Spotify Clustering

2022-12-23

setwd("C:/Users/Joseph/Documents/Codes/2022/mvtec-2022/mvtec-statsprogramming/statsprog-09")  
raw\_data <- read.csv("data/cleaneddata.csv", sep=",");  
  
# # Uncomment if the code will look for 'cluster' package  
# install.packages("cluster")  
  
# set.seed(1)  
# sample <- sample\_n(raw, 1000)  
# names(raw)  
  
dd <- raw\_data   
names(dd)

## [1] "popularity" "duration\_ms" "explicit" "danceability"   
## [5] "energy" "key" "loudness" "mode"   
## [9] "speechiness" "acousticness" "instrumentalness" "liveness"   
## [13] "valence" "tempo" "time\_signature" "track\_genre"   
## [17] "multiple\_artists" "tempo\_cat"

dim(dd)

## [1] 3000 18

summary(dd)

## popularity duration\_ms explicit danceability   
## Min. : 0.00 Min. : 28946 Length:3000 Min. :0.0000   
## 1st Qu.:19.00 1st Qu.:167156 Class :character 1st Qu.:0.4270   
## Median :30.00 Median :215889 Mode :character Median :0.5480   
## Mean :32.78 Mean :224103 Mean :0.5410   
## 3rd Qu.:46.00 3rd Qu.:268796 3rd Qu.:0.6653   
## Max. :97.00 Max. :594533 Max. :0.9750   
## energy key loudness mode   
## Min. :0.000242 Length:3000 Min. :-42.631 Length:3000   
## 1st Qu.:0.423000 Class :character 1st Qu.:-11.155 Class :character   
## Median :0.666500 Mode :character Median : -7.498 Mode :character   
## Mean :0.624997 Mean : -8.888   
## 3rd Qu.:0.861250 3rd Qu.: -5.181   
## Max. :0.999000 Max. : 0.377   
## speechiness acousticness instrumentalness liveness   
## Min. :0.00000 Min. :0.000001 Min. :0.0000000 Min. :0.0112   
## 1st Qu.:0.03470 1st Qu.:0.013300 1st Qu.:0.0000000 1st Qu.:0.1010   
## Median :0.04690 Median :0.217000 Median :0.0000889 Median :0.1410   
## Mean :0.07862 Mean :0.348478 Mean :0.1887061 Mean :0.2379   
## 3rd Qu.:0.07490 3rd Qu.:0.678250 3rd Qu.:0.1462500 3rd Qu.:0.3070   
## Max. :0.96200 Max. :0.996000 Max. :1.0000000 Max. :0.9920   
## valence tempo time\_signature track\_genre   
## Min. :0.0000 Min. : 0.0 Min. :0.000 Length:3000   
## 1st Qu.:0.2500 1st Qu.:100.0 1st Qu.:4.000 Class :character   
## Median :0.4730 Median :121.9 Median :4.000 Mode :character   
## Mean :0.4771 Mean :121.9 Mean :3.894   
## 3rd Qu.:0.6933 3rd Qu.:139.7 3rd Qu.:4.000   
## Max. :0.9850 Max. :220.0 Max. :5.000   
## multiple\_artists tempo\_cat   
## Mode :logical Length:3000   
## FALSE:2927 Class :character   
## TRUE :73 Mode :character   
##   
##   
##

attach(dd)

# Set a list of numerical variables  
names(dd)

## [1] "popularity" "duration\_ms" "explicit" "danceability"   
## [5] "energy" "key" "loudness" "mode"   
## [9] "speechiness" "acousticness" "instrumentalness" "liveness"   
## [13] "valence" "tempo" "time\_signature" "track\_genre"   
## [17] "multiple\_artists" "tempo\_cat"

numeric\_vars <- c("loudness", "energy", "acousticness", "instrumentalness", "valence")  
dcon <- dd %>% select(all\_of(numeric\_vars))  
dim(dcon)

## [1] 3000 5

# CLUSTERING

## KMEANS RUN, BUT HOW MANY CLASSES?

# k1 <- kmeans(dcon,5)  
k1 <- kmeans(dcon, 5)  
names(dcon)

## [1] "loudness" "energy" "acousticness" "instrumentalness"  
## [5] "valence"

print(k1)

## K-means clustering with 5 clusters of sizes 268, 1000, 970, 629, 133  
##   
## Cluster means:  
## loudness energy acousticness instrumentalness valence  
## 1 -16.653843 0.2919045 0.6755804 0.45942345 0.3105297  
## 2 -4.325700 0.8448720 0.1457188 0.07555445 0.5265169  
## 3 -7.514068 0.6587546 0.3123390 0.13592307 0.5194282  
## 4 -11.491707 0.4632712 0.4906543 0.21662819 0.4710480  
## 5 -25.251895 0.1616488 0.8050250 0.74687103 0.1621206  
##   
## Clustering vector:  
## [1] 2 2 2 4 4 3 2 4 3 3 2 4 3 3 5 4 3 2 3 3 4 4 2 3 3 4 2 4 2 2 2 2 3 5 2 2 3  
## [38] 1 3 2 1 2 2 1 4 1 2 2 2 2 1 4 2 3 2 5 3 3 2 2 4 2 2 2 3 2 3 1 2 4 3 3 2 3  
## [75] 2 3 2 3 3 2 4 4 2 3 5 3 2 3 3 3 2 4 4 2 2 2 3 3 3 4 3 4 2 4 2 2 1 3 3 3 4  
## [112] 4 3 2 2 2 1 1 4 2 3 1 2 4 4 1 3 3 4 4 2 2 3 3 2 4 3 2 2 3 3 3 4 2 1 2 2 5  
## [149] 3 2 2 5 5 4 3 4 1 2 3 2 2 2 1 3 3 2 2 2 4 2 2 3 3 5 3 3 3 4 2 2 3 3 3 1 2  
## [186] 2 3 2 3 1 2 4 3 2 2 2 3 5 2 2 2 5 3 2 3 2 3 4 4 3 3 4 3 2 1 3 3 4 1 3 2 4  
## [223] 1 3 2 4 2 3 3 4 2 3 2 5 3 3 2 2 3 2 3 1 4 1 1 2 3 4 2 4 3 3 3 3 2 3 2 4 2  
## [260] 3 5 2 1 2 1 2 2 3 2 2 3 1 4 1 3 3 3 3 4 4 3 3 4 2 2 3 2 3 3 1 4 2 4 2 2 3  
## [297] 3 3 5 2 3 3 2 3 3 2 5 2 5 4 3 4 3 3 1 2 1 2 3 3 4 3 1 2 3 2 1 2 4 4 4 2 3  
## [334] 2 2 3 4 4 3 4 3 3 4 1 2 2 3 2 3 2 1 4 2 2 3 4 3 2 2 2 4 3 3 3 4 2 3 4 2 1  
## [371] 2 4 2 3 2 2 3 3 3 2 2 3 1 3 4 4 3 2 4 2 3 4 3 3 4 2 4 4 1 2 2 1 4 3 5 3 4  
## [408] 5 4 3 1 4 5 1 3 4 2 1 2 3 2 5 4 2 4 3 3 1 3 3 2 3 2 4 2 2 4 4 2 1 3 3 2 4  
## [445] 4 3 1 3 3 2 4 3 3 5 4 3 4 2 2 5 4 4 4 4 4 3 3 3 2 3 3 2 3 2 2 2 4 4 3 1 1  
## [482] 1 4 4 4 5 2 3 3 2 2 5 1 4 3 3 1 4 4 2 2 4 2 2 2 2 1 2 4 2 3 1 3 3 2 2 3 1  
## [519] 2 3 3 2 2 3 3 2 4 3 2 3 3 4 3 1 2 4 3 2 1 3 4 3 3 1 4 3 2 4 2 2 2 2 5 2 3  
## [556] 3 3 2 2 2 5 2 2 1 4 4 2 4 3 2 2 2 4 3 4 4 3 2 1 2 4 4 2 3 2 3 2 4 2 2 2 3  
## [593] 3 3 5 3 1 4 3 2 4 2 3 3 2 4 1 2 2 4 4 3 2 4 3 4 1 5 3 2 3 2 4 3 3 4 2 3 1  
## [630] 3 4 3 2 2 2 2 4 5 1 2 3 3 2 3 3 2 3 2 3 1 2 4 2 2 3 4 2 1 4 4 4 4 2 2 2 2  
## [667] 2 2 4 2 4 1 3 2 3 1 4 1 4 2 2 5 4 2 2 3 2 2 3 2 2 5 3 3 2 3 3 4 3 3 4 2 3  
## [704] 3 2 2 2 2 2 3 5 4 3 2 1 4 5 3 4 2 4 1 2 2 3 2 2 3 2 4 1 2 2 2 3 4 5 4 2 2  
## [741] 4 2 3 2 2 4 2 5 3 4 2 1 3 4 2 4 4 2 2 2 2 4 2 1 4 1 4 4 1 4 3 3 3 4 4 2 3  
## [778] 2 5 2 2 2 3 3 4 4 3 2 3 1 3 4 5 1 1 2 2 2 4 3 1 3 2 2 4 4 4 2 3 2 2 2 4 2  
## [815] 3 3 2 3 3 4 3 3 1 3 3 2 3 4 2 1 2 4 4 2 2 2 1 4 2 2 1 3 3 4 2 3 2 3 2 3 3  
## [852] 3 2 4 2 4 4 3 2 3 2 3 2 4 4 3 1 1 3 2 4 2 1 4 4 5 4 3 3 4 3 3 3 3 3 2 2 3  
## [889] 2 5 2 3 4 3 3 3 3 2 3 2 1 2 2 4 5 2 2 2 2 2 3 4 3 2 2 3 2 1 4 3 3 1 2 3 4  
## [926] 1 3 4 2 4 2 3 2 3 3 3 3 2 3 4 2 2 3 2 3 2 2 2 1 3 1 2 4 1 2 1 2 4 2 4 3 2  
## [963] 2 2 4 1 3 2 2 2 2 3 4 2 2 3 4 2 2 2 2 4 3 3 2 3 2 1 2 3 1 4 2 2 3 2 3 1 3  
## [1000] 5 4 3 4 3 2 3 5 4 4 2 2 4 3 2 3 2 2 2 4 4 4 3 3 4 3 3 2 3 2 4 4 3 1 2 3 4  
## [1037] 3 4 1 5 2 4 3 2 4 3 1 3 4 2 2 2 4 4 3 3 4 1 3 4 2 4 3 4 4 4 4 4 2 2 3 2 3  
## [1074] 4 3 2 3 3 4 2 3 2 3 5 3 4 4 4 1 1 3 2 2 4 3 2 3 2 1 4 3 3 2 2 3 2 3 2 2 4  
## [1111] 2 3 2 2 2 2 4 3 3 4 2 2 3 3 2 2 4 3 3 4 2 3 4 2 3 2 4 1 3 3 1 4 1 1 2 4 2  
## [1148] 3 4 4 3 3 4 4 2 2 4 1 3 3 2 2 5 2 2 3 2 5 1 4 4 1 5 2 2 2 2 2 2 3 4 2 3 2  
## [1185] 3 1 3 4 4 3 1 3 3 5 2 3 3 3 3 1 3 5 1 2 2 2 3 3 4 5 4 5 2 1 4 3 5 4 4 3 2  
## [1222] 2 5 2 3 2 4 1 3 1 4 3 2 1 4 4 2 4 2 3 3 2 3 2 2 2 2 3 1 2 4 3 3 3 4 3 3 3  
## [1259] 3 2 3 3 3 4 4 2 3 4 1 3 3 2 4 2 2 1 3 2 4 3 2 2 2 2 1 1 3 3 3 3 2 2 3 3 3  
## [1296] 2 2 2 2 2 4 4 5 4 1 4 4 3 2 2 1 2 2 3 3 4 2 2 2 2 5 3 4 3 3 4 2 4 5 2 5 1  
## [1333] 2 4 2 3 3 4 4 3 3 2 3 3 3 2 3 3 2 2 2 4 2 2 1 3 1 3 1 3 4 3 5 4 2 3 4 2 5  
## [1370] 2 4 3 3 3 5 3 3 4 2 2 1 2 3 3 2 4 4 1 3 3 1 2 2 2 2 2 1 4 2 1 2 4 2 2 2 4  
## [1407] 3 4 3 2 3 2 4 2 2 3 2 3 2 3 3 4 3 3 3 1 3 2 4 3 2 2 2 2 2 2 3 2 3 2 2 3 3  
## [1444] 4 2 2 2 1 2 1 3 5 4 4 3 3 1 2 3 3 4 1 3 2 4 4 3 3 3 3 4 2 3 3 4 3 1 2 2 2  
## [1481] 3 3 2 3 3 1 2 4 1 2 2 5 3 4 2 3 4 2 2 2 5 2 1 1 4 3 2 2 3 2 2 2 4 2 1 4 2  
## [1518] 3 2 2 3 3 2 2 1 3 5 4 4 2 3 4 4 3 3 4 3 2 3 4 2 3 3 2 3 2 3 1 4 2 4 4 2 4  
## [1555] 2 3 2 3 2 3 2 3 2 4 3 2 2 3 3 3 3 3 4 3 3 5 2 3 2 2 3 3 2 2 4 2 2 3 3 2 2  
## [1592] 2 2 2 4 2 5 3 2 3 2 2 3 1 2 3 1 3 3 4 2 3 2 3 3 1 4 3 2 3 2 3 3 4 3 1 4 3  
## [1629] 4 5 3 3 3 2 3 3 3 1 2 4 3 3 1 2 2 4 3 2 3 3 4 4 3 3 2 2 4 4 3 4 3 3 2 1 3  
## [1666] 4 2 3 2 3 4 2 2 2 4 3 2 1 2 1 3 2 5 4 3 1 1 4 2 3 4 2 4 2 2 3 2 4 2 3 4 3  
## [1703] 2 3 4 2 4 2 4 2 3 3 2 3 3 4 3 3 3 5 3 2 2 2 1 1 2 3 4 5 4 1 2 1 2 3 4 2 5  
## [1740] 2 4 3 2 3 3 1 4 3 2 2 4 4 2 4 2 2 3 4 5 3 3 3 2 2 2 2 2 3 3 1 4 2 2 2 2 3  
## [1777] 1 2 5 3 4 3 4 2 4 4 3 3 4 4 4 2 1 1 2 4 4 4 2 2 3 2 2 3 3 3 3 2 3 3 2 2 4  
## [1814] 2 2 3 2 4 4 3 2 4 3 5 5 3 3 4 1 2 1 3 4 3 4 3 2 1 3 3 4 3 1 4 3 3 3 3 5 5  
## [1851] 3 4 3 5 2 4 3 3 3 2 2 3 4 2 2 2 3 3 4 2 2 3 3 3 2 3 4 3 2 3 2 2 2 5 1 5 4  
## [1888] 2 3 2 5 3 3 3 2 2 3 4 3 3 5 2 2 2 2 5 2 1 4 2 4 4 3 4 3 2 4 4 3 2 3 2 1 2  
## [1925] 3 3 1 4 2 3 4 1 2 3 2 3 3 1 4 3 2 3 4 4 3 4 1 3 3 1 3 3 3 3 3 4 1 5 3 2 3  
## [1962] 3 5 3 3 3 2 2 4 2 4 5 2 4 5 4 4 4 3 4 2 1 4 2 2 3 4 1 4 5 4 2 2 2 4 4 3 2  
## [1999] 3 2 2 3 5 4 2 2 4 3 4 4 3 2 4 5 4 2 2 4 2 2 4 4 3 2 3 3 2 2 4 3 1 4 4 2 3  
## [2036] 3 1 3 4 3 2 3 4 3 2 4 3 2 2 1 3 1 3 4 3 1 3 2 2 4 3 3 5 2 3 3 2 4 3 2 3 4  
## [2073] 2 3 3 2 2 4 3 2 3 3 2 3 2 2 2 2 2 2 2 2 4 3 1 3 2 3 2 3 2 2 4 5 3 2 2 3 4  
## [2110] 5 4 1 2 1 3 5 2 2 2 3 4 3 3 3 2 2 2 4 3 4 3 2 4 4 3 4 1 3 2 4 2 2 2 3 3 2  
## [2147] 3 2 4 4 2 1 3 1 2 3 3 3 2 2 4 1 3 3 3 2 3 1 3 1 4 3 4 2 4 4 2 2 3 4 3 2 3  
## [2184] 1 3 2 3 1 3 3 2 2 2 2 2 3 4 4 1 2 2 4 3 3 2 3 3 2 2 4 1 4 3 1 3 2 2 2 2 2  
## [2221] 2 2 5 2 3 4 2 3 2 4 3 2 2 4 4 2 2 3 2 2 4 4 2 3 5 4 1 5 2 3 4 3 3 4 3 3 3  
## [2258] 2 1 2 3 3 4 4 5 2 2 2 3 3 3 3 5 5 3 2 3 2 3 3 2 2 3 3 4 1 3 2 2 4 4 3 2 4  
## [2295] 2 2 3 2 2 4 3 3 2 3 2 3 3 4 2 4 3 3 2 3 1 4 5 2 2 3 1 2 2 3 4 4 3 4 3 3 3  
## [2332] 4 4 2 3 4 3 2 1 3 2 2 2 4 4 2 1 3 2 5 2 3 2 2 2 3 4 2 4 2 3 3 2 1 2 4 2 4  
## [2369] 2 3 2 2 5 3 3 4 2 2 3 3 2 4 4 2 2 4 2 2 2 4 2 3 5 4 3 5 1 1 3 3 3 2 2 1 2  
## [2406] 2 4 4 2 1 1 2 3 4 2 3 3 3 2 3 3 1 4 3 4 1 2 2 2 1 2 3 3 4 4 2 4 2 4 3 1 2  
## [2443] 2 1 5 2 4 2 3 3 4 2 2 4 4 3 2 3 3 3 2 2 4 4 2 1 1 2 3 4 4 4 4 4 3 2 2 4 5  
## [2480] 4 3 2 2 3 2 1 4 3 4 4 3 4 4 4 2 2 3 3 3 3 1 5 2 3 3 4 1 3 4 4 1 4 2 3 3 2  
## [2517] 2 4 3 4 3 5 2 3 4 3 3 2 2 5 1 3 4 3 2 3 5 3 2 4 4 3 2 1 4 3 4 5 2 1 2 5 3  
## [2554] 2 3 3 4 4 2 2 3 4 4 2 2 3 3 3 2 4 4 2 3 3 1 2 3 4 2 3 3 2 3 3 2 2 1 4 3 3  
## [2591] 2 2 4 3 2 4 2 2 4 3 3 3 1 5 2 2 5 4 3 3 4 4 3 4 2 3 4 1 3 5 3 2 3 3 3 3 1  
## [2628] 1 4 4 2 3 4 2 2 3 4 5 4 3 2 2 3 1 1 3 3 5 4 3 1 3 2 2 1 1 2 2 3 5 3 1 2 2  
## [2665] 4 4 3 4 3 2 3 4 3 2 4 2 3 3 3 3 1 2 3 5 4 2 3 2 4 3 2 2 2 5 1 2 4 3 3 2 3  
## [2702] 2 1 3 2 4 4 2 4 3 4 3 3 2 2 4 2 4 3 3 3 2 3 2 2 4 1 2 3 2 1 4 2 3 4 4 3 2  
## [2739] 4 3 2 3 2 3 4 2 3 5 3 1 4 2 1 3 3 4 2 2 3 4 1 3 3 2 1 3 4 2 3 3 3 1 3 2 5  
## [2776] 4 4 2 3 3 3 3 2 3 4 3 3 1 4 4 3 5 3 3 2 3 1 3 2 2 3 4 2 3 3 4 3 3 4 4 2 2  
## [2813] 3 3 3 2 2 3 3 3 2 2 3 1 2 3 2 2 3 5 2 2 2 2 1 3 2 4 4 2 3 2 3 3 4 1 2 2 3  
## [2850] 4 3 4 3 4 2 5 2 5 2 4 1 4 3 4 2 2 1 2 2 5 3 4 3 1 3 2 3 3 1 3 3 2 2 1 1 4  
## [2887] 2 3 2 3 3 3 3 2 4 4 2 2 4 2 3 1 2 3 5 5 3 3 3 2 1 3 2 3 2 4 2 4 5 3 3 2 3  
## [2924] 3 2 4 2 3 2 4 2 2 1 4 3 4 4 5 2 3 2 3 4 1 4 2 2 3 4 3 1 4 4 4 4 4 3 4 4 2  
## [2961] 2 3 5 3 2 2 2 2 2 2 2 4 2 3 3 3 2 2 4 3 4 3 3 2 1 2 2 4 3 3 4 5 4 3 3 4 3  
## [2998] 4 3 1  
##   
## Within cluster sum of squares by cluster:  
## [1] 1215.095 1308.479 1282.044 1274.792 1993.708  
## (between\_SS / total\_SS = 91.8 %)  
##   
## Available components:  
##   
## [1] "cluster" "centers" "totss" "withinss" "tot.withinss"  
## [6] "betweenss" "size" "iter" "ifault"

attributes(k1)

## $names  
## [1] "cluster" "centers" "totss" "withinss" "tot.withinss"  
## [6] "betweenss" "size" "iter" "ifault"   
##   
## $class  
## [1] "kmeans"

k1$size

## [1] 268 1000 970 629 133

k1$withinss

## [1] 1215.095 1308.479 1282.044 1274.792 1993.708

k1$centers

## loudness energy acousticness instrumentalness valence  
## 1 -16.653843 0.2919045 0.6755804 0.45942345 0.3105297  
## 2 -4.325700 0.8448720 0.1457188 0.07555445 0.5265169  
## 3 -7.514068 0.6587546 0.3123390 0.13592307 0.5194282  
## 4 -11.491707 0.4632712 0.4906543 0.21662819 0.4710480  
## 5 -25.251895 0.1616488 0.8050250 0.74687103 0.1621206

## LETS COMPUTE THE DECOMPOSITION OF INERTIA

Bss <- sum(rowSums(k1$centers^2)\*k1$size)  
Bss

## [1] 318346.5

Wss <- sum(k1$withinss)  
Wss

## [1] 7074.118

Tss <- k1$totss  
Tss

## [1] 86098.5

Bss+Wss

## [1] 325420.6

Ib1 <- 100\*Bss/(Bss+Wss)  
Ib1

## [1] 97.82616

# LETS REPEAT THE KMEANS RUN WITH K=5  
  
k2 <- kmeans(dcon,5)  
k2$size

## [1] 970 133 1000 268 629

Bss <- sum(rowSums(k2$centers^2)\*k2$size)  
Bss

## [1] 318346.5

Wss <- sum(k2$withinss)  
Wss

## [1] 7074.118

Ib2 <- 100\*Bss/(Bss+Wss)  
Ib2

## [1] 97.82616

Ib1

## [1] 97.82616

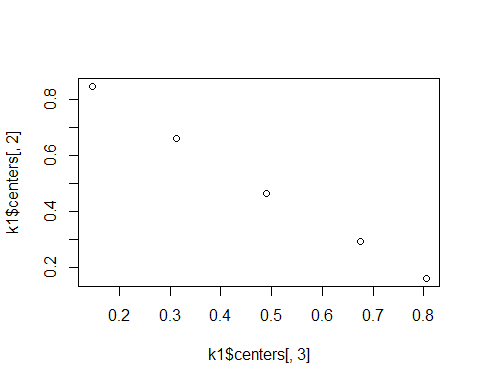
k2$centers

## loudness energy acousticness instrumentalness valence  
## 1 -7.514068 0.6587546 0.3123390 0.13592307 0.5194282  
## 2 -25.251895 0.1616488 0.8050250 0.74687103 0.1621206  
## 3 -4.325700 0.8448720 0.1457188 0.07555445 0.5265169  
## 4 -16.653843 0.2919045 0.6755804 0.45942345 0.3105297  
## 5 -11.491707 0.4632712 0.4906543 0.21662819 0.4710480

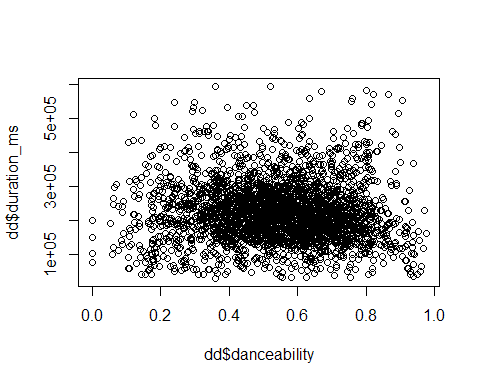
k1$centers

## loudness energy acousticness instrumentalness valence  
## 1 -16.653843 0.2919045 0.6755804 0.45942345 0.3105297  
## 2 -4.325700 0.8448720 0.1457188 0.07555445 0.5265169  
## 3 -7.514068 0.6587546 0.3123390 0.13592307 0.5194282  
## 4 -11.491707 0.4632712 0.4906543 0.21662819 0.4710480  
## 5 -25.251895 0.1616488 0.8050250 0.74687103 0.1621206

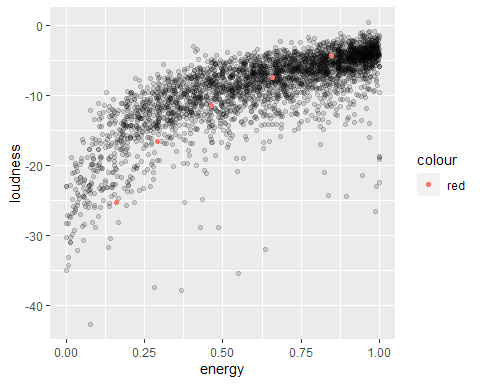
plot(k1$centers[,3],k1$centers[,2])



plot(dd$danceability, dd$duration\_ms)



ggplot() +  
 geom\_point(data = dd, aes(energy, loudness), alpha = 0.15) +  
 geom\_point(data = data.frame(k1$centers), aes(energy, loudness, color = "red"))



table(k1$cluster, k2$cluster)

##   
## 1 2 3 4 5  
## 1 0 0 0 268 0  
## 2 0 0 1000 0 0  
## 3 970 0 0 0 0  
## 4 0 0 0 0 629  
## 5 0 133 0 0 0

## WHY WE HAVE OBTAINED DIFFERENT RESULTS?, AND WHICH RUN IS BETTER?

### NOW TRY K=8

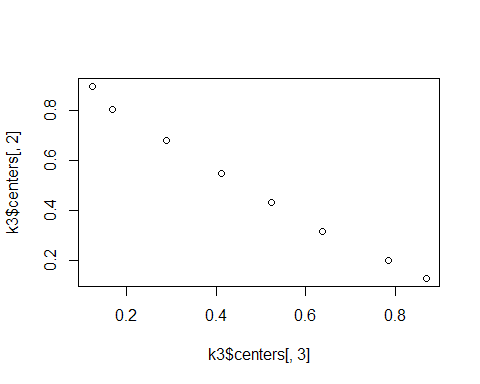
k3 <- kmeans(dcon,8)  
k3$size

## [1] 210 405 51 381 641 508 659 145

Bss <- sum(rowSums(k3$centers^2)\*k3$size)  
Wss <- sum(k3$withinss)  
  
Ib3 <- 100\*Bss/(Bss+Wss)  
Ib3

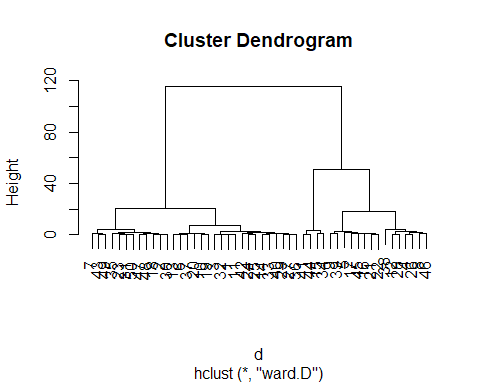
## [1] 98.91375

plot(k3$centers[,3],k3$centers[,2])



## HIERARCHICAL CLUSTERING

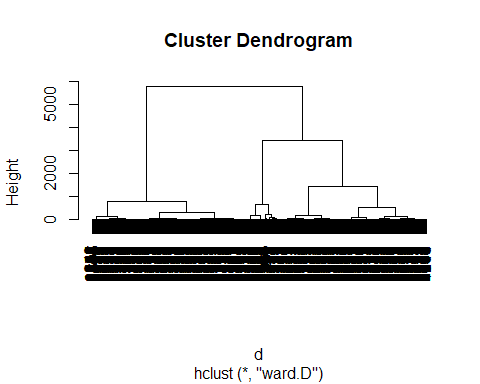
d <- dist(dcon[1:50,])  
h1 <- hclust(d,method="ward.D") # NOTICE THE COST  
plot(h1)



d <- dist(dcon)  
# h1 <- hclust(d,method="ward") # NOTICE THE COST  
# The "ward" method has been renamed to "ward.D"; note new "ward.D2"  
h1 <- hclust(d,method="ward.D") # NOTICE THE COST  
h1

##   
## Call:  
## hclust(d = d, method = "ward.D")  
##   
## Cluster method : ward.D   
## Distance : euclidean   
## Number of objects: 3000

plot(h1)



# We'll cut the tree into 4  
  
# Define condition

## BUT WE ONLY NEED WHERE THERE ARE THE LEAPS OF THE HEIGHT

## WHERE ARE THER THE LEAPS? WHERE WILL YOU CUT THE DENDREOGRAM?, HOW MANY CLASSES WILL YOU OBTAIN?

# nc = 2  
nc = 4  
  
c1 <- cutree(h1,nc)  
  
c1[1:20]

## [1] 1 1 1 2 2 1 1 2 3 1 1 2 1 1 4 2 3 1 3 1

nc = 5  
  
c5 <- cutree(h1,nc)  
  
c5[1:20]

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

table(c1)

## c1  
## 1 2 3 4   
## 1371 716 602 311

table(c5)

## c5  
## 1 2 3 4 5   
## 910 461 716 602 311

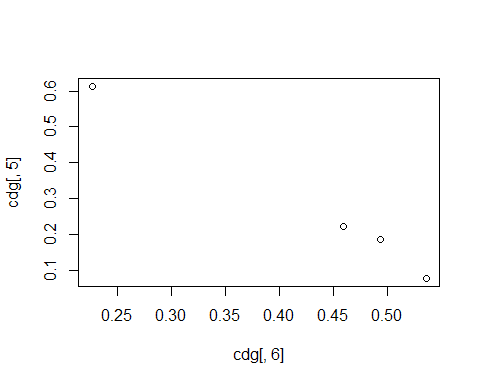
table(c1,c5)

## c5  
## c1 1 2 3 4 5  
## 1 910 461 0 0 0  
## 2 0 0 716 0 0  
## 3 0 0 0 602 0  
## 4 0 0 0 0 311

cdg <- aggregate(as.data.frame(dcon),list(c1),mean)  
cdg

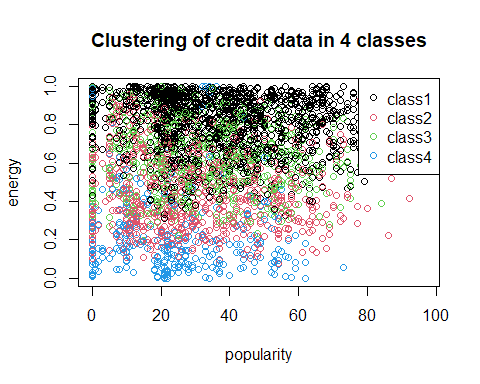
## Group.1 loudness energy acousticness instrumentalness valence  
## 1 1 -4.905945 0.8101400 0.1832846 0.07652361 0.5362125  
## 2 2 -11.880494 0.4459659 0.5002814 0.22287230 0.4594306  
## 3 3 -8.172080 0.6269236 0.3307687 0.18422472 0.4932940  
## 4 4 -20.939823 0.2172659 0.7614973 0.61326230 0.2262794

plot(cdg[,6], cdg[,5])

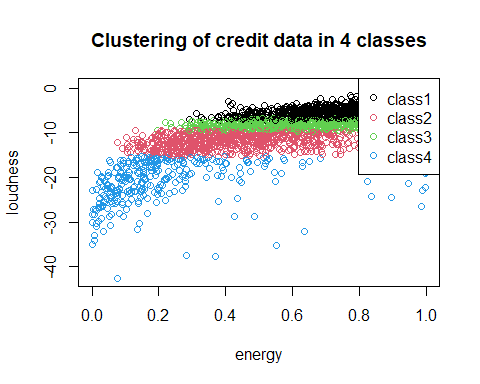


## LETS SEE THE PARTITION VISUALLY

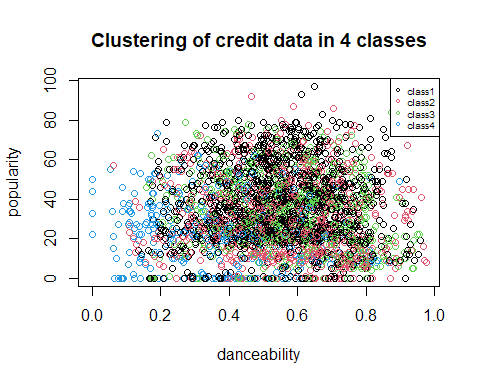
# Need to run together  
plot(popularity, energy, col=c1, main="Clustering of credit data in 4 classes")  
legend("topright", c("class1", "class2", "class3", "class4"), pch=1, col=c(1:4))



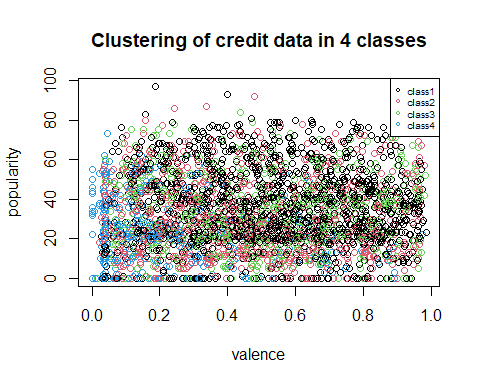
plot(energy, loudness, col=c1, main="Clustering of credit data in 4 classes")  
legend("topright", c("class1", "class2", "class3", "class4"), pch=1, col=c(1:4))



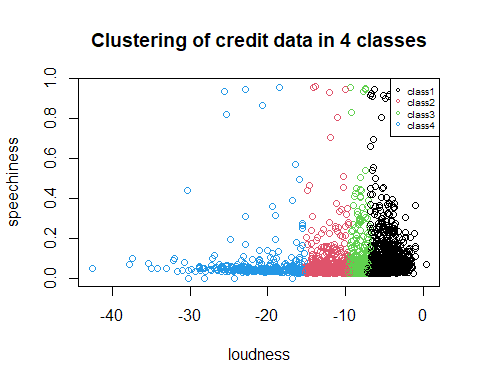
plot(danceability, popularity, col=c1, main="Clustering of credit data in 4 classes")  
legend("topright",c("class1", "class2", "class3", "class4"),pch=1,col=c(1:4), cex=0.6)



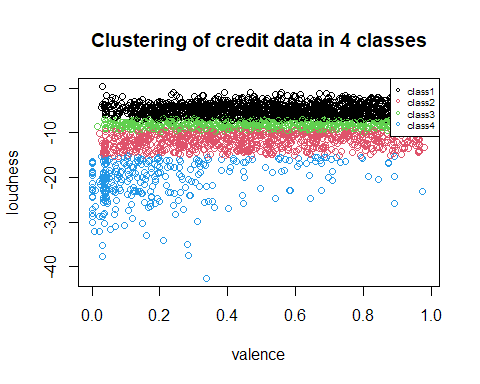
plot(valence, popularity, col=c1,main="Clustering of credit data in 4 classes")  
legend("topright",c("class1", "class2", "class3", "class4"),pch=1,col=c(1:4), cex=0.6)



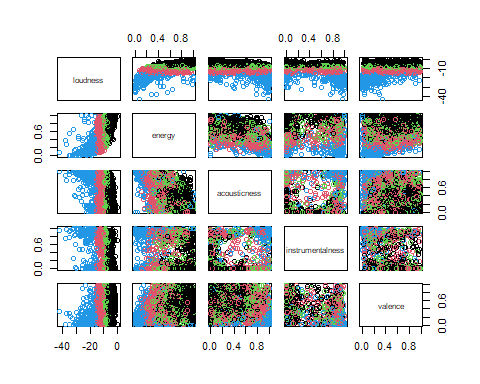
plot(loudness, speechiness,col=c1,main="Clustering of credit data in 4 classes")  
legend("topright",c("class1", "class2", "class3", "class4"),pch=1,col=c(1:4), cex=0.6)



plot(valence, loudness ,col=c1,main="Clustering of credit data in 4 classes")  
legend("topright",c("class1", "class2", "class3", "class4"),pch=1,col=c(1:4), cex=0.6)



pairs(dcon, col=c1)



# No info where the FI came from  
# plot(FI[,1],FI[,2],col=c1,main="Clustering of credit data in 3 classes")  
# legend("topleft",c("c1","c2","c3"),pch=1,col=c(1:3))  
  
# Visualizing the center / centroid of the distribution of data  
# Is it ok to remove the variables that influence PC1

## LETS SEE THE QUALITY OF THE HIERARCHICAL PARTITION

Bss <- sum(rowSums(cdg^2)\*as.numeric(table(c1)))  
  
Ib4 <- 100\*Bss/Tss  
Ib4

## [1] 380.8549

### Move to Gower mixed distance to deal

simoultaneously with numerical and qualitative data

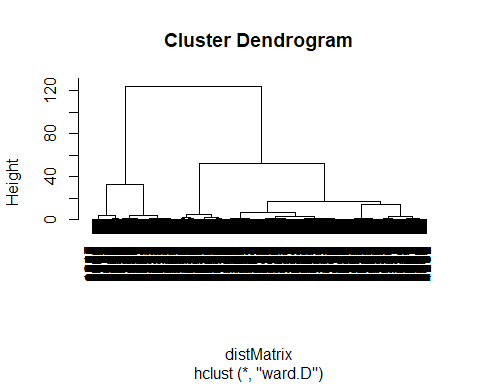
library(cluster)

## Dissimilarity matrix (Main task)

actives<-c(2:16)  
dissimMatrix <- daisy(dd[numeric\_vars], metric = "gower", stand=TRUE)   
  
distMatrix<-dissimMatrix^2

## INSTRUCTION THAT RUN THE HEIRARCHICAL CLUSTERING

h1 <- hclust(distMatrix,method="ward.D") # NOTICE THE COST  
plot(h1)



## CUT THE TREE

c2 <- cutree(h1, 4)  
  
#class sizes   
table(c2)

## c2  
## 1 2 3 4   
## 1811 520 412 257

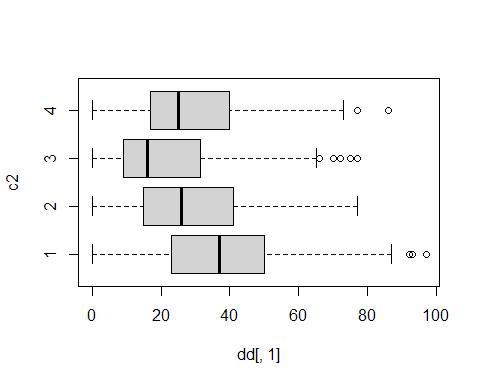
#comparing with other partitions  
table(c1,c2)

## c2  
## c1 1 2 3 4  
## 1 1172 72 124 3  
## 2 250 276 134 56  
## 3 364 105 116 17  
## 4 25 67 38 181

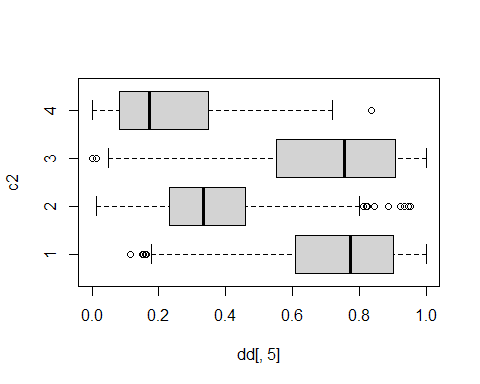
names(dd)

## [1] "popularity" "duration\_ms" "explicit" "danceability"   
## [5] "energy" "key" "loudness" "mode"   
## [9] "speechiness" "acousticness" "instrumentalness" "liveness"   
## [13] "valence" "tempo" "time\_signature" "track\_genre"   
## [17] "multiple\_artists" "tempo\_cat"

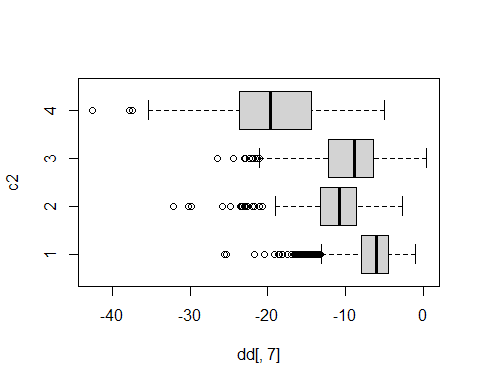
# Popularity  
boxplot(dd[,1]~c2, horizontal=TRUE)



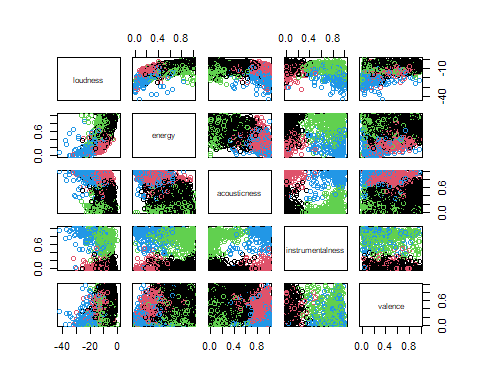
# Energy  
boxplot(dd[,5]~c2, horizontal=TRUE)



# Loudness  
boxplot(dd[,7]~c2, horizontal=TRUE)



# pairs(dcon[,1:7], col=c2)  
pairs(dcon, col=c2)

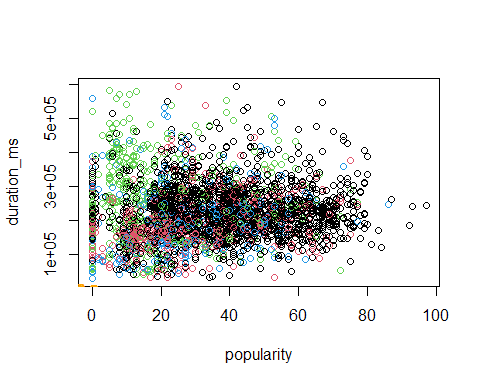


# plot(popularity, danceability, col=c2, main="Clustering of credit data in 3 classes")  
# legend("topright",levels(c2),pch=1,col=c(1:4), cex=0.6)

cdg <- aggregate(as.data.frame(dcon),list(c2),mean)  
cdg

## Group.1 loudness energy acousticness instrumentalness valence  
## 1 1 -6.583336 0.7380751 0.19736549 0.01214573 0.5458278  
## 2 2 -11.235687 0.3561565 0.81888654 0.01971245 0.4513965  
## 3 3 -9.548485 0.7146766 0.09705143 0.77733010 0.3702006  
## 4 4 -19.320650 0.2283625 0.86458366 0.83117471 0.2167026

# plot(Edad, Gastos, col= c2)  
plot(popularity, duration\_ms, col= c2)  
points(cdg[,4],cdg[,5],pch=16,col="orange")  
text(cdg[,4],cdg[,5], labels=cdg[,1], pos=2, font=2, cex=0.7, col="orange")



# potencials<-c(1,3,4,6,7,10)  
# pairs(dcon[,potencials],col=c2)  
pairs(dcon, col=c2)

