Clustering

Faith

9/8/2020

Installing all the necessary packages Creating a fuction to help load the data

```
# cleam unnecessary items

gc()
```

```
##          used (Mb) gc trigger (Mb) max used (Mb)
## Ncells 398395 21.3     818947 43.8   638648 34.2
## Vcells 725154  5.6    8388608 64.0  1632005 12.5
```

```
rm(list = ls(all = TRUE))

packages<-function(x){
  x<-as.character(match.call()[[2]])
  if (!require(x,character.only=TRUE)){
    install.packages(pkgs=x,repos="http://cran.r-project.org")
    require(x,character.only=TRUE)
  }
}

packages(corrplot)
```

```
## Loading required package: corrplot

## corrplot 0.84 loaded
```

```
packages(gridExtra)
```

```
## Loading required package: gridExtra
```

```
packages(GGally)
```

```
## Loading required package: GGally

## Loading required package: ggplot2

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

```
packages(cluster) # clustering algorithms
```

```
## Loading required package: cluster
```

```
packages(factoextra) # clustering algorithms & visualization
```

```
## Loading required package: factoextra
```

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

Load the data

```
setwd('C:/Users/FGakori/Documents/supervised and unsupervised')
wines <- read.csv('wine.csv')
```

```
head(wines)
```

```
##   Wine Alcohol Malic.acid  Ash  Acl  Mg Phenols Flavanoids Nonflavanoid.phenols
## 1    1   14.23       1.71 2.43 15.6 127    2.80       3.06                  0.28
## 2    1   13.20       1.78 2.14 11.2 100    2.65       2.76                  0.26
## 3    1   13.16       2.36 2.67 18.6 101    2.80       3.24                  0.30
## 4    1   14.37       1.95 2.50 16.8 113    3.85       3.49                  0.24
## 5    1   13.24       2.59 2.87 21.0 118    2.80       2.69                  0.39
## 6    1   14.20       1.76 2.45 15.2 112    3.27       3.39                  0.34
##   Proanth Color.int  Hue   OD Proline
## 1    2.29      5.64 1.04 3.92    1065
## 2    1.28      4.38 1.05 3.40    1050
## 3    2.81      5.68 1.03 3.17    1185
## 4    2.18      7.80 0.86 3.45    1480
## 5    1.82      4.32 1.04 2.93     735
## 6    1.97      6.75 1.05 2.85    1450
```

Removing the first column

Data Analysis

```
summary(wines)
```

```
##       Wine          Alcohol        Malic.acid         Ash
##  Min.   :1.000   Min.   :11.03   Min.   :0.740   Min.   :1.360
##  1st Qu.:1.000   1st Qu.:12.36   1st Qu.:1.603   1st Qu.:2.210
##  Median :2.000   Median :13.05   Median :1.865   Median :2.360
##  Mean   :1.938   Mean   :13.00   Mean   :2.336   Mean   :2.367
##  3rd Qu.:3.000   3rd Qu.:13.68   3rd Qu.:3.083   3rd Qu.:2.558
##  Max.   :3.000   Max.   :14.83   Max.   :5.800   Max.   :3.230
##       Acl              Mg           Phenols        Flavanoids
##  Min.   :10.60   Min.   : 70.00   Min.   :0.980   Min.   :0.340
```

```
##  1st Qu.:17.20   1st Qu.: 88.00   1st Qu.:1.742   1st Qu.:1.205
##  Median :19.50   Median : 98.00   Median :2.355   Median :2.135
##  Mean   :19.49   Mean   : 99.74   Mean   :2.295   Mean   :2.029
##  3rd Qu.:21.50   3rd Qu.:107.00   3rd Qu.:2.800   3rd Qu.:2.875
##  Max.   :30.00   Max.   :162.00   Max.   :3.880   Max.   :5.080
##  Nonflavanoid.phenols    Proanth        Color.int           Hue
##  Min.   :0.1300       Min.   :0.410   Min.   : 1.280   Min.   :0.4800
##  1st Qu.:0.2700       1st Qu.:1.250   1st Qu.: 3.220   1st Qu.:0.7825
##  Median :0.3400       Median :1.555   Median : 4.690   Median :0.9650
##  Mean   :0.3619       Mean   :1.591   Mean   : 5.058   Mean   :0.9574
##  3rd Qu.:0.4375       3rd Qu.:1.950   3rd Qu.: 6.200   3rd Qu.:1.1200
##  Max.   :0.6600       Max.   :3.580   Max.   :13.000   Max.   :1.7100
##        OD           Proline
##  Min.   :1.270   Min.   : 278.0
##  1st Qu.:1.938   1st Qu.: 500.5
##  Median :2.780   Median : 673.5
##  Mean   :2.612   Mean   : 746.9
##  3rd Qu.:3.170   3rd Qu.: 985.0
##  Max.   :4.000   Max.   :1680.0
```

```
str(wines)
```

```
## 'data.frame':    178 obs. of  14 variables:
##  $ Wine                : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Alcohol             : num  14.2 13.2 13.2 14.4 13.2 ...
##  $ Malic.acid          : num  1.71 1.78 2.36 1.95 2.59 1.76 1.87 2.15 1.64 1.35 ...
##  $ Ash                 : num  2.43 2.14 2.67 2.5 2.87 2.45 2.45 2.61 2.17 2.27 ...
##  $ Acl                 : num  15.6 11.2 18.6 16.8 21 15.2 14.6 17.6 14 16 ...
##  $ Mg                  : int  127 100 101 113 118 112 96 121 97 98 ...
##  $ Phenols             : num  2.8 2.65 2.8 3.85 2.8 3.27 2.5 2.6 2.8 2.98 ...
##  $ Flavanoids          : num  3.06 2.76 3.24 3.49 2.69 3.39 2.52 2.51 2.98 3.15 ...
##  $ Nonflavanoid.phenols: num  0.28 0.26 0.3 0.24 0.39 0.34 0.3 0.31 0.29 0.22 ...
##  $ Proanth             : num  2.29 1.28 2.81 2.18 1.82 1.97 1.98 1.25 1.98 1.85 ...
##  $ Color.int           : num  5.64 4.38 5.68 7.8 4.32 6.75 5.25 5.05 5.2 7.22 ...
##  $ Hue                 : num  1.04 1.05 1.03 0.86 1.04 1.05 1.02 1.06 1.08 1.01 ...
##  $ OD                  : num  3.92 3.4 3.17 3.45 2.93 2.85 3.58 3.58 2.85 3.55 ...
##  $ Proline             : int  1065 1050 1185 1480 735 1450 1290 1295 1045 1045 ...
```

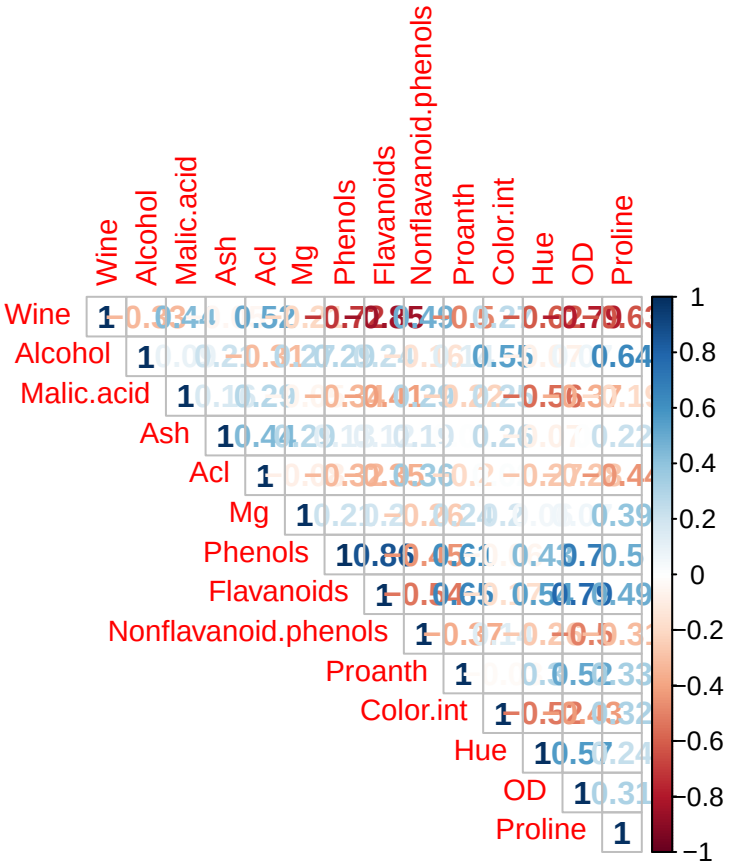
The variables are either numerical / integers

```
library(magrittr)
```

plot histogram for each attribute

Correlation matrix

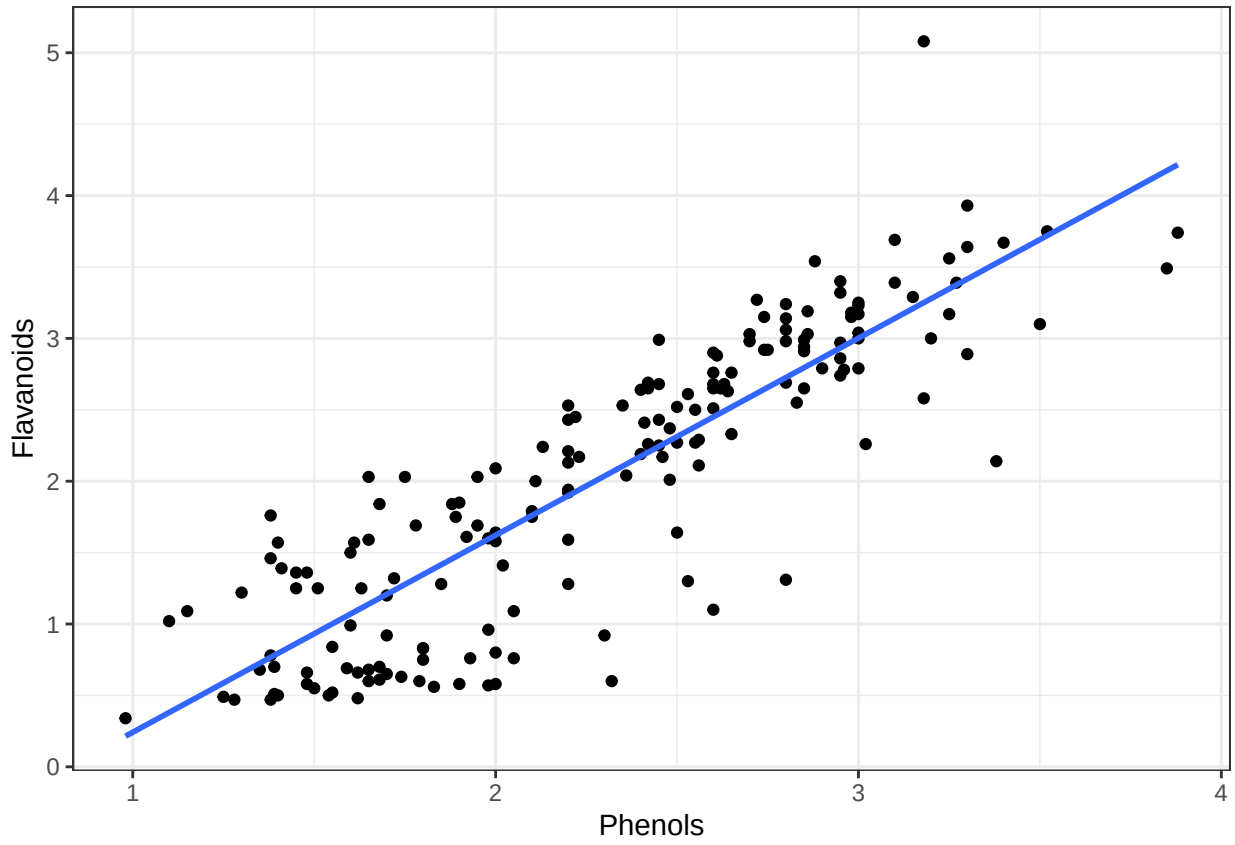
```
corrplot(cor(wines), type = 'upper', method = 'number', tl.cex = 0.9)
```



There is a strong correlation between total_phenols and flavanoids. We can model the relationship between these two variables by fitting a linear equation.

```
#  relationship btwn phenols and flavanoids

ggplot(wines, aes(x = Phenols, y = Flavanoids)) +
  geom_point() +
  geom_smooth(method = 'lm', se = FALSE) +
  theme_bw()
```

```
## `geom_smooth()` using formula 'y ~ x'
```



preparing data for k-means

normalize the data

```
winesNorm <- as.data.frame(scale(wines))
summary(winesNorm)
```

```
##       Wine              Alcohol          Malic.acid          Ash
##  Min.   :-1.21053   Min.   :-2.42739   Min.   :-1.4290   Min.   :-3.66881
##  1st Qu.:-1.21053   1st Qu.:-0.78603   1st Qu.:-0.6569   1st Qu.:-0.57051
##  Median : 0.07974   Median : 0.06083   Median :-0.4219   Median :-0.02375
##  Mean   : 0.00000   Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.: 1.37000   3rd Qu.: 0.83378   3rd Qu.: 0.6679   3rd Qu.: 0.69615
##  Max.   : 1.37000   Max.   : 2.25341   Max.   : 3.1004   Max.   : 3.14745
##       Acl                Mg             Phenols          Flavanoids
##  Min.   :-2.663505   Min.   :-2.0824   Min.   :-2.10132   Min.   :-1.6912
##  1st Qu.:-0.687199   1st Qu.:-0.8221   1st Qu.:-0.88298   1st Qu.:-0.8252
##  Median : 0.001514   Median :-0.1219   Median : 0.09569   Median : 0.1059
##  Mean   : 0.000000   Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.600395   3rd Qu.: 0.5082   3rd Qu.: 0.80672   3rd Qu.: 0.8467
##  Max.   : 3.145637   Max.   : 4.3591   Max.   : 2.53237   Max.   : 3.0542
##  Nonflavanoid.phenols    Proanth          Color.int           Hue
##  Min.   :-1.8630      Min.   :-2.06321   Min.   :-1.6297   Min.   :-2.08884
##  1st Qu.:-0.7381      1st Qu.:-0.59560   1st Qu.:-0.7929   1st Qu.:-0.76540
```

```
##  Median :-0.1756      Median :-0.06272   Median :-0.1588   Median : 0.03303
##  Mean   : 0.0000      Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.: 0.6078      3rd Qu.: 0.62741   3rd Qu.: 0.4926   3rd Qu.: 0.71116
##  Max.   : 2.3956      Max.   : 3.47527   Max.   : 3.4258   Max.   : 3.29241
##        OD             Proline
##  Min.   :-1.8897   Min.   :-1.4890
##  1st Qu.:-0.9496   1st Qu.:-0.7824
##  Median : 0.2371   Median :-0.2331
##  Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.7864   3rd Qu.: 0.7561
##  Max.   : 1.9554   Max.   : 2.9631
```

Computing k-means clustering in R

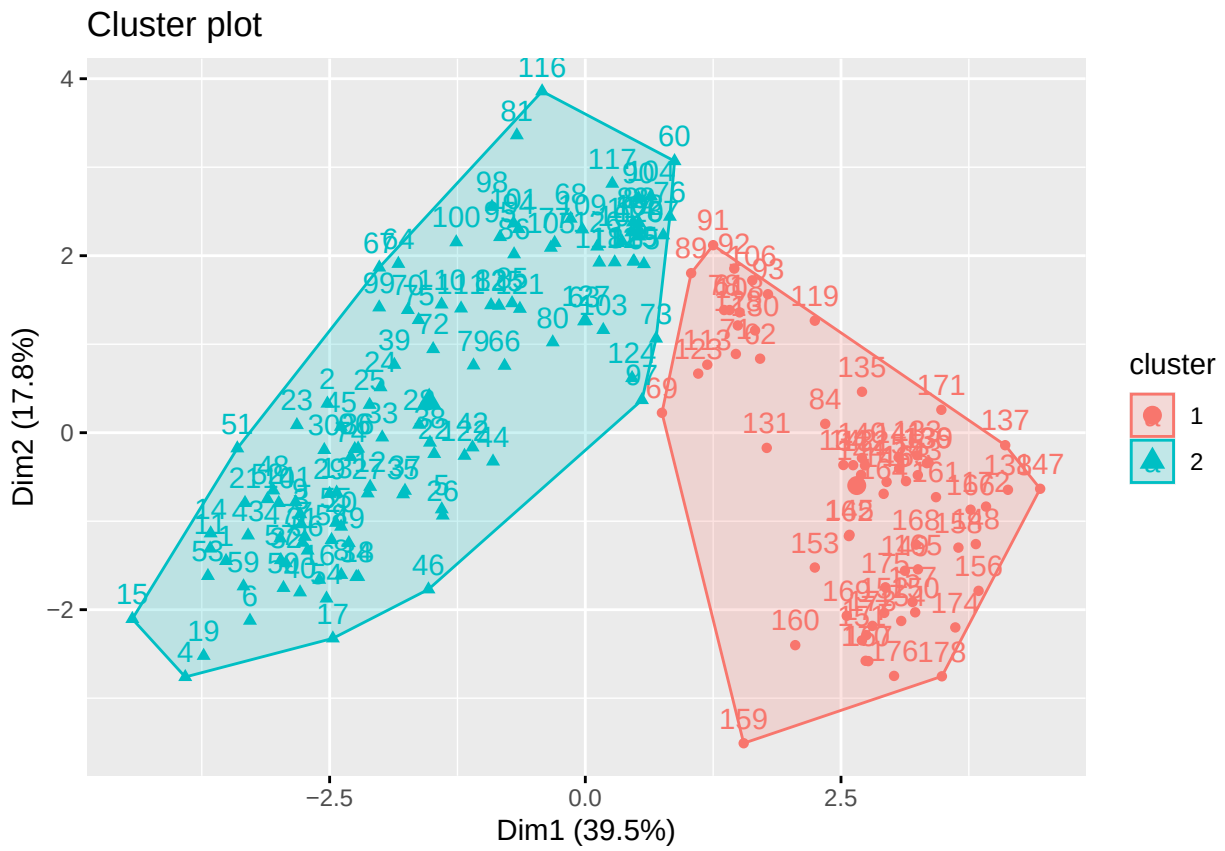
We can compute k-means in R with the kmeans function. Here will group the data into two clusters (centers = 2). The kmeans function also has an nstart option that attempts multiple initial configurations and reports on the best one. For example, adding nstart = 25 will generate 25 initial configurations and reports on the best one. For example, adding nstart=25 generates 25 initial configurations. This approach is often recommended.

```
set.seed(123)
wines_k2 <- kmeans(winesNorm, centers = 2, nstart = 25)
print(wines_k2)
```

```
## K-means clustering with 2 clusters of sizes 65, 113
##
## Cluster means:
##         Wine     Alcohol Malic.acid        Ash        Acl          Mg
## 1  1.0325460 -0.07277357  0.6626412  0.1893414  0.5151693 -0.15425269
## 2 -0.5939424  0.04186090 -0.3811653 -0.1089132 -0.2963363  0.08872942
##      Phenols Flavanoids Nonflavanoid.phenols    Proanth  Color.int        Hue
## 1 -0.9410522 -1.0436916            0.8355920 -0.7141412  0.5419399 -0.8795908
## 2  0.5413132  0.6003536           -0.4806503  0.4107892 -0.3117353  0.5059593
##           OD    Proline
## 1 -1.0663102 -0.4519062
## 2  0.6133643  0.2599460
##
## Clustering vector:
##   [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [38] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 1 2 1 2 2 2
##  [75] 2 2 2 1 2 2 2 2 2 1 2 2 2 2 1 2 1 1 1 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2
## [112] 2 1 2 2 2 2 2 2 1 2 2 2 2 1 2 2 2 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [149] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##
## Within cluster sum of squares by cluster:
## [1]  532.4956 1185.0295
##  (between_SS / total_SS =  30.7 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

visualize cluster created

```
fviz_cluster(wines_k2, data = winesNorm)
```



When we print the model we build (wines_k2), it shows information like, number of clusters, centers of the clusters, size of the clusters and sum of square. Let's check how to get these attributes of our model.

```
# cluster to whih each point is associated
wines_k2$cluster
```

```
##   [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [38] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 1 2 1 2 2 2
##  [75] 2 2 2 1 2 2 2 2 2 1 2 2 2 2 1 2 1 1 1 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2
## [112] 2 1 2 2 2 2 2 1 2 2 2 1 2 2 2 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [149] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
# cluster centers(means)

wines_k2$centers
```

```
##         Wine     Alcohol Malic.acid        Ash        Acl          Mg
## 1  1.0325460 -0.07277357  0.6626412  0.1893414  0.5151693 -0.15425269
## 2 -0.5939424  0.04186090 -0.3811653 -0.1089132 -0.2963363  0.08872942
##      Phenols Flavanoids Nonflavanoid.phenols    Proanth  Color.int        Hue
## 1 -0.9410522 -1.0436916            0.8355920 -0.7141412  0.5419399 -0.8795908
## 2  0.5413132  0.6003536           -0.4806503  0.4107892 -0.3117353  0.5059593
##           OD    Proline
```

```
## 1 -1.0663102 -0.4519062
## 2  0.6133643  0.2599460
```

```
# cluster size
wines_k2$size
```

```
## [1]  65 113
```

```
# between clusters sum of squares
wines_k2$betweenss
```

```
## [1] 760.4749
```

```
# total sum of squares
wines_k2$tot.withinss
```

```
## [1] 1717.525
```

```
# total sum of squares
wines_k2$totss
```

```
## [1] 2478
```

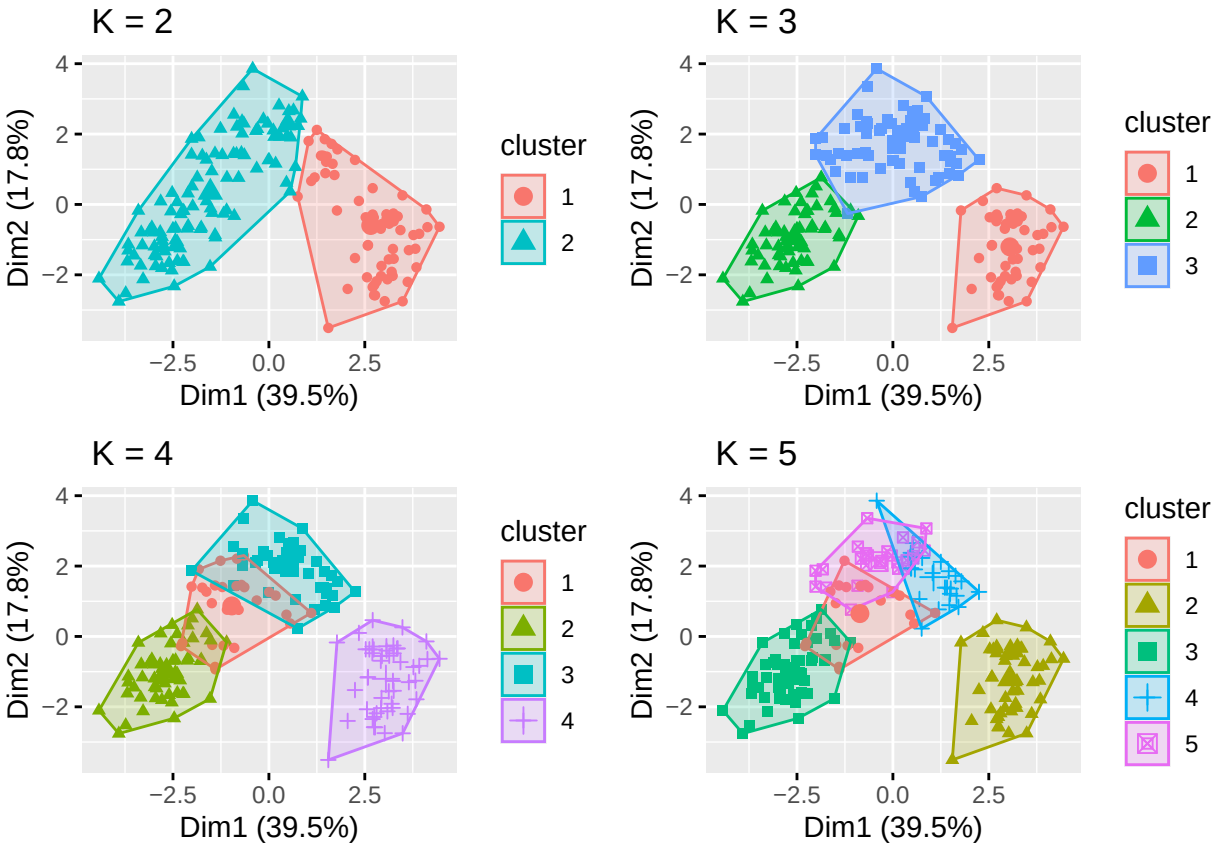
the number of clusters (k) must be set before we start the algorithm, it is often advantageous to use several different values of k and examine the differences in the results.

We can execute the same process for 3, 4, and 5 clusters, and the results are shown in the figure:

```
wines_K3 <- kmeans(winesNorm, centers = 3, nstart = 25)
wines_K4 <- kmeans(winesNorm, centers = 4, nstart = 25)
wines_K5 <- kmeans(winesNorm, centers = 5, nstart = 25)
```

plot the clusters to compare different k values

```
p1 <- fviz_cluster(wines_k2, geom = "point", data = winesNorm) + ggtitle(" K = 2")
p2 <- fviz_cluster(wines_K3, geom = "point", data = winesNorm) + ggtitle(" K = 3")
p3 <- fviz_cluster(wines_K4, geom = "point", data = winesNorm) + ggtitle(" K = 4")
p4 <- fviz_cluster(wines_K5, geom = "point", data = winesNorm) + ggtitle(" K = 5")

grid.arrange(p1, p2, p3, p4, nrow = 2)
```



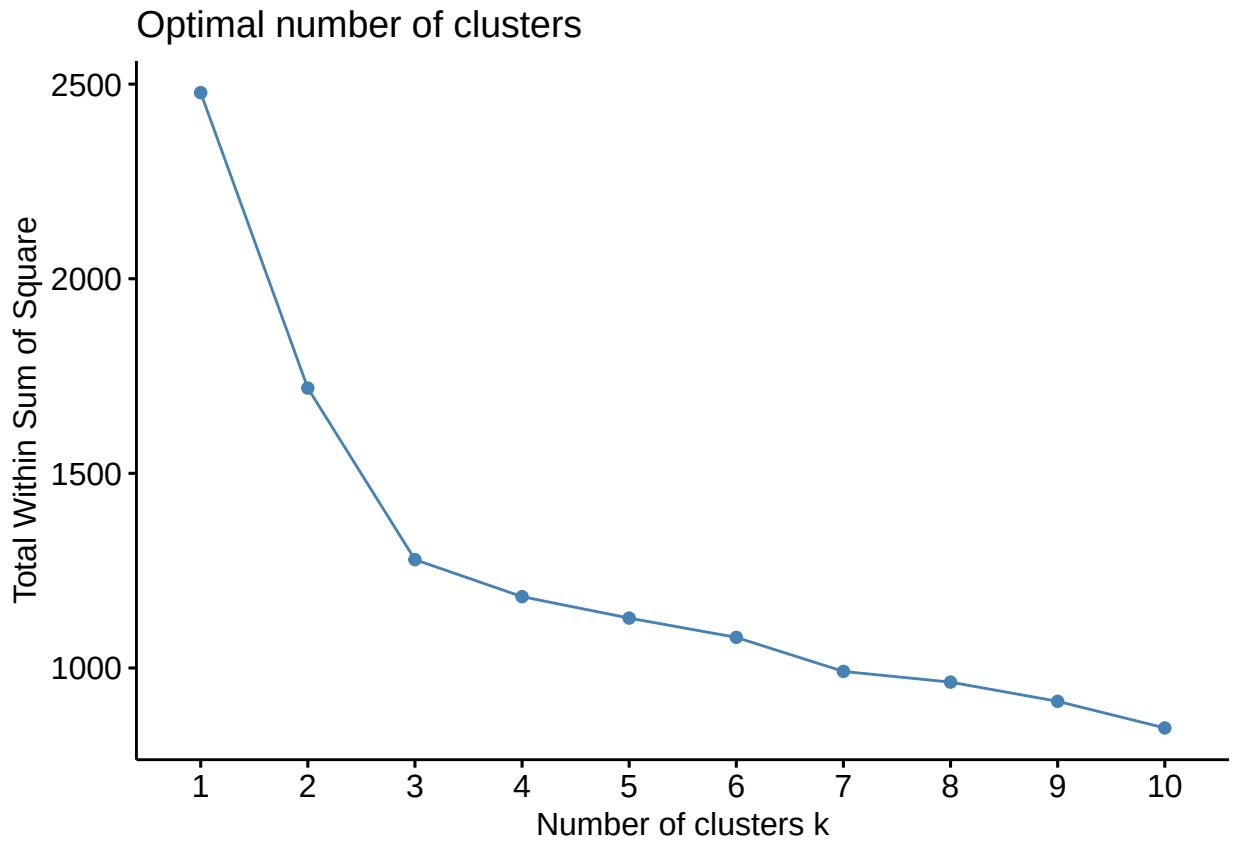
Determine optimal clusters

K-means clustering requires that you specify in advance the number of clusters to extract. A plot of the total within-groups sums of squares against the number of clusters in a k-means solution can be helpful. A bend in the graph can suggest the appropriate number of clusters.

Below are the methods to determine the optimal number of clusters

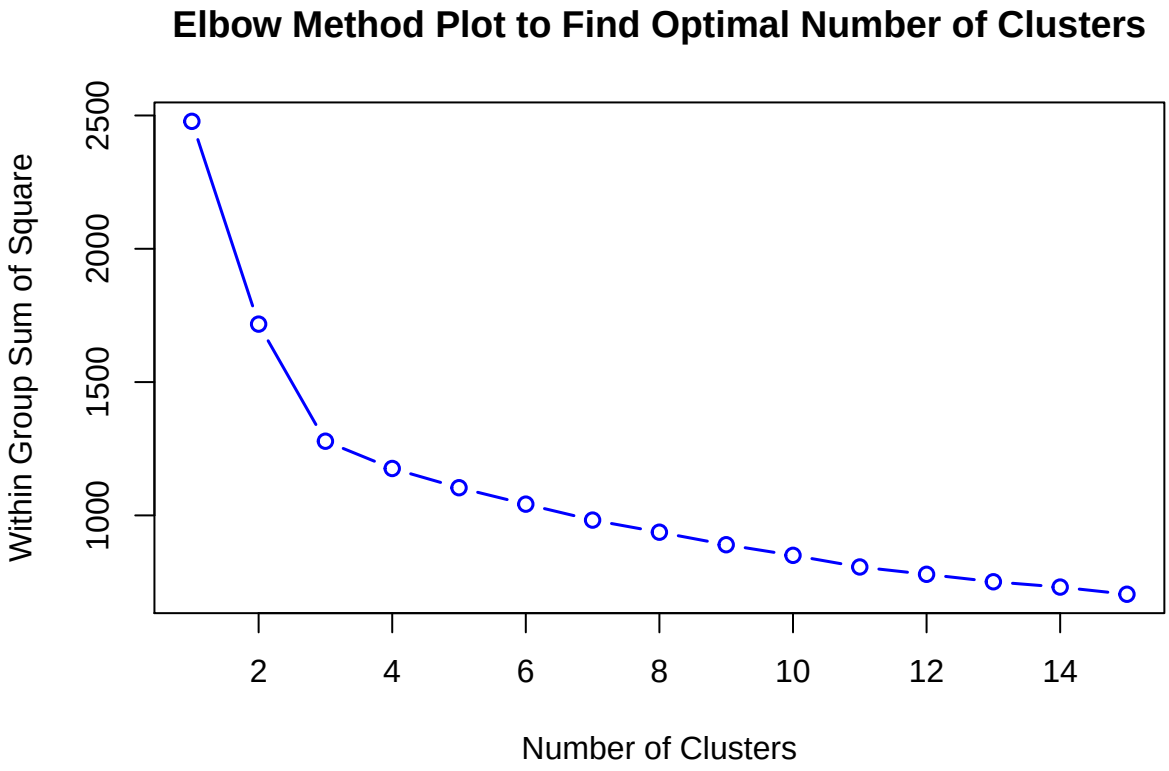
Elbow method Silhouette method Gap statistic

```
# method 1
# determine optimal clusters(k) using elbow method

fviz_nbclust(x = winesNorm,FUNcluster = kmeans, method = 'wss' )
```



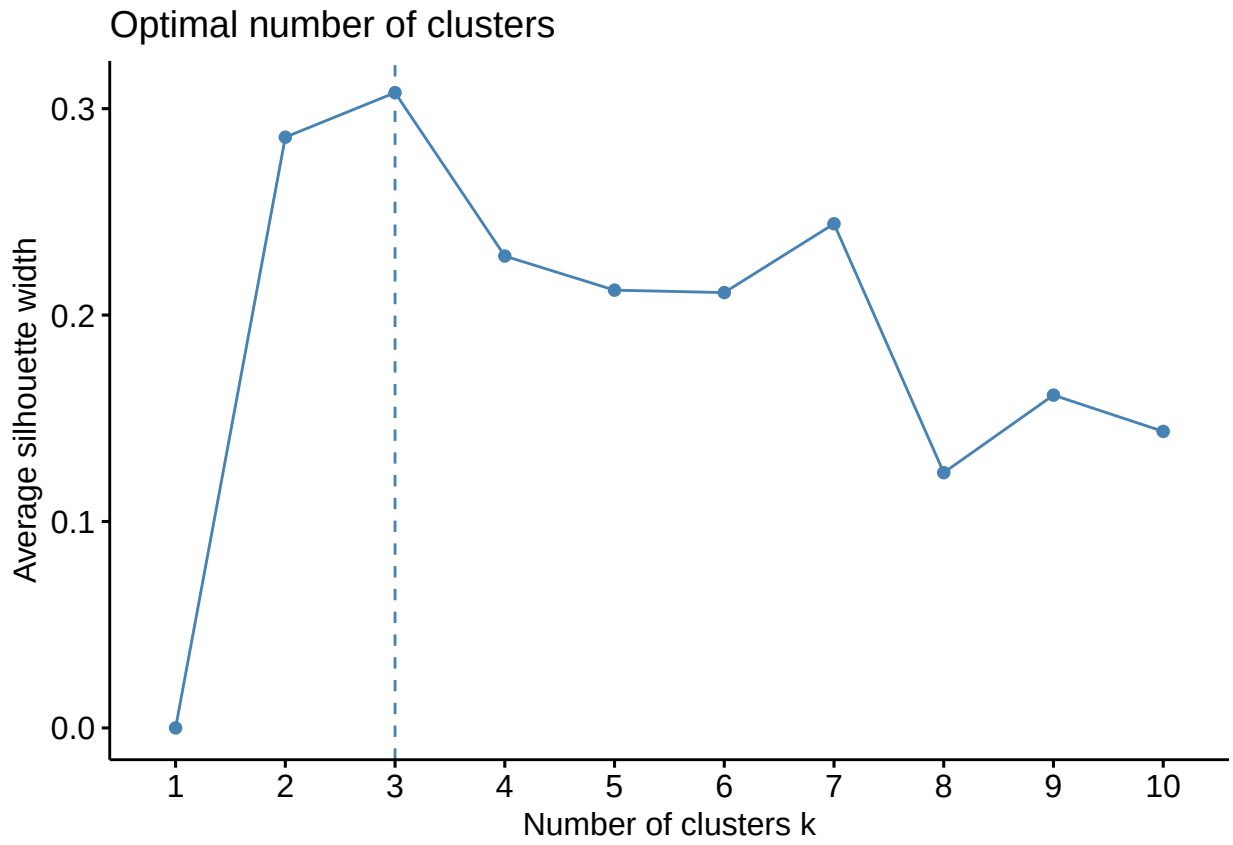
creating a function

```
wssplot <- function(data, nc = 15, set.seed = 1234){
  wss <- (nrow(data) - 1)*sum(apply(data, 2, var))
  for(i in 2:nc) {
    set.seed(1234)
    wss[i] <- sum(kmeans(x = data, centers = i, nstart = 25)$withinss)
  }
  plot(1:nc, wss, type = 'b', xlab = 'Number of Clusters', ylab = 'Within Group Sum of Square',
       main = 'Elbow Method Plot to Find Optimal Number of Clusters', frame.plot = T,
       col = 'blue', lwd = 1.5)
}

wssplot(winesNorm)
```



Determining Optimal clusters (k) Using Average Silhouette Method

```
fviz_nbclust(x = winesNorm,FUNcluster = kmeans, method = 'silhouette' )
```



There is another method called Gap-Static used for finding the optimal value of K.

```
# compute gap statistic
set.seed(123)
gap_stat <- clusGap(x = winesNorm, FUN = kmeans, K.max = 15, nstart = 25, B = 50 )
```

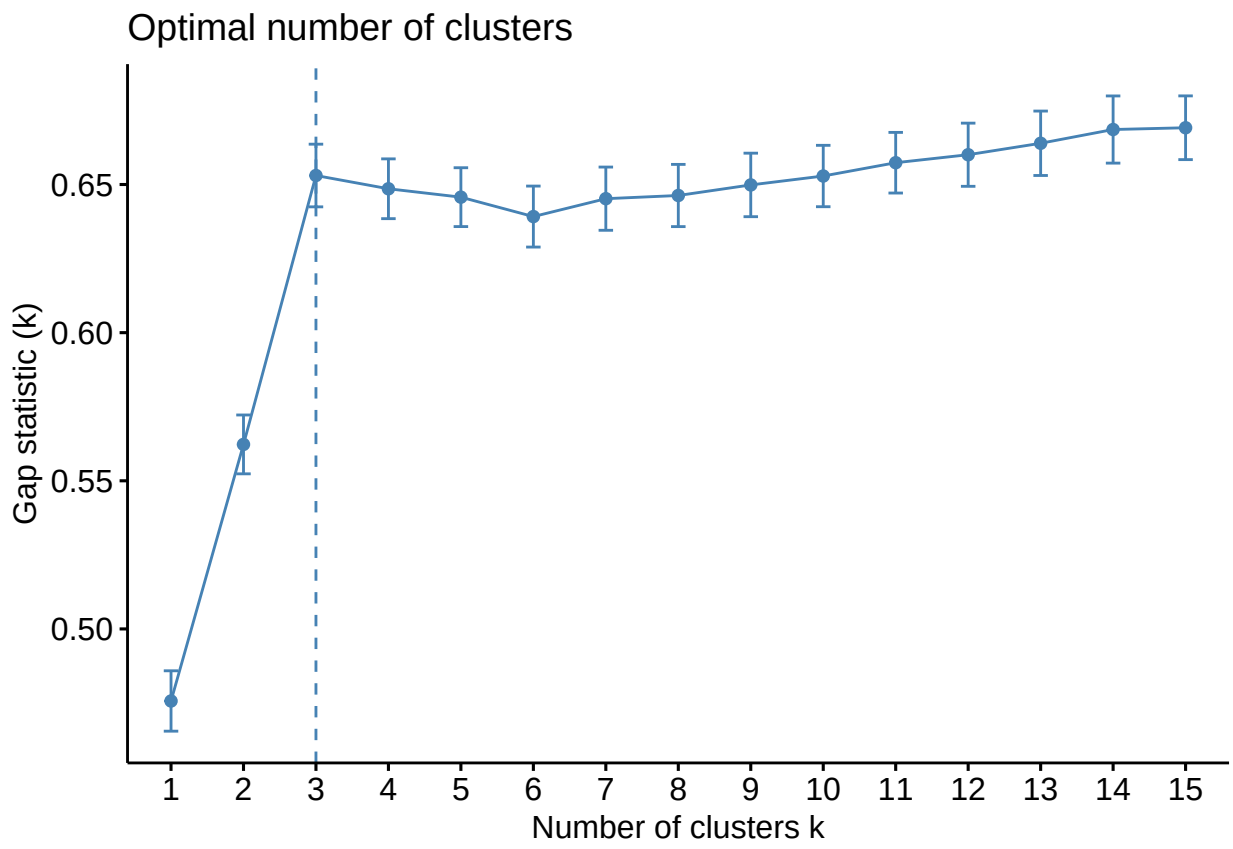
```
## Warning: did not converge in 10 iterations
```

```
# Print the result
print(gap_stat, method = "firstmax")
```

```
## Clustering Gap statistic ["clusGap"] from call:
## clusGap(x = winesNorm, FUNcluster = kmeans, K.max = 15, B = 50,     nstart = 25)
## B=50 simulated reference sets, k = 1..15; spaceH0="scaledPCA"
##  --> Number of clusters (method 'firstmax'): 3
##           logW   E.logW       gap      SE.sim
##  [1,] 5.412665 5.888350 0.4756845 0.010188148
##  [2,] 5.222775 5.785055 0.5622803 0.009933207
##  [3,] 5.068611 5.721651 0.6530396 0.010576078
##  [4,] 5.026552 5.675115 0.6485629 0.010091246
##  [5,] 4.993110 5.638843 0.6457332 0.009935349
##  [6,] 4.968524 5.607703 0.6391795 0.010293469
##  [7,] 4.934266 5.579486 0.6452193 0.010670975
##  [8,] 4.907208 5.553503 0.6462943 0.010499218
##  [9,] 4.879826 5.529685 0.6498588 0.010717119
## [10,] 4.855766 5.508632 0.6528662 0.010363584
```

```
## [11,] 4.831005 5.488366 0.6573604 0.010232228
## [12,] 4.809628 5.469682 0.6600544 0.010663561
## [13,] 4.787576 5.451497 0.6639205 0.010870990
## [14,] 4.765697 5.434255 0.6685587 0.011333163
## [15,] 4.748267 5.417427 0.6691592 0.010754198
```

```r
# plot the result to determine the optimal number of clusters.
fviz_gap_stat(gap_stat)
```



Final analysis using three clusters

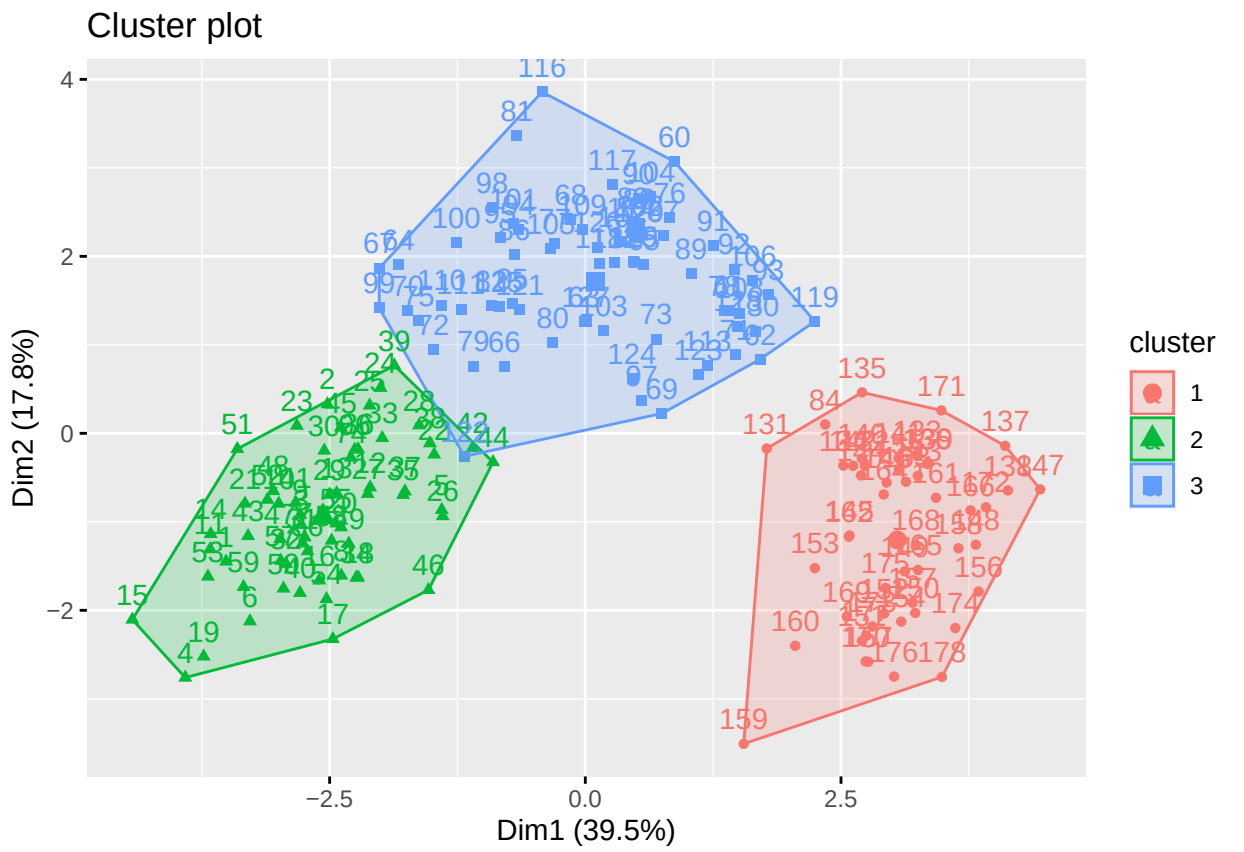
```r
# compute k-means clustering with k=3
set.seed(123)
final <- kmeans(winesNorm, centers = 3, nstart = 25)
print(final)
```

```
## K-means clustering with 3 clusters of sizes 49, 61, 68
##
## Cluster means:
##          Wine    Alcohol Malic.acid        Ash        Acl          Mg
## 1  1.34366784  0.1860184  0.9024258  0.2485092  0.5820616 -0.05049296
## 2 -1.16822514  0.8756272 -0.3037196  0.3180446 -0.6626544  0.56329925
## 3  0.07973544 -0.9195318 -0.3778231 -0.4643776  0.1750133 -0.46892793
##       Phenols  Flavanoids Nonflavanoid.phenols     Proanth  Color.int
## 1 -0.98577624 -1.23271740          0.714825281 -0.74749896  0.9857177
## 2  0.87403990  0.94098462         -0.583942581  0.58014642  0.1667181
```

```
## 3 -0.07372644  0.04416309          0.008736157  0.01821349 -0.8598525
##          Hue          OD    Proline
## 1 -1.1879477 -1.2978785 -0.3789756
## 2  0.4823674  0.7648958  1.1550888
## 3  0.4233092  0.2490794 -0.7630972
##
## Clustering vector:
##   [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [38] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 2
##  [75] 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [112] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [149] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##
## Within cluster sum of squares by cluster:
## [1] 304.6223 350.5475 623.1702
##  (between_SS / total_SS =  48.4 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

visualize the results

```
fviz_cluster(final, data = winesNorm)
```



We can extract the clusters and add to our initial data to do some descriptive statistics at the cluster level