Clustering Assignment\_Woodruff

Eric Woodruff

March 3, 2019

options(tidyverse.quiet=TRUE)  
library(tidyverse)  
library(cluster) #algorithms for clustering  
library(factoextra) #visualization

## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/13EFCZ

library(dendextend) #viewing clustering dendograms

##   
## ---------------------  
## Welcome to dendextend version 1.9.0  
## Type citation('dendextend') for how to cite the package.  
##   
## Type browseVignettes(package = 'dendextend') for the package vignette.  
## The github page is: https://github.com/talgalili/dendextend/  
##   
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues  
## Or contact: <tal.galili@gmail.com>  
##   
## To suppress this message use: suppressPackageStartupMessages(library(dendextend))  
## ---------------------

##   
## Attaching package: 'dendextend'

## The following object is masked from 'package:stats':  
##   
## cutree

trucks = read\_csv("trucks.csv")

## Parsed with column specification:  
## cols(  
## Driver\_ID = col\_double(),  
## Distance = col\_double(),  
## Speeding = col\_integer()  
## )

str(trucks)

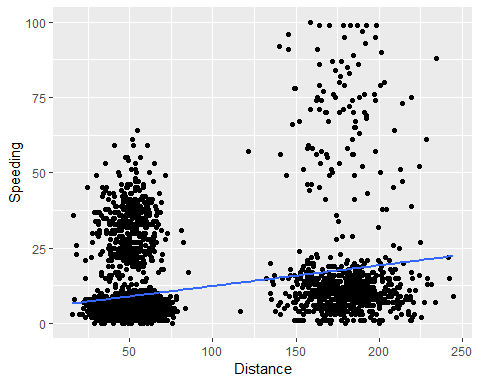
## Classes 'tbl\_df', 'tbl' and 'data.frame': 4000 obs. of 3 variables:  
## $ Driver\_ID: num 3.42e+09 3.42e+09 3.42e+09 3.42e+09 3.42e+09 ...  
## $ Distance : num 71.2 52.5 64.5 55.7 54.6 ...  
## $ Speeding : int 28 25 27 22 25 10 20 8 34 19 ...  
## - attr(\*, "spec")=List of 2  
## ..$ cols :List of 3  
## .. ..$ Driver\_ID: list()  
## .. .. ..- attr(\*, "class")= chr "collector\_double" "collector"  
## .. ..$ Distance : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_double" "collector"  
## .. ..$ Speeding : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_integer" "collector"  
## ..$ default: list()  
## .. ..- attr(\*, "class")= chr "collector\_guess" "collector"  
## ..- attr(\*, "class")= chr "col\_spec"

summary(trucks)

## Driver\_ID Distance Speeding   
## Min. :3.423e+09 Min. : 15.52 Min. : 0.00   
## 1st Qu.:3.423e+09 1st Qu.: 45.25 1st Qu.: 4.00   
## Median :3.423e+09 Median : 53.33 Median : 6.00   
## Mean :3.423e+09 Mean : 76.04 Mean : 10.72   
## 3rd Qu.:3.423e+09 3rd Qu.: 65.63 3rd Qu.: 9.00   
## Max. :3.423e+09 Max. :244.79 Max. :100.00

Task 1: Plot relationship between Distance and Speeding. Describe this relationship, does there appear to be any natural clustering of drivers?

ggplot(trucks, aes(x=Distance, y=Speeding)) +geom\_point() + geom\_smooth(method="lm", se= FALSE)



There most certainly does appear to be natural clustering of the drivers. As the distance drivens increases, it appears that the time spent speeding increases. This is further represented by the regression line which displays a slight increase.

Task 2: Create a new data frame (called trucks2) that excludes the Driver\_ID variable and includes scaled versions of the Distance and Speeding variables. NOTE: Wrap the scale(trucks2) command in an as.data.frame command to ensure that the resulting object is a data frame. By default, scale converts dataframes to lists.

trucks2 = trucks %>% select("Distance", "Speeding")  
str(trucks2)

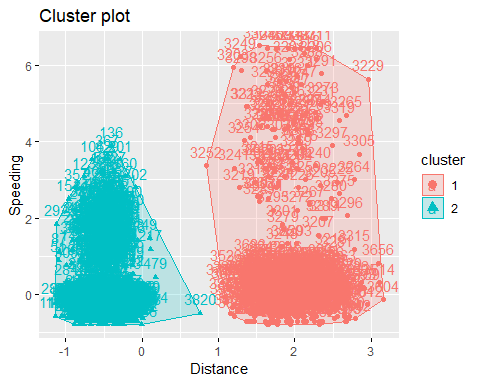
## Classes 'tbl\_df', 'tbl' and 'data.frame': 4000 obs. of 2 variables:  
## $ Distance: num 71.2 52.5 64.5 55.7 54.6 ...  
## $ Speeding: int 28 25 27 22 25 10 20 8 34 19 ...  
## - attr(\*, "spec")=List of 2  
## ..$ cols :List of 3  
## .. ..$ Driver\_ID: list()  
## .. .. ..- attr(\*, "class")= chr "collector\_double" "collector"  
## .. ..$ Distance : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_double" "collector"  
## .. ..$ Speeding : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_integer" "collector"  
## ..$ default: list()  
## .. ..- attr(\*, "class")= chr "collector\_guess" "collector"  
## ..- attr(\*, "class")= chr "col\_spec"

trucks2 = data.frame(scale(trucks2))  
summary(trucks2)

## Distance Speeding   
## Min. :-1.1319 Min. :-0.7821   
## 1st Qu.:-0.5759 1st Qu.:-0.4903   
## Median :-0.4248 Median :-0.3444   
## Mean : 0.0000 Mean : 0.0000   
## 3rd Qu.:-0.1947 3rd Qu.:-0.1255   
## Max. : 3.1560 Max. : 6.5127

Task 3: Use k-Means clustering with two clusters (k=2) to cluster the trucks 2 data frame. Use a random number seed of 1234. Visualize the clusters using the fviz\_cluster function. Comment on the clusters.

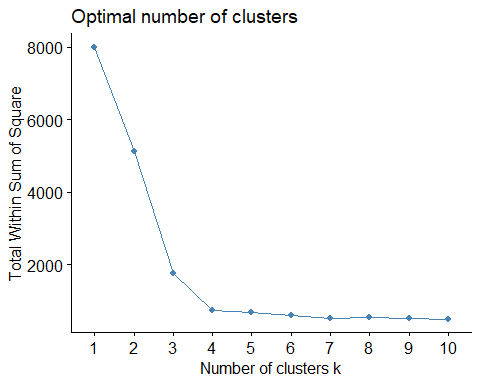
set.seed(1234)  
clusters = kmeans(trucks2, 2)  
fviz\_cluster(clusters, trucks2)



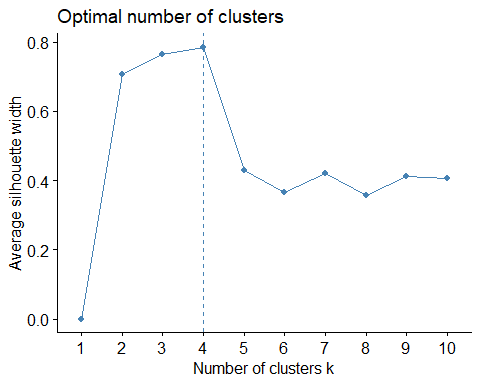
Above you see the 2 clusters. You could argue that there are four groups above. Both clusters could be split, cluster 1 could be split between the tightly grouped variables at the bottom and the more spread variables at the top end of the cluster above speeding 2. Cluster 2 could arguably be split into two groups as well, the first group being the tightly grouped variables at the bottom somewhere inbetween speeding o and 1, and the second group being the variables above group one.

Task 4: Use the two methods from the k-Means lecture to identify the optimal number of clusters. Use a random number seed of 123 for these methods. Is there consensus between these two methods as the optimal number of clusters?

set.seed(123)  
fviz\_nbclust(trucks2, kmeans, method = "wss") #minimize within-cluster variation



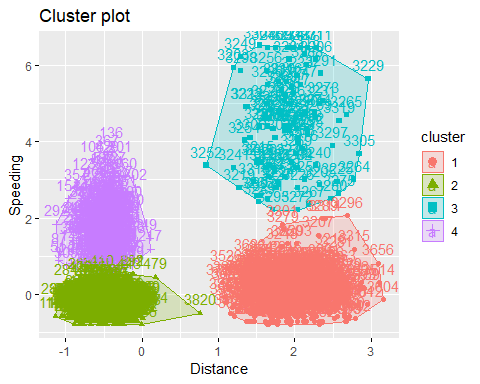
set.seed(123)  
fviz\_nbclust(trucks2, kmeans, method = "silhouette") #maximize how well points sit in their clusters



The first method used wss, to minimize within-cluster variation, we can see the bend at 3 or 4.  
The second method used silhouette, to maximize how well points sit in their clusters, highlights 4 as the optimal number of clusters.  
I would say yes, there is a consensus between these two methods at 4 for the optimal number of clusters.

Task 5: Use the optimal number of clusters that you identified in Task 4 to create k-Means clusters. Use a random number seed of 1234. Use the fviz\_cluster function to visualize the clusters.

set.seed(1234)  
clusters2 = kmeans(trucks2, 4)  
fviz\_cluster(clusters2, trucks2)



Task 6: In words, how would you characterize the clusters you created in Task 5?  
I was pretty spot on in my break down from Task 3 where I predicted how the data could be further clustered. I think using four clusters gives the best result for the data set.

Pre Task 7: Import new data set and create a new dataframe called wine2 that removes the Year and FrancePop varaibles and scales the remaining variables.

wine = read\_csv("wineprice.csv")

## Parsed with column specification:  
## cols(  
## Year = col\_integer(),  
## Price = col\_double(),  
## WinterRain = col\_integer(),  
## AGST = col\_double(),  
## HarvestRain = col\_integer(),  
## Age = col\_integer(),  
## FrancePop = col\_double()  
## )

str(wine)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 25 obs. of 7 variables:  
## $ Year : int 1952 1953 1955 1957 1958 1959 1960 1961 1962 1963 ...  
## $ Price : num 7.5 8.04 7.69 6.98 6.78 ...  
## $ WinterRain : int 600 690 502 420 582 485 763 830 697 608 ...  
## $ AGST : num 17.1 16.7 17.1 16.1 16.4 ...  
## $ HarvestRain: int 160 80 130 110 187 187 290 38 52 155 ...  
## $ Age : int 31 30 28 26 25 24 23 22 21 20 ...  
## $ FrancePop : num 43184 43495 44218 45152 45654 ...  
## - attr(\*, "spec")=List of 2  
## ..$ cols :List of 7  
## .. ..$ Year : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_integer" "collector"  
## .. ..$ Price : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_double" "collector"  
## .. ..$ WinterRain : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_integer" "collector"  
## .. ..$ AGST : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_double" "collector"  
## .. ..$ HarvestRain: list()  
## .. .. ..- attr(\*, "class")= chr "collector\_integer" "collector"  
## .. ..$ Age : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_integer" "collector"  
## .. ..$ FrancePop : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_double" "collector"  
## ..$ default: list()  
## .. ..- attr(\*, "class")= chr "collector\_guess" "collector"  
## ..- attr(\*, "class")= chr "col\_spec"

summary(wine)

## Year Price WinterRain AGST   
## Min. :1952 Min. :6.205 Min. :376.0 Min. :14.98   
## 1st Qu.:1960 1st Qu.:6.519 1st Qu.:536.0 1st Qu.:16.20   
## Median :1966 Median :7.121 Median :600.0 Median :16.53   
## Mean :1966 Mean :7.067 Mean :605.3 Mean :16.51   
## 3rd Qu.:1972 3rd Qu.:7.495 3rd Qu.:697.0 3rd Qu.:17.07   
## Max. :1978 Max. :8.494 Max. :830.0 Max. :17.65   
## HarvestRain Age FrancePop   
## Min. : 38.0 Min. : 5.0 Min. :43184   
## 1st Qu.: 89.0 1st Qu.:11.0 1st Qu.:46584   
## Median :130.0 Median :17.0 Median :50255   
## Mean :148.6 Mean :17.2 Mean :49694   
## 3rd Qu.:187.0 3rd Qu.:23.0 3rd Qu.:52894   
## Max. :292.0 Max. :31.0 Max. :54602

wine2 = wine %>% select("Price", "WinterRain", "AGST", "HarvestRain", "Age")  
str(wine2)

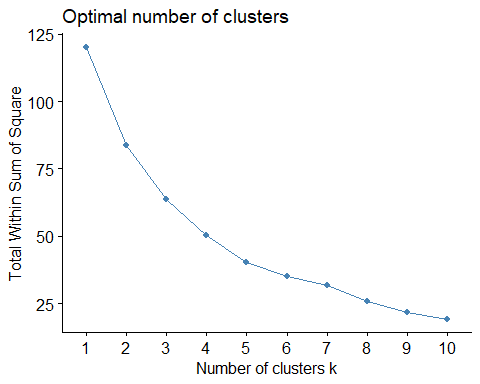
## Classes 'tbl\_df', 'tbl' and 'data.frame': 25 obs. of 5 variables:  
## $ Price : num 7.5 8.04 7.69 6.98 6.78 ...  
## $ WinterRain : int 600 690 502 420 582 485 763 830 697 608 ...  
## $ AGST : num 17.1 16.7 17.1 16.1 16.4 ...  
## $ HarvestRain: int 160 80 130 110 187 187 290 38 52 155 ...  
## $ Age : int 31 30 28 26 25 24 23 22 21 20 ...  
## - attr(\*, "spec")=List of 2  
## ..$ cols :List of 7  
## .. ..$ Year : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_integer" "collector"  
## .. ..$ Price : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_double" "collector"  
## .. ..$ WinterRain : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_integer" "collector"  
## .. ..$ AGST : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_double" "collector"  
## .. ..$ HarvestRain: list()  
## .. .. ..- attr(\*, "class")= chr "collector\_integer" "collector"  
## .. ..$ Age : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_integer" "collector"  
## .. ..$ FrancePop : list()  
## .. .. ..- attr(\*, "class")= chr "collector\_double" "collector"  
## ..$ default: list()  
## .. ..- attr(\*, "class")= chr "collector\_guess" "collector"  
## ..- attr(\*, "class")= chr "col\_spec"

wine2 = data.frame(scale(wine2))  
summary(wine2)

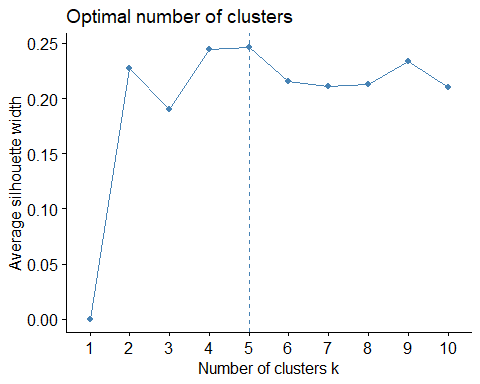
## Price WinterRain AGST   
## Min. :-1.32596 Min. :-1.73332 Min. :-2.25947   
## 1st Qu.:-0.84329 1st Qu.:-0.52375 1st Qu.:-0.45801   
## Median : 0.08284 Median :-0.03992 Median : 0.03548   
## Mean : 0.00000 Mean : 0.00000 Mean : 0.00000   
## 3rd Qu.: 0.65777 3rd Qu.: 0.69339 3rd Qu.: 0.82524   
## Max. : 2.19343 Max. : 1.69885 Max. : 1.68888   
## HarvestRain Age   
## Min. :-1.4856 Min. :-1.586   
## 1st Qu.:-0.8003 1st Qu.:-0.806   
## Median :-0.2494 Median :-0.026   
## Mean : 0.0000 Mean : 0.000   
## 3rd Qu.: 0.5165 3rd Qu.: 0.754   
## Max. : 1.9275 Max. : 1.794

Task 7: Use the two methods from Task 4 to determine the optimal number of k-Means clusters for this data. Use a random number seed of 123. Is there consensus between these two methods as the optimal number of clusters?

set.seed(123)  
fviz\_nbclust(wine2, kmeans, method = "wss") #minimize within-cluster variation



set.seed(123)  
fviz\_nbclust(wine2, kmeans, method = "silhouette") #maximize how well points sit in their clusters

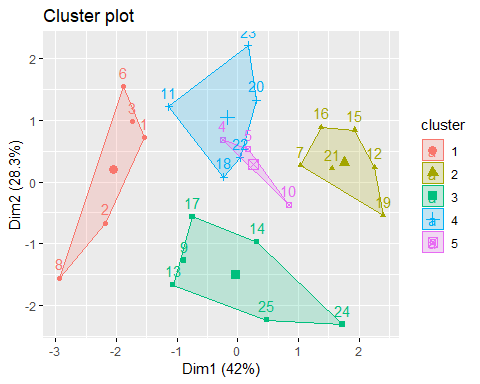


The first method, wss, is hard to pinpont a good optimal point, there is no real elbow. You could argue that there is a slight bend at 7, but it is very slight. Based on this method alone I’d choose somewhere between 5 and 7 because it is in the mid range, but its not concrete by any means.

The second method, silhouette, pinpoints an optimal cluster of 5, which is close to that very slight elbow mentioned above at 7. I can’t say these two reach a consensus however because the first method isn’t that definite.

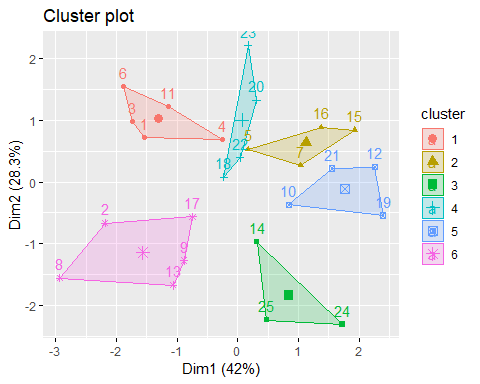
Task 8: Use the optimal number of clusters that you identified in Task 4 to create k-Means clusters. Use a random number seed of 1234. Use the fviz\_cluster function to visualize the clusters.

set.seed(1234)  
winecluster = kmeans(wine2, 5)  
fviz\_cluster(winecluster, wine2)



I don’t like cluster 4 and 5 overlapping so I am going to try using 6 clusters as opposed to 5 to how that changes.

set.seed(1234)  
winecluster2 = kmeans(wine2, 6)  
fviz\_cluster(winecluster2, wine2)



I like the 6 clusters as opposed to only 5 clusters.

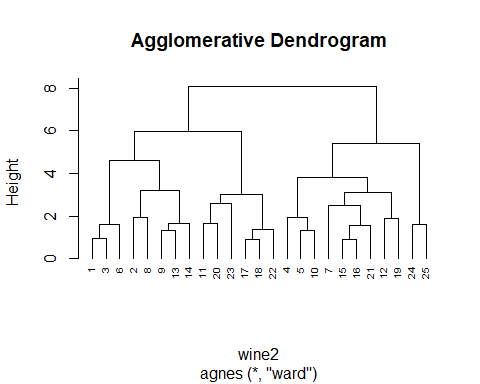
Task 9: Use agglomerative clustering to develop a dendogram for the scaled wine data. Follow the same process from the lecture where we used a custom function to identify the distance metric that maximized the agglomerative coefficient. Plot the dendogram.

m = c( "average", "single", "complete", "ward")  
names(m) = c( "average", "single", "complete", "ward")  
  
ac = function(x) {  
 agnes(wine2, method = x)$ac  
}  
map\_dbl(m, ac)

## average single complete ward   
## 0.5666719 0.2920143 0.7196616 0.8112139

Ward’s is highest.We will use Ward’s method to agglomeratively develop a heirachial clustering approach.

hc = agnes(wine2, method = "ward") #change ward to other method if desired  
pltree(hc, cex = 0.6, hang = -1, main = "Agglomerative Dendrogram")



Task 10: Repeat Task 9, but with divisive clustering.

hc2 = diana(wine2)  
pltree(hc2, cex = 0.6, hang = -1, main = "Divisive Dendogram")

