Assignment 1 (Statistical Machine Learning)

- **Shubhang Periwal (19201104)**

Abstract

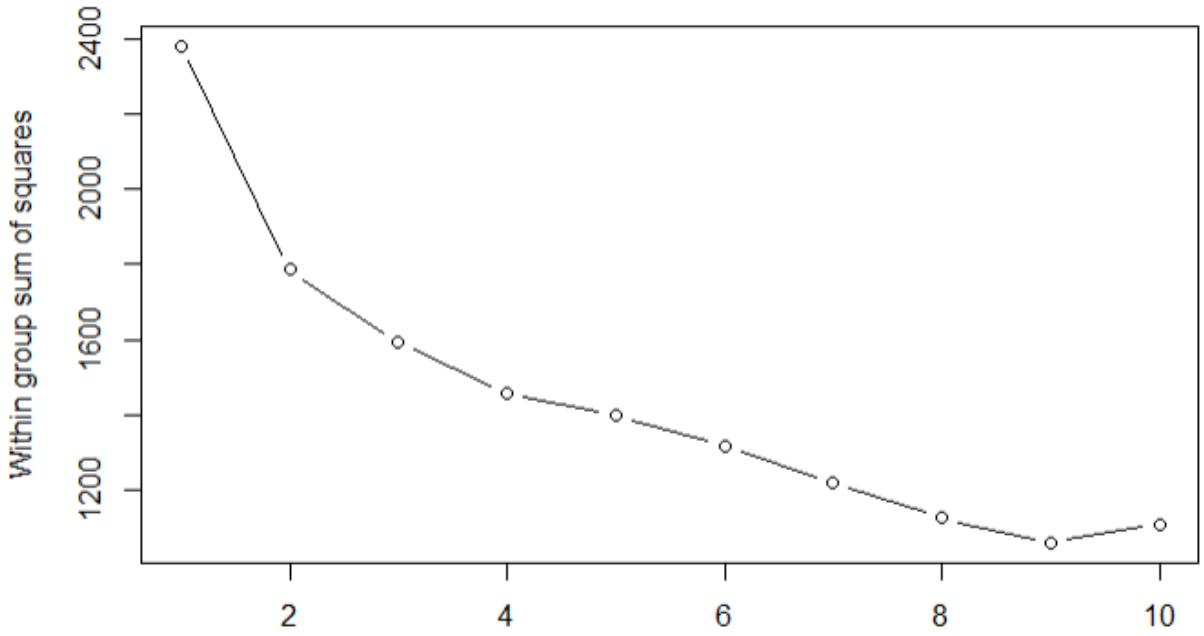
This assignment seeks to us to do a complete cluster analysis of the Spotify audio features data using k-means followed by k-medoids. The dataset data_spotify_songs.rda contains data about audio features for a collection of songs. The songs belong to three genres: acoustic, pop and rock. The dataset contains 239 songs. The properties that we can use for clustering are song duration, danceability, energy, liveness, speech, tempo and audio valence.

Methodology and Observations

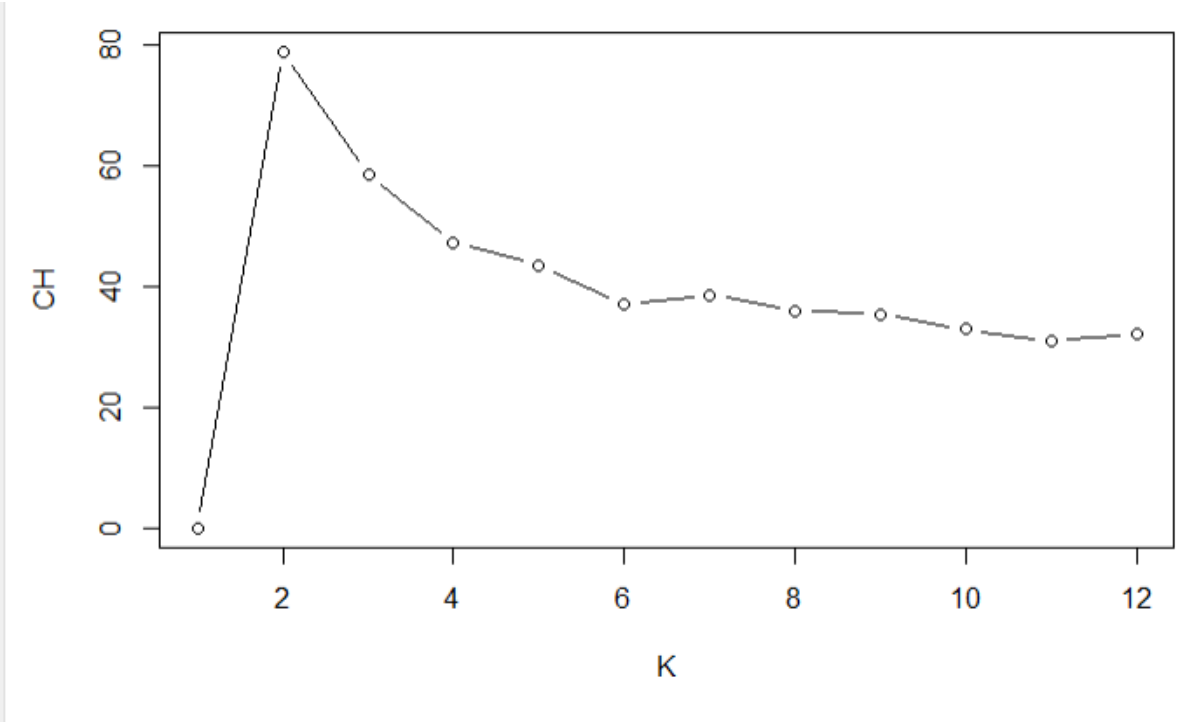
The data is sliced into a subset which removes the song name and band name as features. (We could have included them into the data by using one-hot encoding, or by creating subclasses of each band but this would not help us classify new songs from new bands so I'll be using features which are common to all songs).

The data is then scaled as the input variables do not have a uniform unit, and we need to normalize the data into a specified range so that we can use this data for clustering. Dataset is standardized using scale function. Then we perform a pair plot to see the variation of every feature with respect to each other as well as how a pair separates genre.

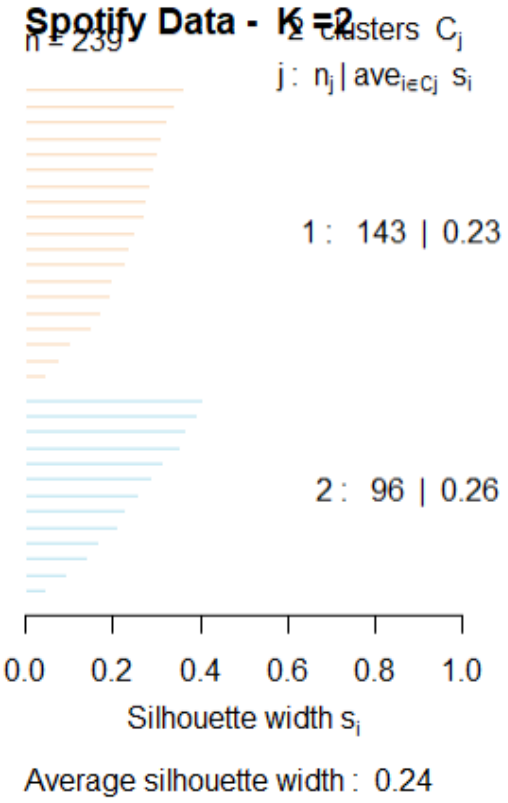
We then perform clustering multiple times with different number of cluster centers to find the optimum number of centers in the cluster. We calculate WGSS, WTSS, BGSS to calculate internal and external validation index. Firstly, we plot within sum of squares against the number of clusters, to determine the optimum number of clusters.

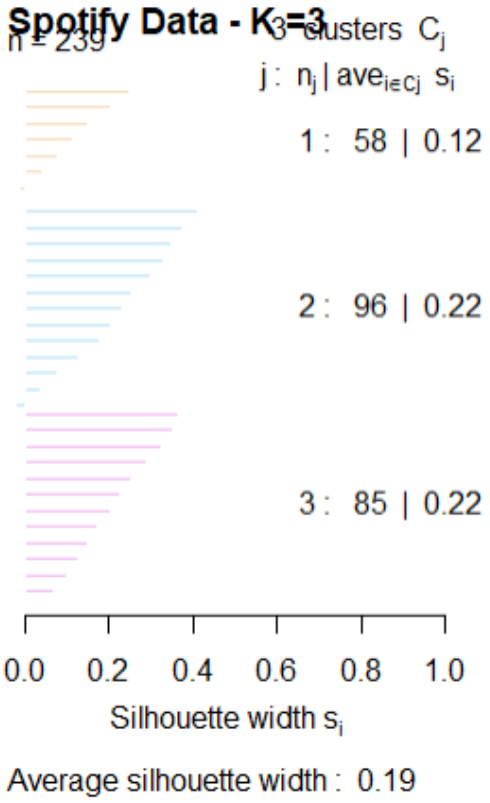


It is unclear, as using the elbow rule, it could either be 2 or 3 clusters, as the value is constantly reducing. So, we now use another validation method to compute the optimum number of clusters. We now calculate Calinski-Harabasz index. In this we can see that the peak is at the second index hence we can conclude that according to CH-Index, the optimum number of clusters is 2.



We now perform external validation test for both 2 and 3 clusters. I am then performing silhouette test for both 2 and 3 clusters. The average width is higher for 2 cluster than 3 clusters. So, according to silhouette test, model with 2 clusters is better.





Class Agreement computes several coefficients of agreement between the columns and rows of a 2-way contingency table. We then perform classAgreement tests on the data to compute rand and crand index on the data. Rand for 2 clusters : 0.7342, Crand for 2 clusters: 0.4741, Rand for 3 clusters : 0.7744, Crand: .50

Conclusion and Future Works

From the table between genre and the models we can see that 2 clusters differentiate genres in a much more accurate way.

```
            1  2

rock       58  1
pop        75  5
acoustic   10 90
            1  2  3

rock       35  1 23
pop        19  5 56
acoustic    4 90  6
```

2 Clusters :

Correctly Classified : 223

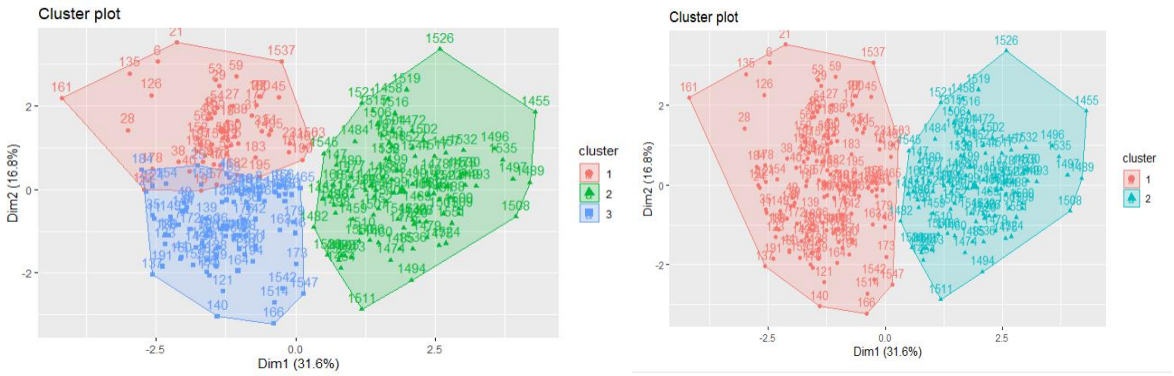
Incorrect Classified : 16 (Different cluster as features of pop and rock are in 1)

3 Clusters :

Correctly Classified : 181

Incorrect Classified : 58

From this we can conclude that having 2 clusters is better than having 3 clusters even though the indices of 3 cluster models were high. We can also conclude that acoustic is different from pop and rock. So, we can use clustering to separate acoustic from pop and rock, but we need to modify features to correctly separate pop and rock genres. We can use MFCC features from individual songs or use dimensionality reduction techniques such as PCA before performing clustering. We can even see in the cluster plots about the overlap of data in case of 3 clusters, where as no overlap incase of 2 clusters.



References

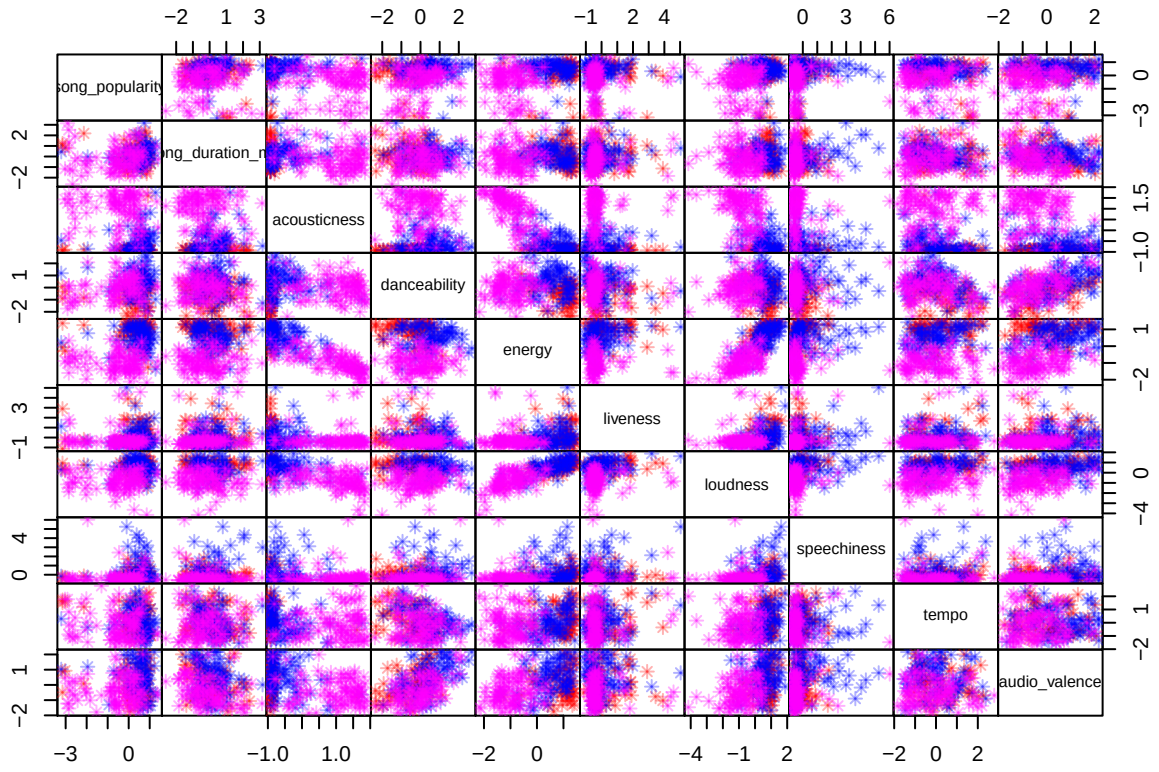
- Lab 2 material
- Lab 3 material
- https://www.rdocumentation.org/packages/factoextra/versions/1.0.6/topics/fviz_cluster
- https://stats.stackexchange.com/questions/263374/clusters-and-data-visualisation-in-r

Appendix

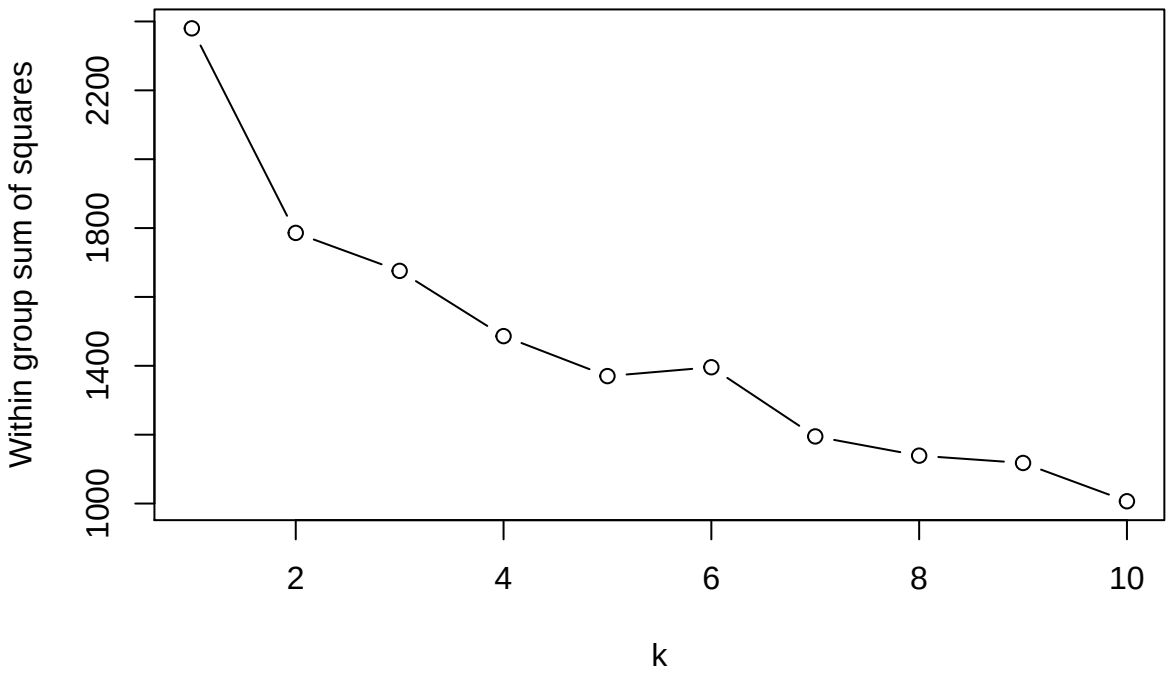
```
# removing categorical variables
x <- spotify[, -1:-3]
x = scale(x)
```

Scaling the data here is necessary the distribution and units along the column is not same

```
# plot the data to you see if any clusters are present in the data.
ccol <- c("red", "blue", "magenta")
pairs(x, gap = 0, pch = 8, col = adjustcolor(ccol[spotify$genre], 0.3))
```



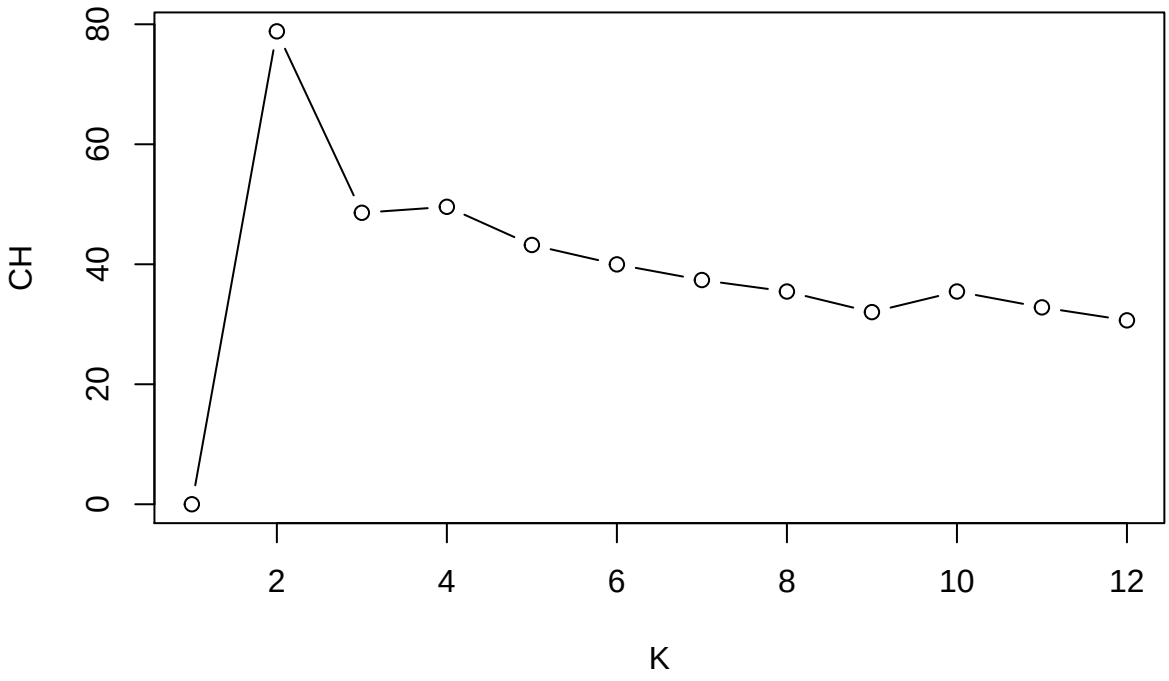

```
# This is to show variation of genre based on each feature
WGSS <- rep(0, 10)  #setting up a vector to record the WGSS for each clustering solution.

n <- nrow(x)  #to find the number of rows in the dataset
WGSS[1] <- (n - 1) * sum(apply(x, 2, var))  #to find the variance of each column in the
# dataset and then adding them up.

for (k in 2:10) {
    WGSS[k] <- sum(kmeans(x, centers = k)$withinss)
}
plot(1:10, WGSS, type = "b", xlab = "k", ylab = "Within group sum of squares")
```



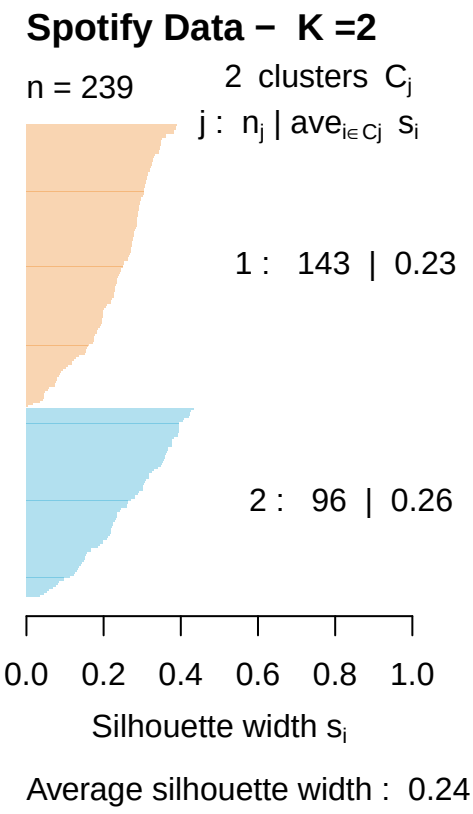

```r
# ch index for validation as it is unclear from the dataset
k <- 12
wss <- bss <- rep(0, k)
for (k in 1:k) {
    # run kmeans for each value of k
    fit <- kmeans(x, centers = k)
    wss[k] <- fit$tot.withinss  # store total within sum of squares
    bss[k] <- fit$betweenss
}

N <- nrow(x)
ch <- (bss/(1:k - 1))/(wss/(N - 1:k))
ch[1] <- 0
ch
```

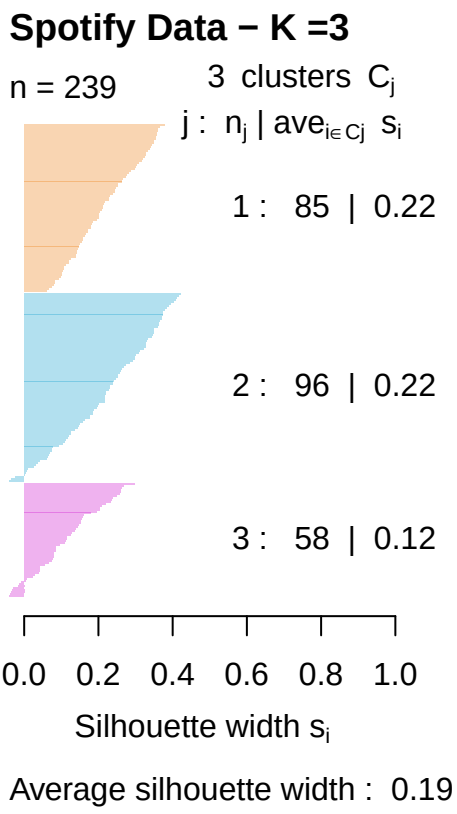
```
##  [1]  0.00000 78.82269 48.57951 49.57608 43.21767 39.97980 37.37033
##  [8] 35.45590 32.02386 35.46004 32.81733 30.65108
```

```r
plot(1:k, ch, type = "b", ylab = "CH", xlab = "K")
```




```
d <- dist(x, method = "euclidean")
fit2 <- kmeans(x, centers = 2, nstart = 50)
fit3 <- kmeans(x, centers = 3, nstart = 50)
sil2 <- silhouette(fit2$cluster, d)
sil3 <- silhouette(fit3$cluster, d)
col <- c("darkorange2", "deepskyblue3", "magenta3")
par(mfrow = c(1, 2))
plot(sil2, col = adjustcolor(col[1:2], 0.3), main = "Spotify Data -  K =2")
plot(sil3, col = adjustcolor(col, 0.3), main = "Spotify Data - K =3")
```






```
# used to calculate rand and crand index
classAgreement(table(spotify$genre, fit2$cluster))
```

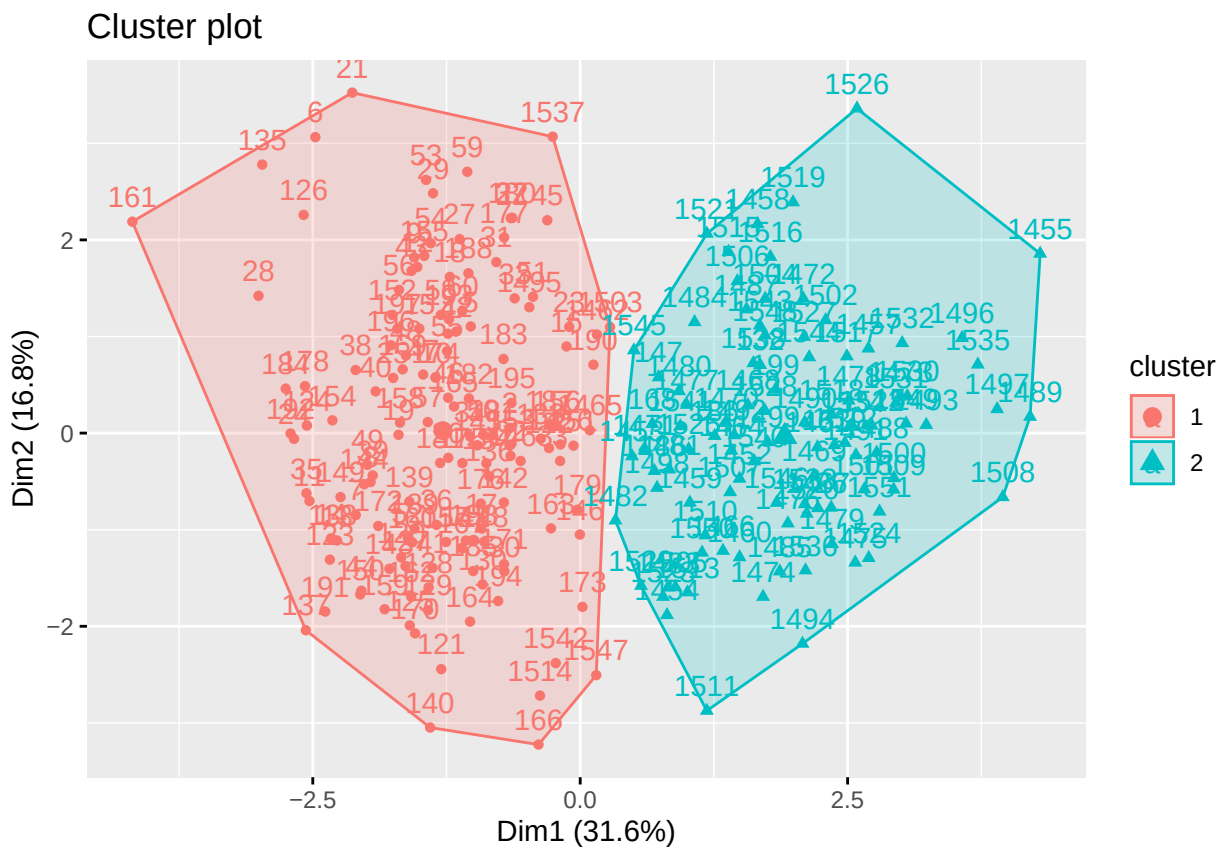
```
## $diag
## [1] 0.2635983
##
## $kappa
## [1] -0.02585114
##
## $rand
## [1] 0.7342569
##
## $crand
## [1] 0.4741481
```

```
classAgreement(table(spotify$genre, fit3$cluster))
```

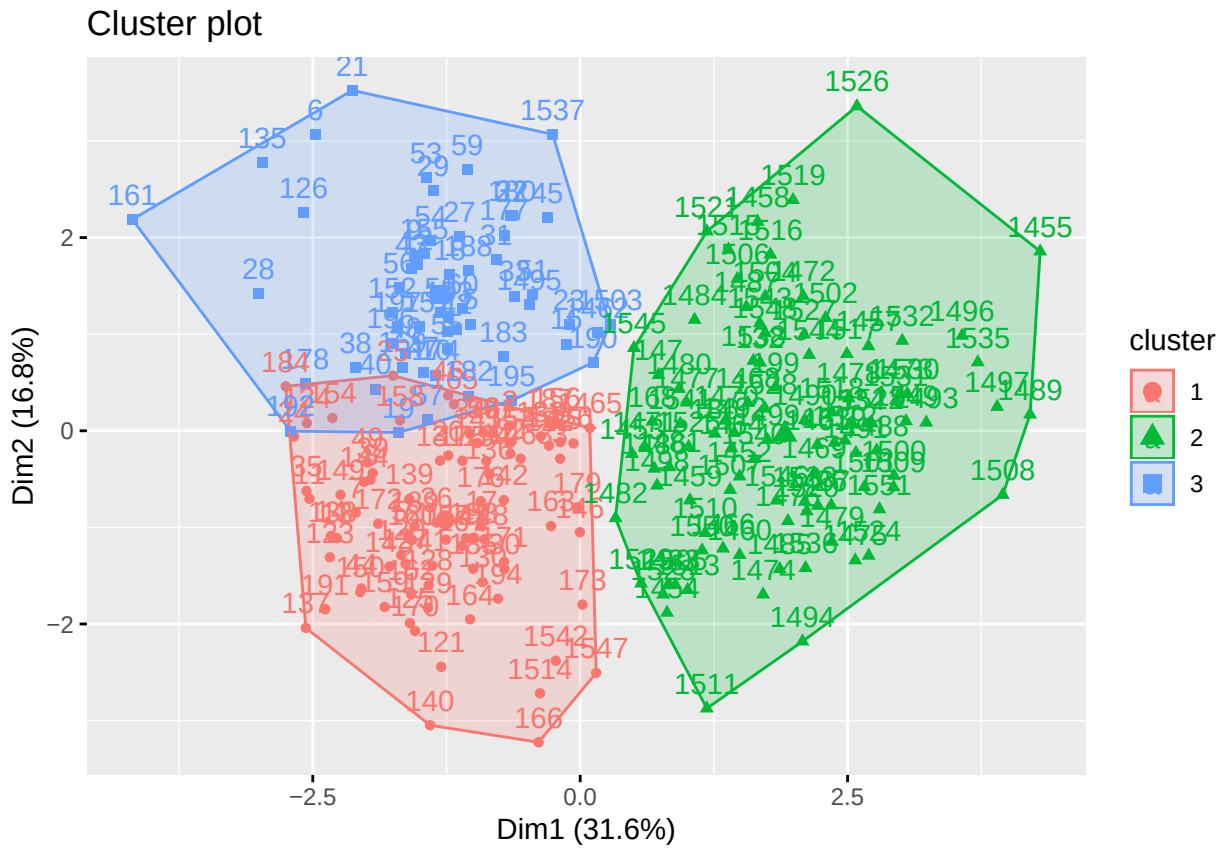
```
## $diag
## [1] 0.1338912
##
## $kappa
## [1] -0.2808212
##
## $rand
## [1] 0.7744805
```

```
##
## $crand
## [1] 0.5007643
```

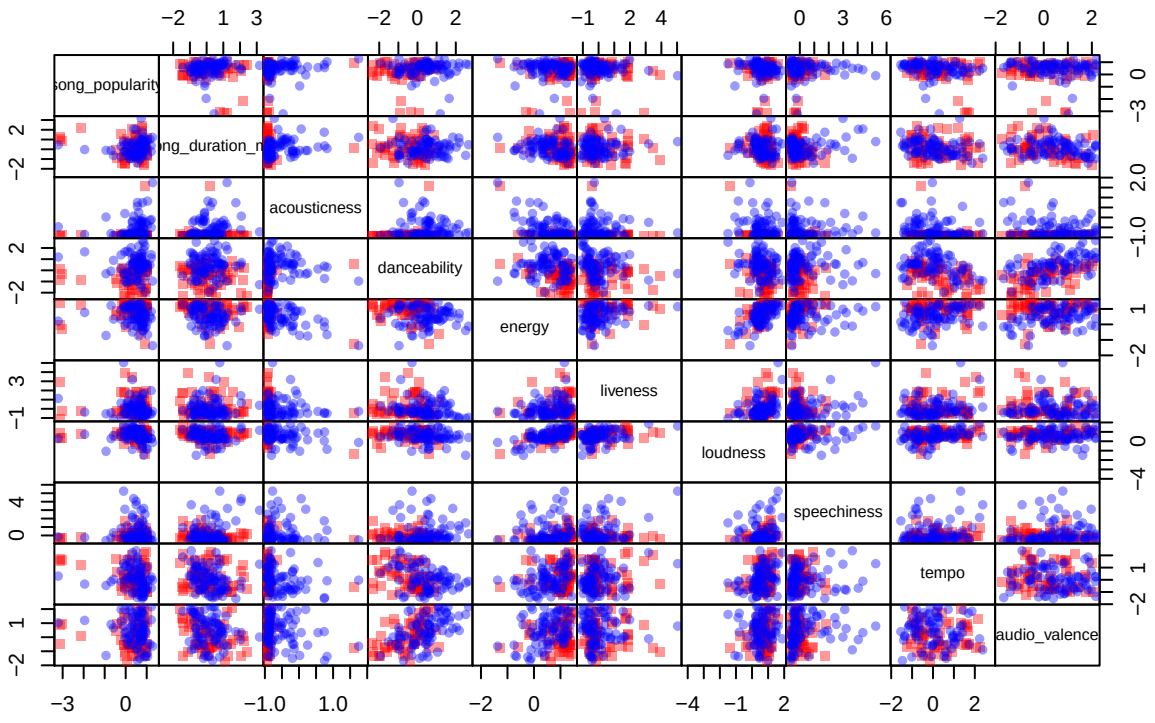
```
fviz_cluster(fit2, data = x)
```



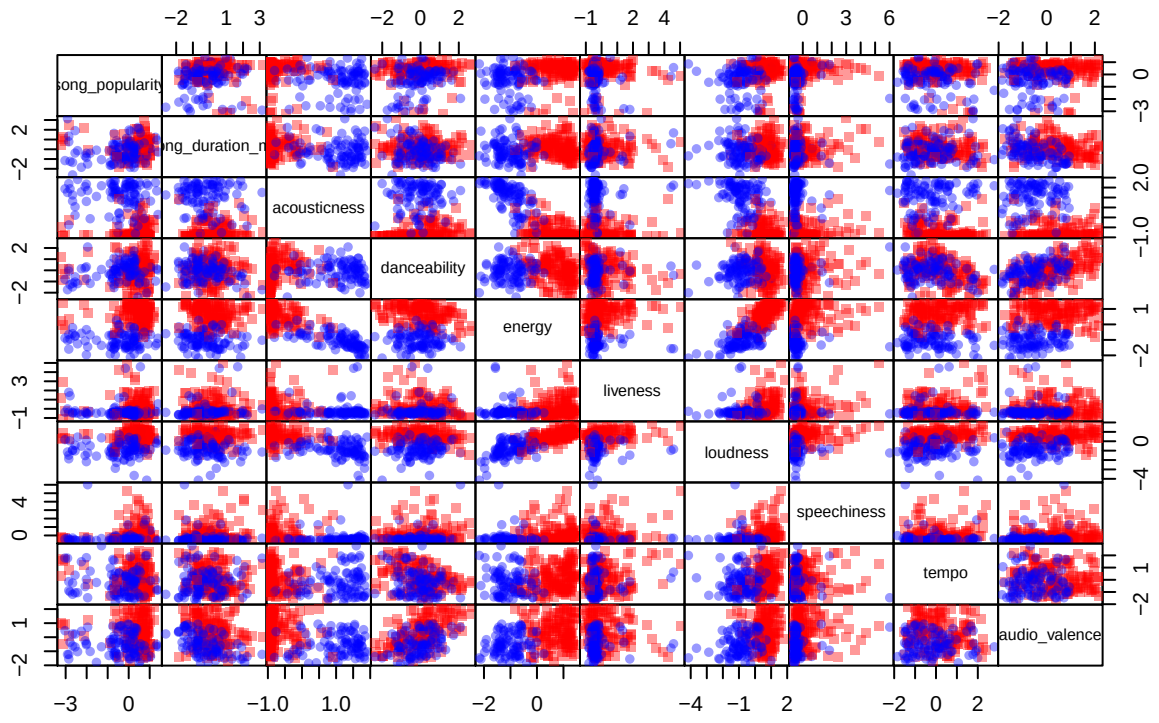

```
fviz_cluster(fit3, data = x)
```




```
fitkm <- kmeans(x, centers = 2, nstart = 20)

par(mfrow = c(1, 2))  # split the plot window in 2 screens
symb <- c(15, 16, 17)
col <- c("red", "blue")

# plot with symbol and color corresponding to the species
pairs(x, gap = 0, pch = symb[spotify$genre], col = adjustcolor(col[spotify$genre],
    0.4), main = "Songs geners on Spotify")
```




```
# plot with symbol and color corresponding to the species
pairs(x, gap = 0, pch = symb[fitkm$cluster], col = adjustcolor(col[fitkm$cluster],
    0.4), main = "Clustering result - K = 2")
```




```
# computing table to calculate how different genres associated to different
# genres
ftable(spotify$genre, fitkm$cluster)
```

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

```
ftable(spotify$genre, fit3$cluster)
```

```
##            1  2  3
##
## rock      23  1 35
## pop       56  5 19
## acoustic   6 90  4
```