Cluster Analysis

Anisha Mittal

Objective

Perform a cluster analysis of the audio features data using k-means:

1. Consider a range of values of K from 1 to 10. Select the appropriate number of clusters for this data.
2. Compare the clustering obtained to the classification into genres, providing a brief interpretation of the clusters.

ETL: Extract Transfor Load

First we load the data. Then see the first few observations and structure of the data set.

Head of the data

```
##   genre         song_name                artist song_popularity
## 2  rock        In The End           Linkin Park              66
## 3  rock Seven Nation Army     The White Stripes              76
## 4  rock        By The Way Red Hot Chili Peppers              74
## 5  rock How You Remind Me            Nickelback              56
## 6  rock  Bring Me To Life           Evanescence              80
## 7  rock       Last Resort            Papa Roach              81
##   song_duration_ms acousticness danceability energy liveness loudness
## 2           216933     0.010300        0.542  0.853    0.108   -6.407
## 3           231733     0.008170        0.737  0.463    0.255   -7.828
## 4           216933     0.026400        0.451  0.970    0.102   -4.938
## 5           223826     0.000954        0.447  0.766    0.113   -5.065
## 6           235893     0.008950        0.316  0.945    0.396   -3.169
## 7           199893     0.000504        0.581  0.887    0.268   -3.659
##   speechiness   tempo audio_valence
## 2      0.0498 105.256         0.370
## 3      0.0792 123.881         0.324
## 4      0.1070 122.444         0.198
## 5      0.0313 172.011         0.574
## 6      0.1240 189.931         0.320
## 7      0.0624  90.578         0.724
```

Structure of the data before cleaning

```
## 'data.frame':    239 obs. of  13 variables:
##  $ genre           : Factor w/ 3 levels "rock","pop","acoustic": 1 1 1 1 1 1 1 1 1 1 ...
##  $ song_name       : Factor w/ 237 levels "Acoustic","Adore - Piano Unplugged",..: 99 171 31 89 29 10
```

```
##  $ artist          : Factor w/ 178 levels "*NSYNC","3 Doors Down",..: 111 169 141 123 55 133 81 164 1
##  $ song_popularity : int  66 76 74 56 80 81 76 80 81 78 ...
##  $ song_duration_ms: int  216933 231733 216933 223826 235893 199893 213800 222586 203346 168253 ...
##  $ acousticness    : num  0.0103 0.00817 0.0264 0.000954 0.00895 0.000504 0.00148 0.00108 0.00172 0.0
##  $ danceability    : num  0.542 0.737 0.451 0.447 0.316 0.581 0.613 0.33 0.542 0.629 ...
##  $ energy          : num  0.853 0.463 0.97 0.766 0.945 0.887 0.953 0.936 0.905 0.897 ...
##  $ liveness        : num  0.108 0.255 0.102 0.113 0.396 0.268 0.152 0.0926 0.136 0.263 ...
##  $ loudness        : num  -6.41 -7.83 -4.94 -5.07 -3.17 ...
##  $ speechiness     : num  0.0498 0.0792 0.107 0.0313 0.124 0.0624 0.0855 0.0917 0.054 0.0483 ...
##  $ tempo           : num  105 124 122 172 190 ...
##  $ audio_valence   : num  0.37 0.324 0.198 0.574 0.32 0.724 0.537 0.234 0.374 0.93 ...
```

We can see the data set has 239 observations and 13 variables. Out of the 13 variables, first 3 variables namely genre, song_name, and artist (columns) are categorical vectors. Since we do not require these columns we will drop them. Since, we have to analyze the audio features of the songs we will drop the song_popularity and song_duration_ms.

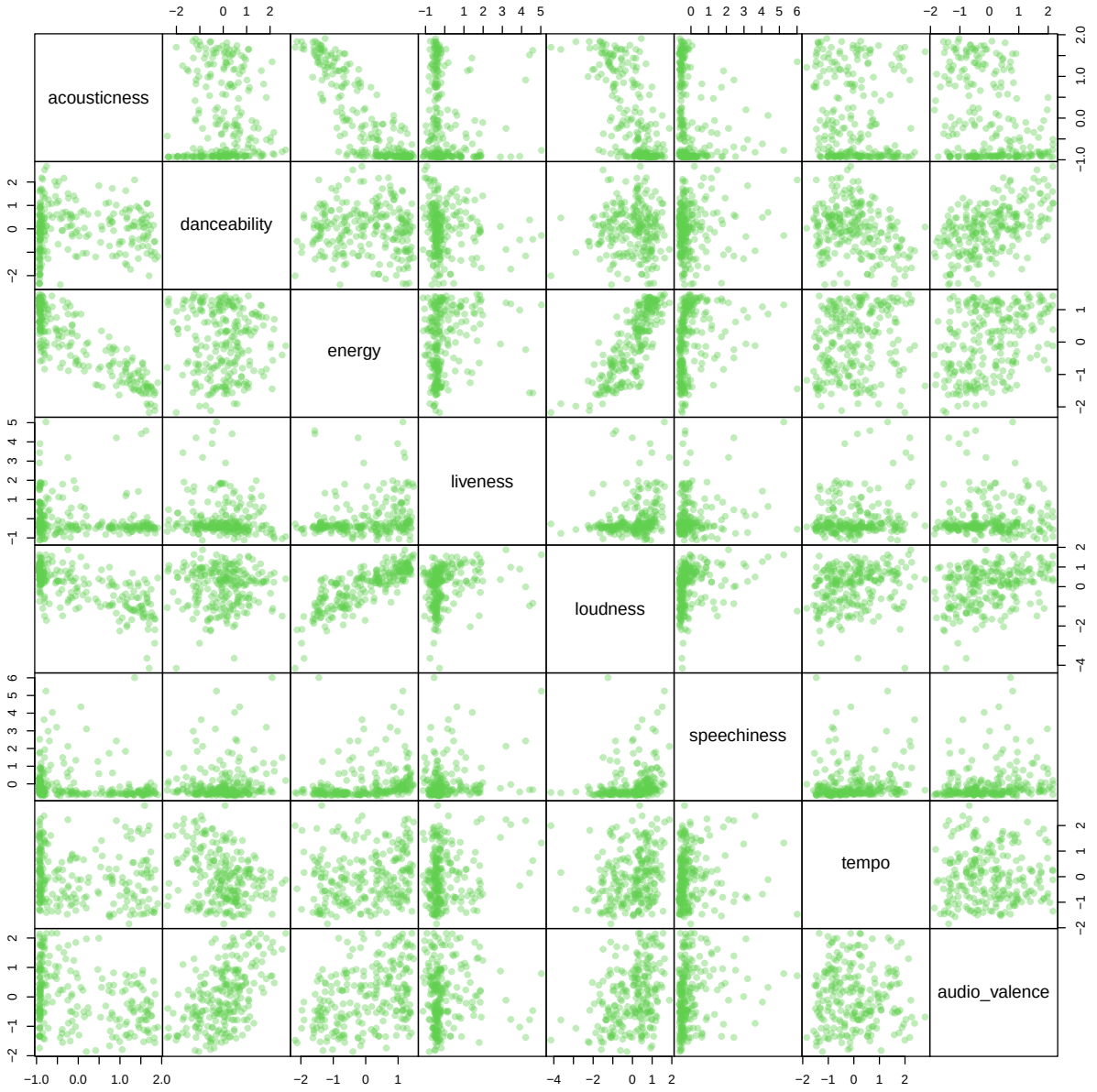
We can also see that the other numerical columns are not leveled. For example, The column loudness has negative values whereas the column tempo has values in hundreds. This means our data is varied and we might get improper results. In order to solve this problem we will SCALE our data using base R scale function.

Structure of the data after cleaning

```
##  num [1:239, 1:8] -0.9 -0.906 -0.853 -0.927 -0.904 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:239] "2" "3" "4" "5" ...
##   ..$ : chr [1:8] "acousticness" "danceability" "energy" "liveness" ...
##  - attr(*, "scaled:center")= Named num [1:8] 0.32 0.567 0.612 0.165 -6.699 ...
##   ..- attr(*, "names")= chr [1:8] "acousticness" "danceability" "energy" "liveness" ...
##  - attr(*, "scaled:scale")= Named num [1:8] 0.344 0.15 0.254 0.121 2.851 ...
##   ..- attr(*, "names")= chr [1:8] "acousticness" "danceability" "energy" "liveness" ...
```

Visualisation

Now, let us visualize the data in order to get an idea of the clusters.

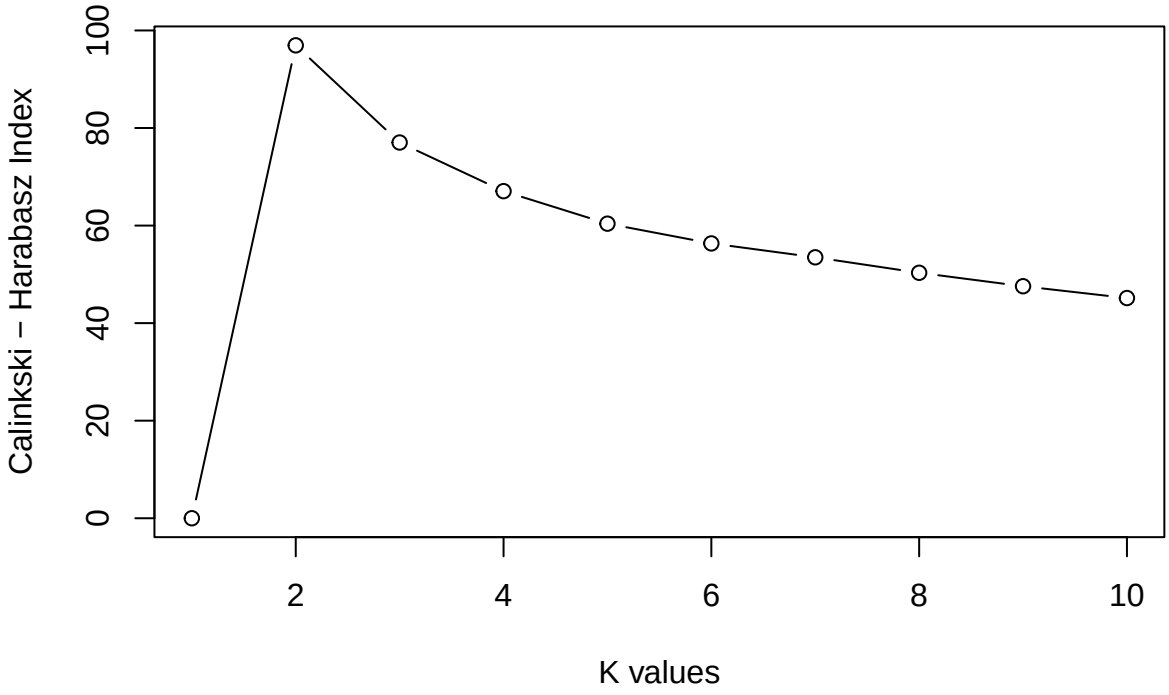


We can see the data might be divided in two or three clusters. To find out the exact number of clusters we will use Calinkski - Harabasz index.

Calinkski - Harabasz index (CH index)

We select the value of k which prompts the highest value of CH index.

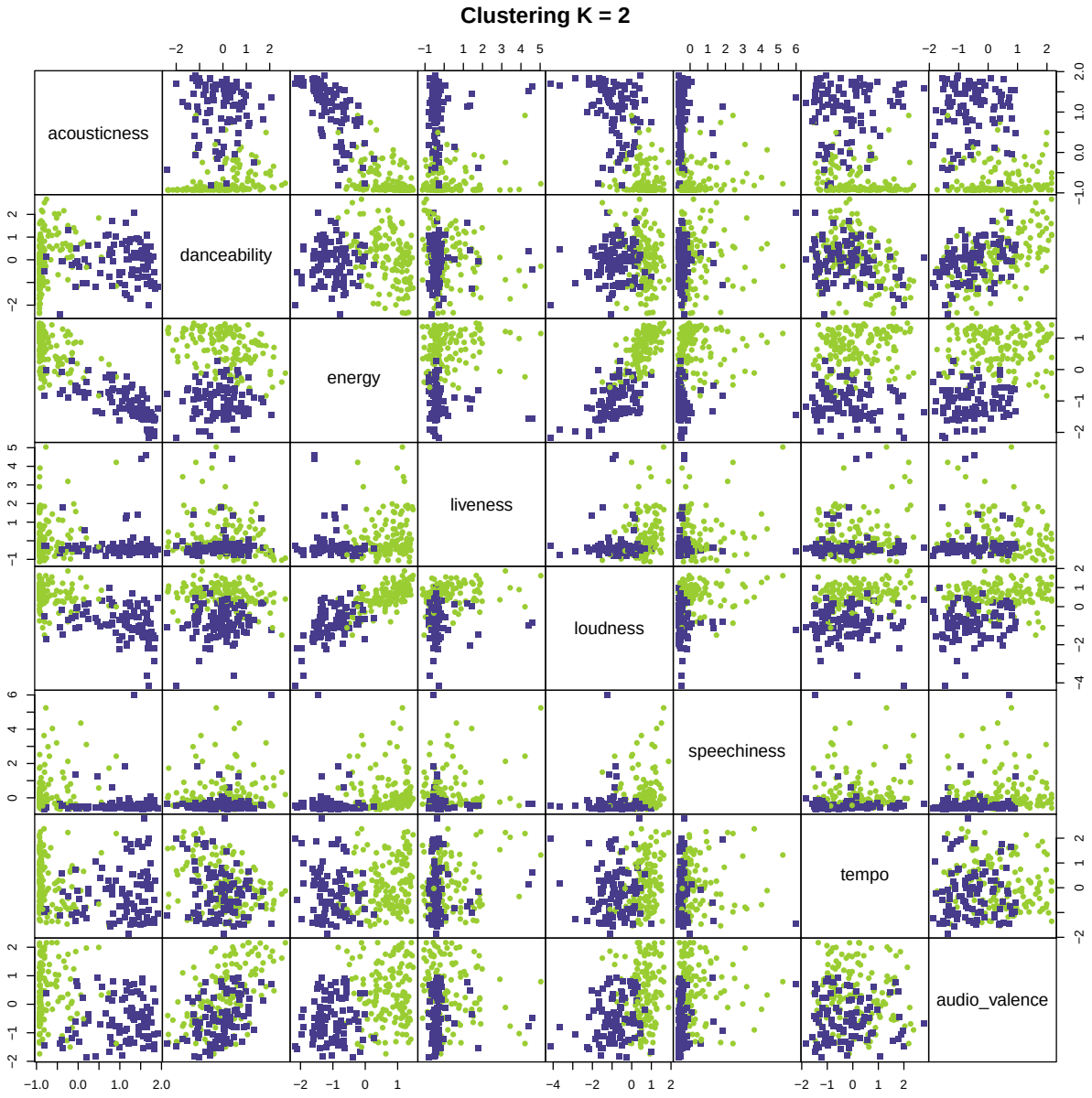
Note: We will set the CH index for k=1 as zero since CH index will by default put all data in one cluster.

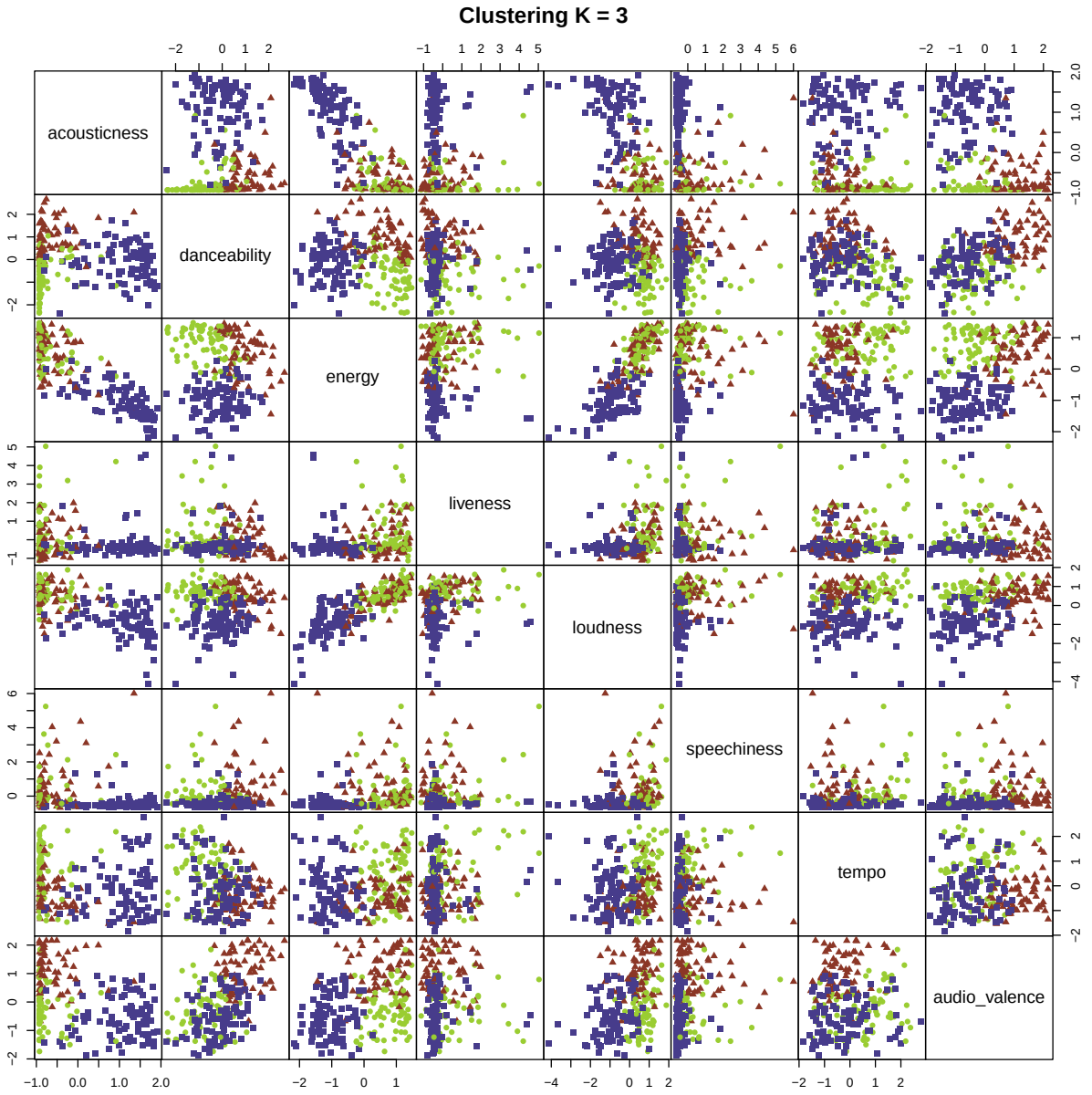


In the above plot, we can see for k=2 the CH index has the highest value. So, we will fit two clusters to our data. We can also check for three clusters.

Plots of the clusters

Pairs plot of the whole data divided into clusters.

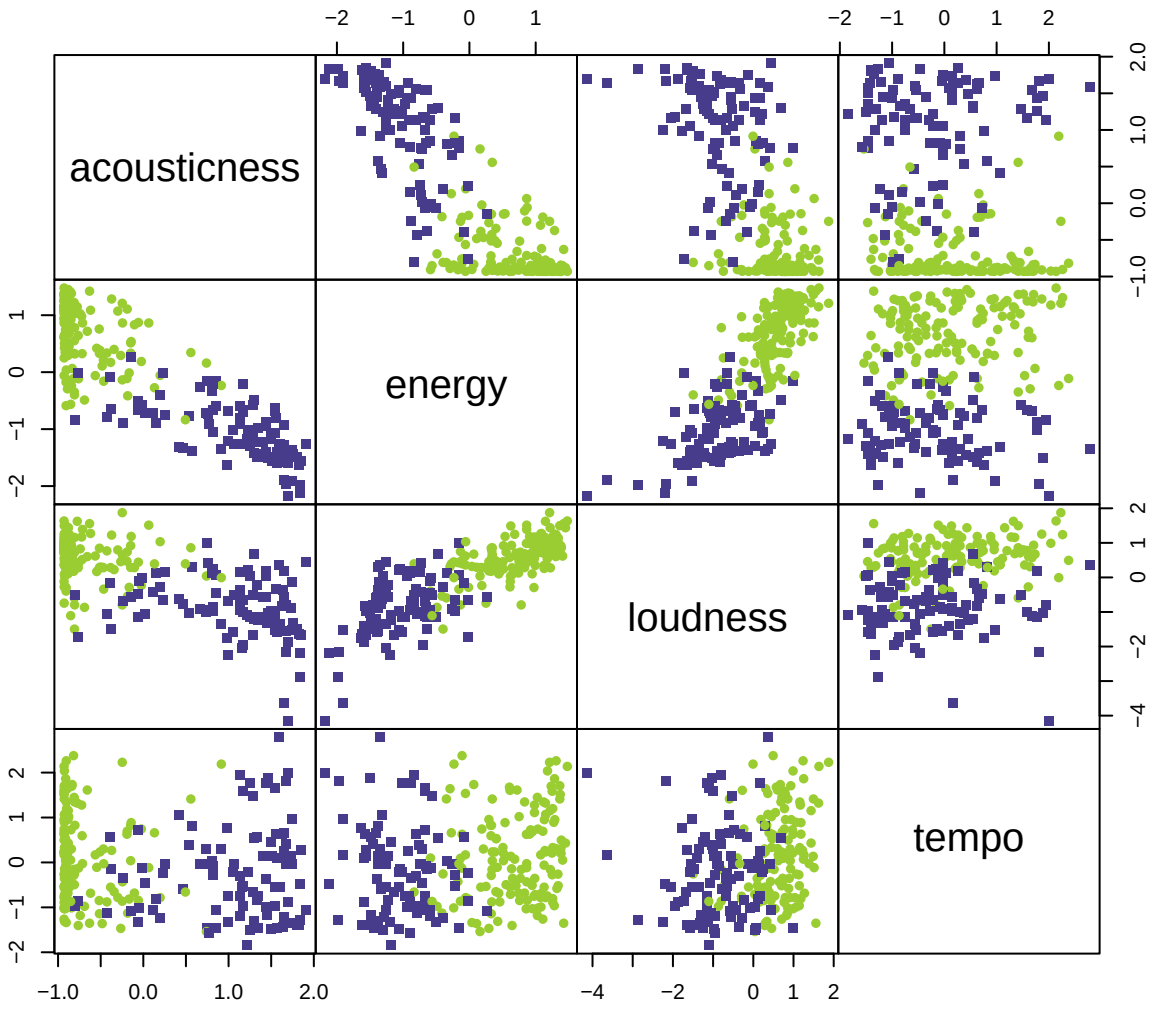


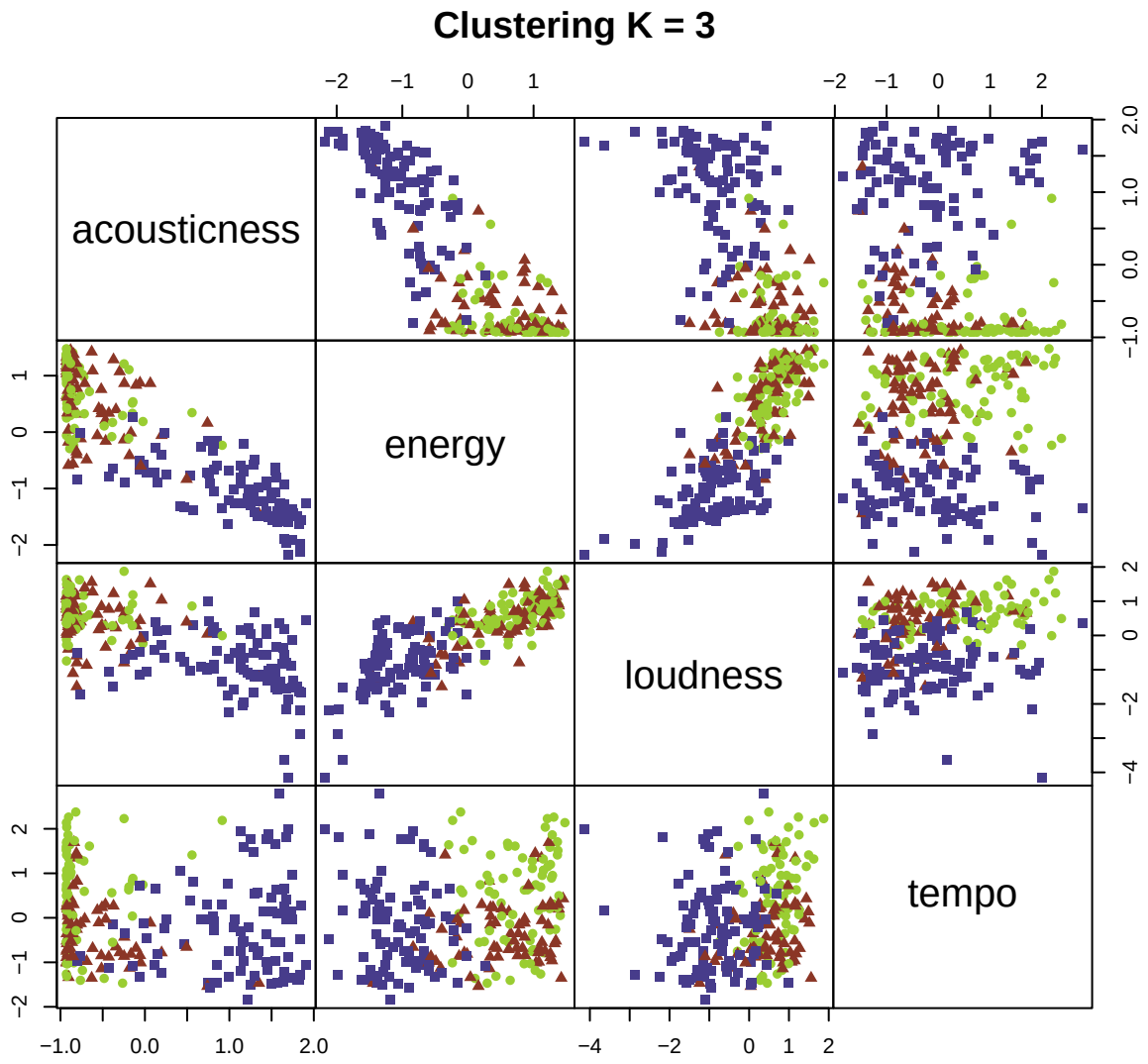


We can see from the pairs plot that k=2 divides the data well.

Pairs plot of Acousticness, tempo, loundness, and energy into clusters.

The data was originally divided into three genre; pop, rock and acoustics. Normally, pop and rock songs have almost similar tempo, energy, and loudness so if we find them in one cluster, it wouldn't be surprising. Let us take a look at the pairs plot of Acoustics, tempo, loudness, and energy into two and three clusters respectively.

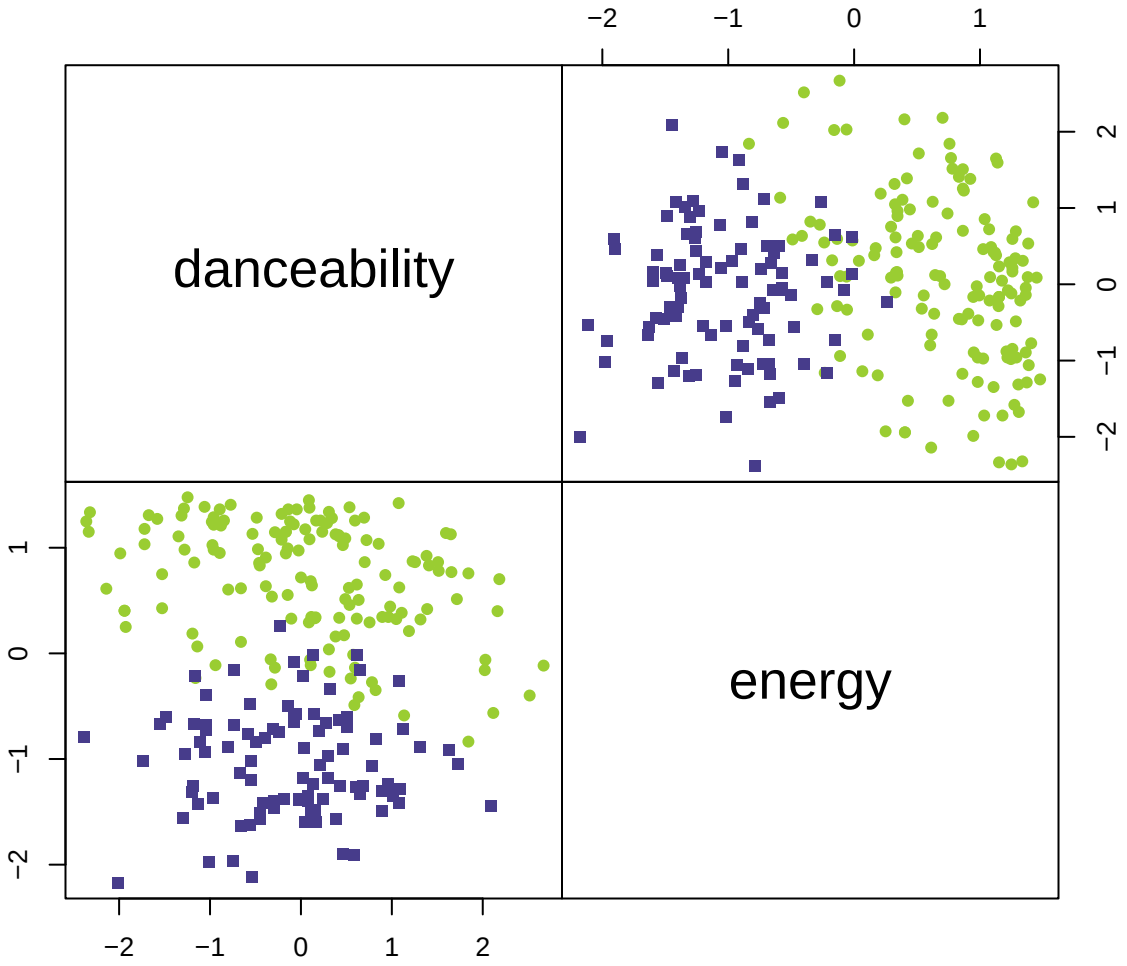

Clustering K = 2
acousticness
energy
loudness
tempo

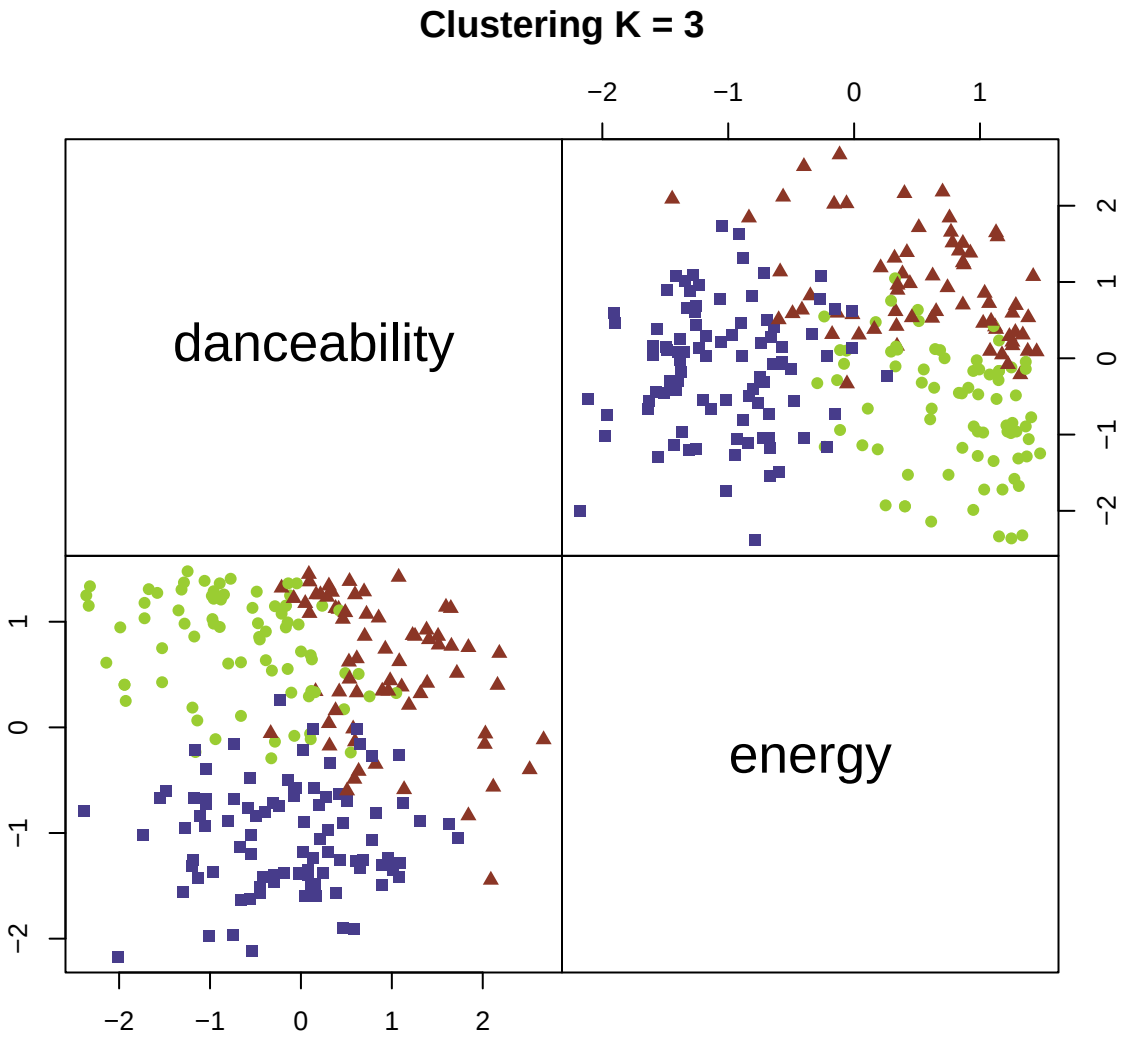


We can see the pairs plot for k=2 divides the data well.

Pairs plot of Danceability and energy into clusters.

Let us take a look at the pairs plot of danceability and energy into two and three clusters respectively.

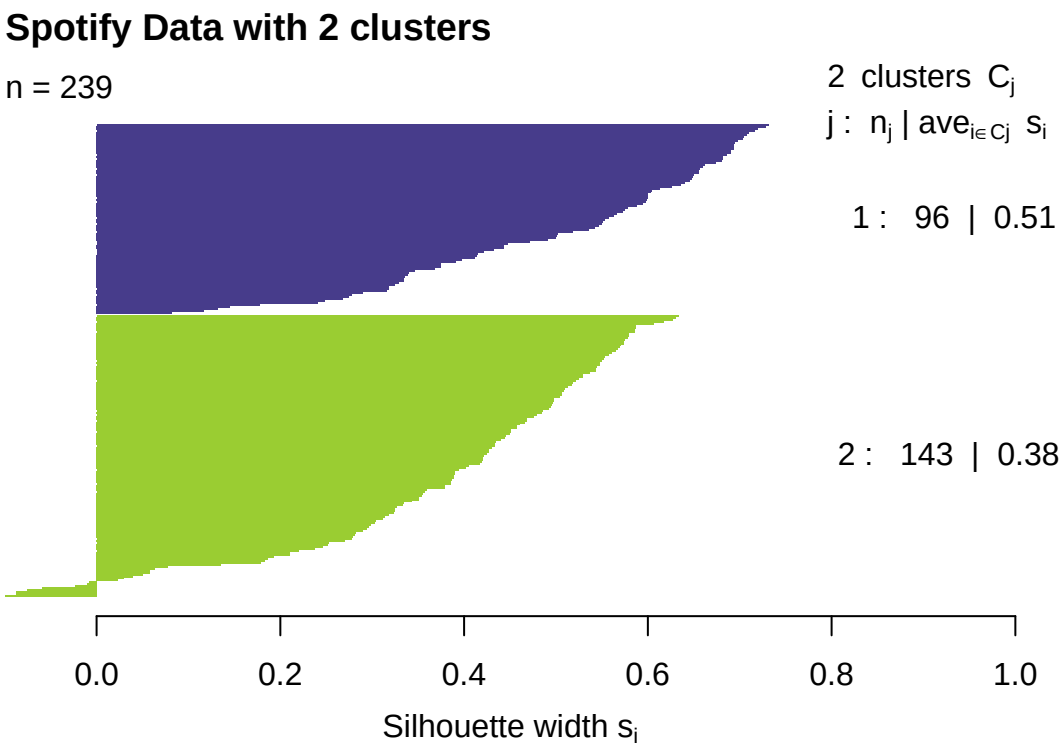

Clustering K = 2
danceability
energy
−2
−1
0
1
2



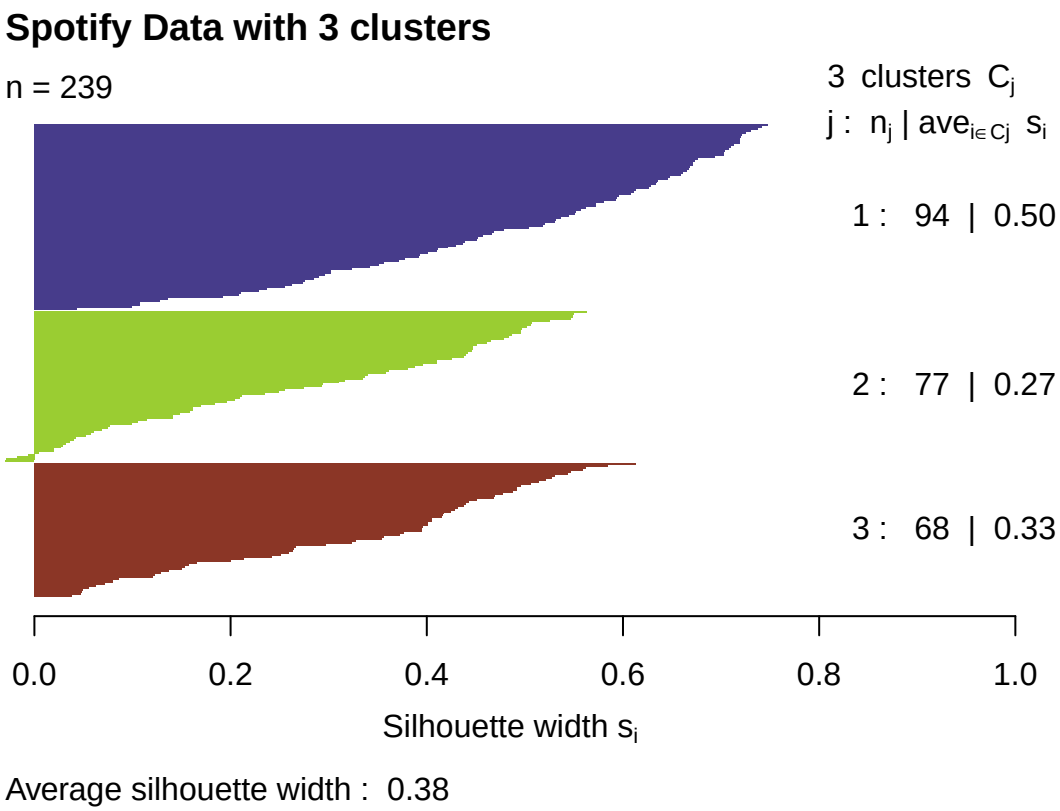
We can see when we move from two clusters to three clusters there are some overlapping data. This means that two of the genre might be too close that they are mixed well as compared to the third genre.

Silhouette Plots

Let's confirm the number of clusters with silhouette plots and average silhouette value. We select the cluster which has the higher average silhouette width.



Average silhouette width : 0.43



Result for average silhouette width: Data with two clusters = 0.43 Data with three clusters = 0.38

Therefore we confirm that the data can be divided into two clusters. Since the average silhouette width for data with two clusters is higher than the average silhouette width for data with three clusters.

K= 2 is good fit for the audio features of the songs available on spotify.

External Validation

We will cross tabulate the genre of the songs with our clusters in order to validate our fit.We will look at the diagonal of these tables. More values on the diagonal the better our fit. But not all table have proper diagonals. To check it further we will calculate the rand index. Higher the rand index the better is our fit. Although the rand index does not give proper information if the tabulation is not square. Since, the table for k=2 clusters and genre will be a 2x3 table, therefore we will also calculate the adjusted rand index. Higher the adjusted rand index the better is our fit.

```
## length of data with 2 cluster = 239

## length of data with 3 cluster = 239

## length of data = 239

##    genre
##     rock pop acoustic
##   1    1   5       90
##   2   58  75       10
```

```
##    genre
##     rock pop acoustic
##   1    1   5       88
##   2   41  30        6
##   3   17  45        6
```

```
## Rand index for Table with two clusters = 0.734256882669386
```

```
## Adjusted Rand index for Table with two clusters = 0.474148108110552
```

```
## Rand index for Table with three clusters = 0.757462817763088
```

```
## Adjusted Rand index for Table with two clusters = 0.460389208790751
```

Since the adjusted rand index for k=2 is higher than the adjusted rank index of k=3, our claim is validated.

Conclusion

The clusters tell us that depending on the audio features the songs can be divided into two groups. Whereas our data was originally divided into three genres. The data has two clusters since some audio features of pop and rock songs are very similar and hence close enough to be in one cluster. Also some audio features of pop and acoustic songs are very similar and hence close enough to be in one cluster.

Appendix

```
knitr::opts_chunk$set(echo = TRUE)
library(cluster) # Clustering
library(e1071) # Class Agreement
library(mclust)# Adjusted Rand Index
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
#                                                                               #
#                            # ETL #                                            #
#                                                                               #
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
# loading the spotify data
load("D:/Anisha/Projects/Cluster Analysis 1/data_spotify_songs.rda")
# looking at the data
head(spotify)
# looking at the structure of the data
str(spotify)
# dropping the categorical vectors
spotify_num = spotify[-c(1,2,3,4,5)]

# Scaling the data
spotify_num = scale(spotify_num)
# looking at the structure of the modified data
str(spotify_num)
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
#                                                                               #
```

```r
#                            # Visualisation #                                  #
#                                                                             #
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#

# pairs plot of all the audio features
pairs(spotify_num, gap = 0, pch = 19, col = adjustcolor(3, 0.4))
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
#                                                                             #
#                       # Calinkski - Harabasz index #                        #
#                                                                             #
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#

# Setting the value of k
K <- 10

# Creating an empty vector for sum of squares
wss <- bss <- rep(NA, K)

# Creating a loop for each value of K
for ( k in 1:K ) {
  fit <- kmeans(spotify_num, centers = k, nstart = 50)
  # Storing the sum of square in the empty vector created
  wss[k] <- fit$tot.withinss
  bss[k] <- fit$betweenss
  }

# Computing the Calinkski - Harabasz index
N <- nrow(spotify_num)

# Setting the value of CH index for K=1 as zero
ch <- ( bss/(1:K - 1) ) / ( wss/(N - 1:K) )
ch[1] <- 0
plot(1:K, ch, type = "b", xlab = "K values", ylab = "Calinkski - Harabasz Index")
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
#                                                                             #
#                       # Fitting the clusters #                              #
#                                                                             #
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
# Fitting K=2 clusters to the data
fit_2 =  kmeans(spotify_num, centers = 2)

# Fitting K=3 clusters to the data
fit_3 = kmeans(spotify_num, centers = 3)
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
#                                                                             #
#                       # Plotting the clusters #                             #
#                                                                             #
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#

# plotting the  fitted clusters
symb <- c(15, 16, 17) # creating a pch vector
col <- c("slateblue4", "olivedrab3", "tomato4") # creating a colours plots
```

```r
# pairs plot for K=2
pairs(spotify_num, gap = 0, pch = symb[fit_2$cluster],
col = adjustcolor(col[fit_2$cluster]), main = "Clustering K = 2")
# pairs plot for K=3
pairs(spotify_num, gap = 0, pch = symb[fit_3$cluster],
col = adjustcolor(col[fit_3$cluster]), main = "Clustering K = 3")
# pairs plot for K=2
pairs(spotify_num[,c(1,3,5,7)], gap = 0, pch = symb[fit_2$cluster],
col = adjustcolor(col[fit_2$cluster]), main = "Clustering K = 2")
# pairs plot for K=3
pairs(spotify_num[,c(1,3,5,7)], gap = 0, pch = symb[fit_3$cluster],
col = adjustcolor(col[fit_3$cluster]), main = "Clustering K = 3")
# pairs plot for K=2
pairs(spotify_num[,c(2,3)], gap = 0, pch = symb[fit_2$cluster],
col = adjustcolor(col[fit_2$cluster]), main = "Clustering K = 2")
# pairs plot for K=3
pairs(spotify_num[,c(2,3)], gap = 0, pch = symb[fit_3$cluster],
col = adjustcolor(col[fit_3$cluster]), main = "Clustering K = 3")
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
#                                                                               #
#                       # Silhouette #                                          #
#                                                                               #
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#

#  Constructing a distance matrix
d <- dist(spotify_num, method = "euclidean")^2 # Squared eucidean distance
sil_2 <- silhouette(fit_2$cluster, d)
sil_3 <- silhouette(fit_3$cluster, d)

# Producing silhouette plots
plot(sil_2, col = adjustcolor(col[1:2]), main = "Spotify Data with 2 clusters")
plot(sil_3, col = adjustcolor(col), main = "Spotify Data with 3 clusters")
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
#                                                                               #
#                       # Rand Index #                                          #
#                                                                               #
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#

# Extracting the genre column for external validation
genre <- spotify$genre

# Looking at the lengths of the clusters and genre
message(paste("length of data with 2 cluster = "),length(fit_2$cluster))
message(paste("length of data with 3 cluster = "),length(fit_3$cluster))
message(paste("length of data = "),length(genre))

# Creating a cross tabulation between the clusters and genre
# k=2 and genre
tab_2 <- table(fit_2$cluster, genre)
tab_2
# k=3 and genre
tab_3 <- table(fit_3$cluster, genre)
tab_3
```

```r
# rand index and Adjusted rand index

#k=2
message(paste("Rand index for Table with two clusters = "),
        classAgreement(tab_2)$rand)
message(paste("Adjusted Rand index for Table with two clusters = "),
        adjustedRandIndex(fit_2$cluster, genre))

#k=3
message(paste("Rand index for Table with three clusters = "),
        classAgreement(tab_3)$rand)
message(paste("Adjusted Rand index for Table with two clusters = "),
        adjustedRandIndex(fit_3$cluster, genre))
```