3 2 3 5 5 3 3 2 5 2 5 3 3 6 4 4 3 3 3 5 5 2 3 3 3 5 4  
## [39] 5 2 3 6 2 2 4 3 3 4 5 4 4 5 6 6 2 4 2 6 5 5 6 6 6 3 5 6 5 4 5 5 3 5 5 3

#from the plot 4 clusters i would choose.

##3  
#separate the data to two parts, 50%A and50% B  
set.seed(1234)  
train\_ID<-sample(1:nrow(norm\_data),0.5\*nrow(norm\_data))  
train<-norm\_data[train\_ID,]#a  
test<-norm\_data[-train\_ID,]#b  
#use cutree to cluster to 4 parts and hclust to clustering by distance.then calculate the distances of points to the every cluster's central points.  
global\_clust<-cutree(hclust(distance,method = "ward.D"),k=4)  
global\_cent<-aggregate(.~global\_clust,data=norm\_data,FUN=mean)  
  
train\_clust<-cutree(hclust(dist(train),method = "ward.D"),k=4)  
train\_cent<-aggregate(.~train\_clust,data=train,FUN=mean)  
  
#The first loop means calculate all the distances of points to the central point, second loop is calculate the train's distances of points to the central point.  
#think about every cluster is a circle. use sum((x-x.clust)^2 calculate each point to central point.  
test\_dist<-apply(test,1,function(x) {  
 apply(train\_cent[,-1],1,function(x.clust) {  
 sum((x-x.clust)^2)  
 })  
})  
test\_clust<-apply(test\_dist,2,which.min)  
  
#through global\_cluster calculate the label then corresponding all the data label.Through this way to get the approximate results.  
label<-as.numeric(as.character((factor(test\_clust,levels=order(apply(train\_cent[,-1]^2,1,sum)),  
 labels = order(apply(global\_cent[,-1]^2,1,sum))))))   
  
table(global\_clust[-train\_ID],label)

## label  
## 1 2 3 4  
## 1 1 0 4 0  
## 2 0 0 0 11  
## 3 0 9 1 0  
## 4 0 0 8 3

# from the table i found out the cluster Assignments don't work well,only 2 cluster working well. 4 cluster is too much for this data.

##4 #The data should not be normalized, cause the concrete values are used as results we should make is meaningful,normalized is a method for comprehensive evaluation of multiple data in order to eliminate the effect of variable units.After normalized the result can be negative then we can’t understand what’s the actually meaning.