Assignment II

Aldo Giovanni e Giacomo

2023-06-02

EXERCISE 2- WE NEED SOME MUSIC!

```
library(boot)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0     v purrr   1.0.1
## v tibble  3.2.1     v dplyr   1.1.2
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.3     v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
##
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
library(combinat)
```

```
##
## Attaching package: 'combinat'
##
## The following object is masked from 'package:utils':
##
##     combn
```

Point a

DESCRIPTION OF THE VARIABLES AND EXPLORATORY ANALYSIS: Considering the data saved in spot.RDS, a description of the variables can be found on Kaggle:

- Acousticness: a confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.

- Danceability: danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.

- Duration_ms: the track length in milliseconds.

- Energy:energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy.

- Explicit: whether or not the track has explicit lyrics (true = yes it does; false = no it does not OR unknown)

- id: the Spotify ID for the track.

- Instrumentalness: predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content.

- Liveness: detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.

- Loudness: the overall loudness of a track in decibels (dB).

- Name:name of the track.

- Popularity: the popularity of a track is a value between 0 and 100, with 100 being the most popular. The popularity is calculated by algorithm and is based, in the most part, on the total number of plays the track has had and how recent those plays are. Generally speaking, songs that are being played a lot now will have a higher popularity than songs that were played a lot in the past. Duplicate tracks (e.g. the same track from a single and an album) are rated independently. Artist and album popularity is derived mathematically from track popularity.

- Release_date of the song

- Speechiness: speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value.Values below 0.33 most likely represent music and other non-speech-like tracks.

- Tempo: the overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.

- Valence: a measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).

- First_auth: first author of the song.

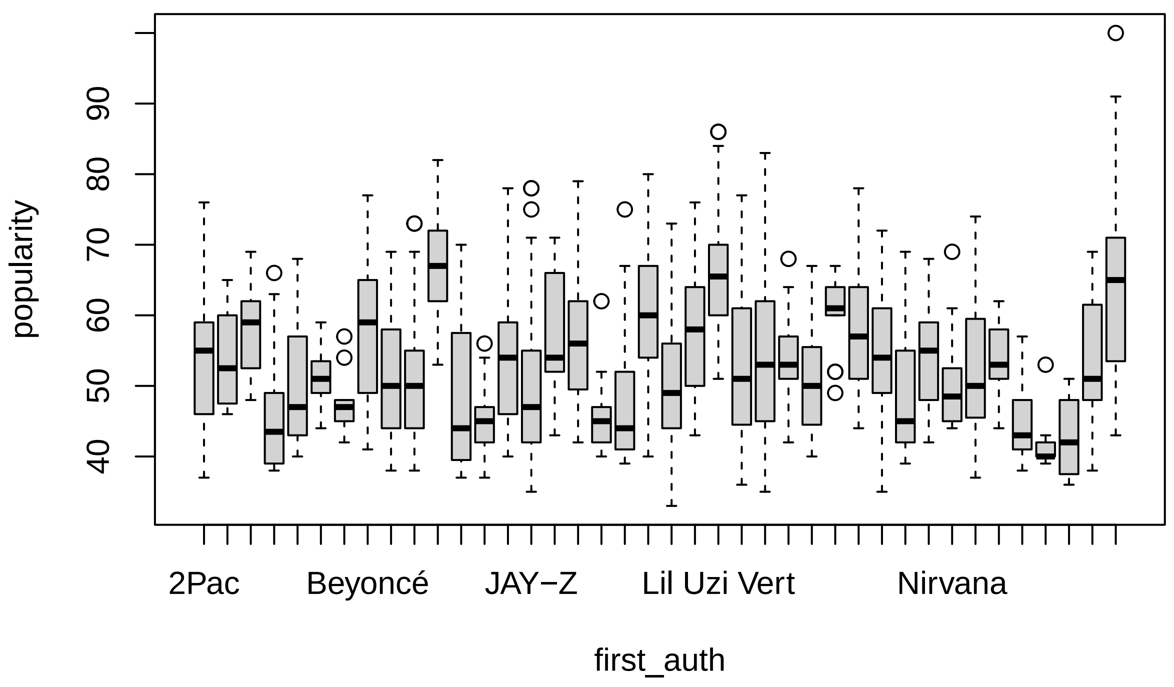
- n: number of song for each artist.

- pop: variable containing the original column popularity rescaled between 0 and 1.

```r
songs <- readRDS("spot (1).RDS")
head(songs)
```

```
## # A tibble: 6 x 18
##   acousticness danceability duration_ms energy explicit id      instrumentalness
##          <dbl>        <dbl>       <int>  <dbl>    <int> <chr>              <dbl>
## 1        0.116        0.908      203520  0.573        1 3iNzFu~         0.000547
## 2        0.299        0.703      247600  0.639        1 3eBBbA~         0
## 3       0.283          0.874      230227  0.64          1 1SgENh~       0.000002
## 4       0.0672         0.748      230493  0.614         1 36UjTT~       0.0000781
## 5       0.0634         0.536      267813  0.836         1 0lqKrm~       0
## 6       0.114          0.824      286893  0.847         1 06jYyA~       0.00000284
## # i 11 more variables: liveness <dbl>, loudness <dbl>, name <chr>,
## #   popularity <int>, release_date <chr>, speechiness <dbl>, tempo <dbl>,
## #   valence <dbl>, first_auth <chr>, n <int>, pop <dbl>
```

```
boxplot(popularity~first_auth,data=songs)
```



Point b

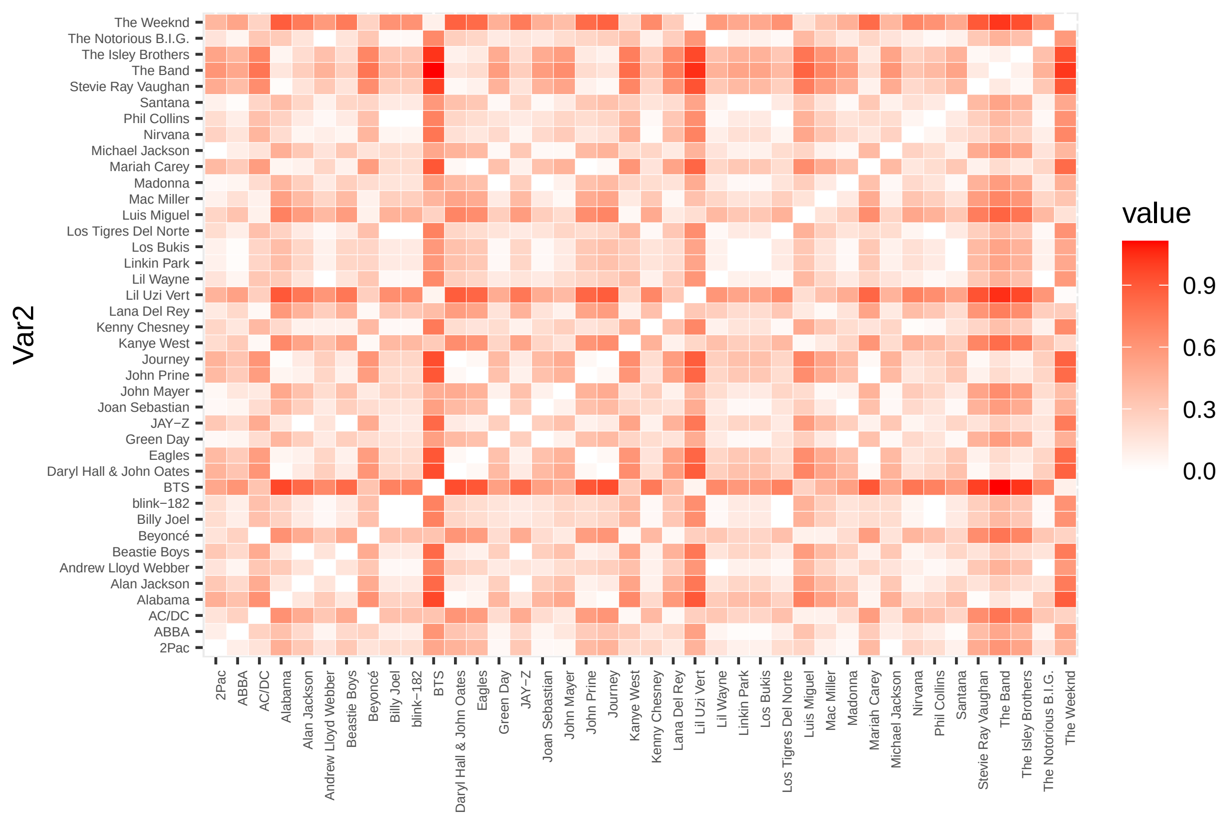
Focusing on "pop" and given two artists, we want to measure their different in popularity. Let X and Y denote the vectors of popularity measurements of the songs of two given artists. We measure their difference with:

$$T(X, Y) = |\text{median}(\text{logit}(X)) - \text{median}(\text{logit}(Y))|$$

After implementing this estimator to compare the popularity of AC/DC and Kanye West, we provide a matrix with all pairwise comparisons.

```
u_art<- unique(songs$first_auth)
```

```r
X <- songs$pop[songs$first_auth == "AC/DC"]

# Extract popularity measurements for Artist Y
Y <- songs$pop[songs$first_auth == "Kanye West"]

med_logit_X <-median( logit(X))
med_logit_Y <-median( logit(Y))

T_XY <- abs(med_logit_X-med_logit_Y)

pop_diff_T <- function(){
  res<-sapply(1:40, function(x) {
    sapply(1:40, function(y) {
      abs(median(logit(songs$pop[songs$first_auth == u_art[x]]))-median(logit(songs$pop[songs$first_aut
    })
  })
  return(res)
}

pop_diff<- pop_diff_T()

rownames(pop_diff) <- u_art
colnames(pop_diff) <- u_art

ggplot(melt(pop_diff),aes(Var1,Var2))+
  geom_tile(aes(fill = value), colour = "white")  +
  scale_fill_gradient(low = "white", high = "red")+
  theme(axis.text.x = element_text(angle=90,hjust = 1,siz=5),
        axis.text.y = element_text(hjust = 1,siz=5))
```



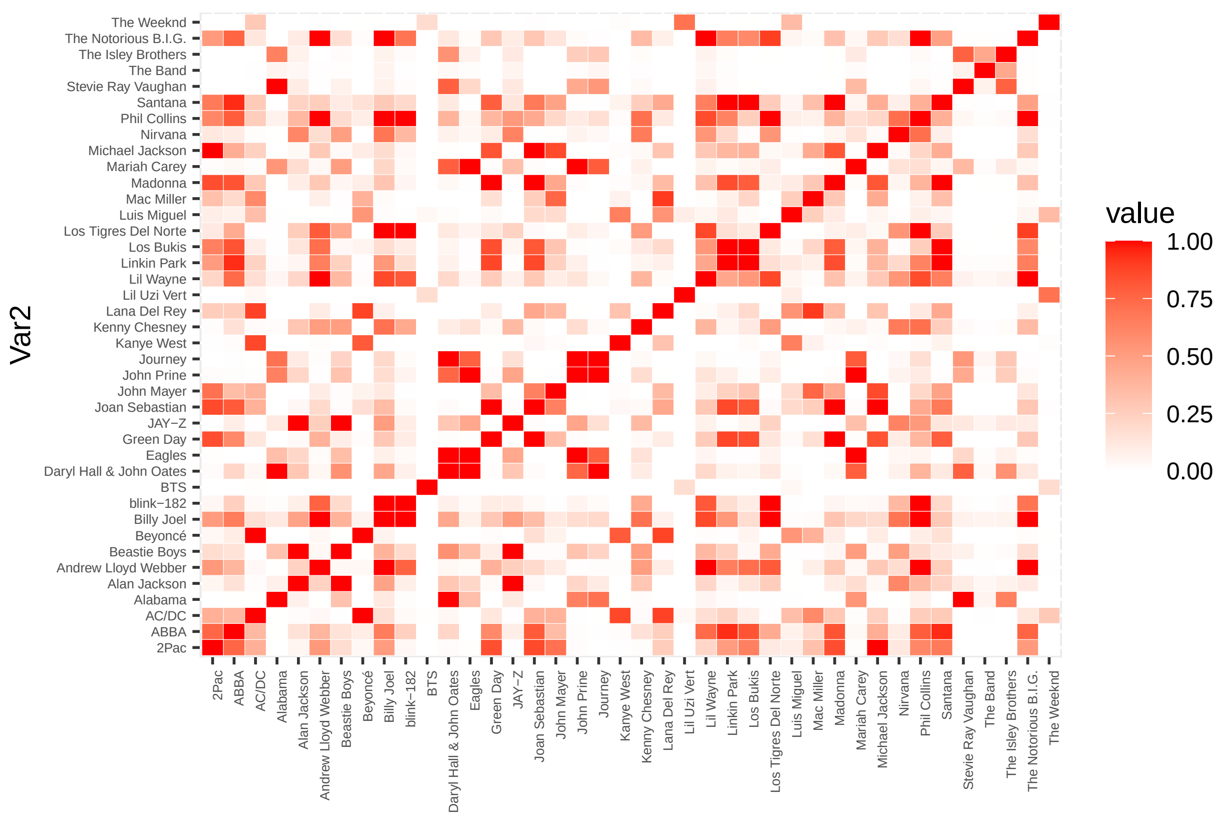
Point c

Now we perform a hypothesis testing to assess the absence of difference between the popularity of two artists measured via T(X,Y) and perform a sequence of pairwise permutation tests reporting the estimated p-values. The null hypothesis states that samples come from the same distribution i.e. the two playlist are similar. The alternative hypothesis stated that the samples are different, i.e. that the two playlist are not similar.

```r
data<-songs
p_values <- matrix(0,nrow = length(u_art),ncol = length(u_art))
for (i in 1:length(u_art)){
  for(j in 1:length(u_art)){
   if(i>=j){

      X <- data[data$first_auth ==  u_art[i],"pop"]
      Y <- data[data$first_auth ==  u_art[j],"pop"]
      observed_stat=abs(median(logit(songs$pop[songs$first_auth == u_art[i]]))-median(logit(songs$pop[s



      num_perm<- 1000
      permutation_stat<- numeric(num_perm)
      for (k in 1:num_perm){
        combined<- union_all(X,Y)
        combined$pop <- sample(combined$pop,nrow(combined),replace = F)
        perm_X<- combined$pop[1:length(X$pop)]
        perm_Y<-combined$pop[(length(X$pop)+1):(length(X$pop)+length(Y$pop))]
        permutation_stat[k]<- abs(median(logit(perm_X))-median(logit(perm_Y)))

      }

      p_value <- mean(permutation_stat>=observed_stat)

      p_values[i,j]<- p_value
      p_values[j,i]<- p_value


    }
  }
}


p_values
```

```
##        [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10] [,11] [,12]
##  [1,] 1.000 0.757 0.410 0.000 0.049 0.525 0.180 0.037 0.510 0.042 0.000 0.019
##  [2,] 0.757 1.000 0.370 0.002 0.154 0.380 0.150 0.095 0.656 0.249 0.000 0.213
##  [3,] 0.410 0.370 1.000 0.002 0.016 0.035 0.004 1.000 0.169 0.025 0.006 0.044
##  [4,] 0.000 0.002 0.002 1.000 0.077 0.002 0.316 0.000 0.116 0.013 0.000 0.999
##  [5,] 0.049 0.154 0.016 0.077 1.000 0.234 1.000 0.000 0.474 0.085 0.000 0.291
##  [6,] 0.525 0.380 0.035 0.002 0.234 1.000 0.037 0.037 1.000 0.769 0.000 0.044
##  [7,] 0.180 0.150 0.004 0.316 1.000 0.037 1.000 0.002 0.393 0.190 0.000 0.561
##  [8,] 0.037 0.095 1.000 0.000 0.000 0.037 0.002 1.000 0.061 0.000 0.000 0.000
##  [9,] 0.510 0.656 0.169 0.116 0.474 1.000 0.393 0.061 1.000 1.000 0.000 0.456
## [10,] 0.042 0.249 0.025 0.013 0.085 0.769 0.190 0.000 1.000 1.000 0.000 0.078
## [11,] 0.000 0.000 0.006 0.000 0.000 0.000 0.000 0.000 0.000 0.000 1.000 0.000
## [12,] 0.019 0.213 0.044 0.999 0.291 0.044 0.561 0.000 0.456 0.078 0.000 1.000
## [13,] 0.006 0.000 0.001 0.333 0.210 0.022 0.340 0.000 0.080 0.029 0.000 1.000
## [14,] 0.844 0.601 0.127 0.001 0.032 0.402 0.091 0.014 0.285 0.099 0.000 0.010
## [15,] 0.003 0.089 0.008 0.112 1.000 0.252 1.000 0.000 0.504 0.095 0.000 0.294
## [16,] 0.861 0.802 0.403 0.000 0.040 0.192 0.015 0.166 0.351 0.027 0.000 0.019
## [17,] 0.708 0.349 0.395 0.000 0.000 0.100 0.015 0.065 0.117 0.003 0.000 0.000
## [18,] 0.017 0.012 0.003 0.637 0.213 0.009 0.313 0.000 0.178 0.058 0.000 0.755
## [19,] 0.000 0.004 0.002 0.705 0.104 0.012 0.223 0.000 0.199 0.021 0.000 1.000
## [20,] 0.005 0.016 0.872 0.000 0.000 0.012 0.000 0.805 0.005 0.000 0.000 0.000
## [21,] 0.013 0.155 0.019 0.014 0.298 0.504 0.500 0.000 0.704 0.443 0.000 0.101
## [22,] 0.262 0.253 0.887 0.000 0.000 0.101 0.008 0.885 0.077 0.000 0.000 0.000
## [23,] 0.000 0.000 0.038 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.175 0.000
## [24,] 0.213 0.739 0.171 0.046 0.207 1.000 0.370 0.014 0.864 0.806 0.000 0.197
## [25,] 0.513 0.943 0.231 0.019 0.046 0.654 0.239 0.008 0.535 0.174 0.000 0.070
## [26,] 0.642 0.825 0.089 0.005 0.132 0.723 0.035 0.059 0.178 0.100 0.000 0.053
## [27,] 0.113 0.436 0.008 0.005 0.259 0.804 0.436 0.000 1.000 1.000 0.000 0.116
## [28,] 0.093 0.069 0.337 0.003 0.002 0.002 0.004 0.538 0.017 0.003 0.038 0.013
## [29,] 0.325 0.191 0.602 0.000 0.000 0.086 0.009 0.403 0.023 0.001 0.000 0.000
## [30,] 0.849 0.825 0.283 0.002 0.094 0.284 0.046 0.092 0.301 0.050 0.000 0.034
## [31,] 0.013 0.033 0.010 0.539 0.177 0.038 0.507 0.000 0.212 0.030 0.000 0.783
## [32,] 1.000 0.423 0.239 0.000 0.034 0.282 0.030 0.098 0.182 0.036 0.000 0.006
## [33,] 0.123 0.099 0.012 0.002 0.613 0.175 0.496 0.001 0.688 0.369 0.000 0.074
## [34,] 0.610 0.794 0.259 0.074 0.370 1.000 0.183 0.091 1.000 1.000 0.000 0.389
## [35,] 0.669 0.946 0.274 0.001 0.229 0.257 0.103 0.157 0.281 0.200 0.000 0.112
## [36,] 0.000 0.001 0.000 1.000 0.098 0.004 0.076 0.000 0.067 0.015 0.000 0.777
## [37,] 0.000 0.000 0.003 0.060 0.044 0.003 0.007 0.000 0.057 0.003 0.000 0.039
## [38,] 0.000 0.000 0.000 0.638 0.076 0.000 0.019 0.000 0.064 0.019 0.000 0.560
## [39,] 0.521 0.764 0.127 0.001 0.103 1.000 0.167 0.017 1.000 0.698 0.000 0.120
## [40,] 0.001 0.001 0.280 0.000 0.000 0.004 0.000 0.002 0.000 0.000 0.182 0.000
##       [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24]
##  [1,] 0.006 0.844 0.003 0.861 0.708 0.017 0.000 0.005 0.013 0.262 0.000 0.213
##  [2,] 0.000 0.601 0.089 0.802 0.349 0.012 0.004 0.016 0.155 0.253 0.000 0.739
##  [3,] 0.001 0.127 0.008 0.403 0.395 0.003 0.002 0.872 0.019 0.887 0.038 0.171
##  [4,] 0.333 0.001 0.112 0.000 0.000 0.637 0.705 0.000 0.014 0.000 0.000 0.046
##  [5,] 0.210 0.032 1.000 0.040 0.000 0.213 0.104 0.000 0.298 0.000 0.000 0.207
##  [6,] 0.022 0.402 0.252 0.192 0.100 0.009 0.012 0.012 0.504 0.101 0.000 1.000
##  [7,] 0.340 0.091 1.000 0.015 0.015 0.313 0.223 0.000 0.500 0.008 0.000 0.370
##  [8,] 0.000 0.014 0.000 0.166 0.065 0.000 0.000 0.805 0.000 0.885 0.000 0.014
##  [9,] 0.080 0.285 0.504 0.351 0.117 0.178 0.199 0.005 0.704 0.077 0.000 0.864
## [10,] 0.029 0.099 0.095 0.027 0.003 0.058 0.021 0.000 0.443 0.000 0.000 0.806
## [11,] 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.175 0.000
## [12,] 1.000 0.010 0.294 0.019 0.000 0.755 1.000 0.000 0.101 0.000 0.000 0.197
## [13,] 1.000 0.003 0.450 0.001 0.000 1.000 0.779 0.000 0.152 0.000 0.000 0.050
## [14,] 0.003 1.000 0.002 1.000 0.345 0.016 0.001 0.002 0.027 0.099 0.000 0.264
## [15,] 0.450 0.002 1.000 0.014 0.000 0.465 0.162 0.000 0.357 0.000 0.000 0.075
## [16,] 0.001 1.000 0.014 1.000 0.636 0.002 0.000 0.040 0.025 0.448 0.000 0.284
## [17,] 0.000 0.345 0.000 0.636 1.000 0.004 0.000 0.017 0.000 0.361 0.000 0.096
## [18,] 1.000 0.016 0.465 0.002 0.004 1.000 1.000 0.000 0.179 0.002 0.000 0.142
## [19,] 0.779 0.001 0.162 0.000 0.000 1.000 1.000 0.000 0.032 0.000 0.000 0.034
## [20,] 0.000 0.002 0.000 0.040 0.017 0.000 0.000 1.000 0.000 0.313 0.001 0.000
## [21,] 0.152 0.027 0.357 0.025 0.000 0.179 0.032 0.000 1.000 0.000 0.000 0.378
## [22,] 0.000 0.099 0.000 0.448 0.361 0.002 0.000 0.313 0.000 1.000 0.000 0.021
## [23,] 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.001 0.000 0.000 1.000 0.000
## [24,] 0.050 0.264 0.075 0.284 0.096 0.142 0.034 0.000 0.378 0.021 0.000 1.000
## [25,] 0.008 0.873 0.007 0.865 0.241 0.078 0.004 0.000 0.052 0.078 0.000 0.454
## [26,] 0.009 0.843 0.055 0.813 0.301 0.010 0.005 0.021 0.113 0.194 0.000 0.531
## [27,] 0.057 0.130 0.228 0.024 0.007 0.097 0.050 0.000 0.513 0.005 0.000 0.870
## [28,] 0.000 0.027 0.001 0.188 0.184 0.001 0.002 0.646 0.005 0.553 0.090 0.053
## [29,] 0.000 0.162 0.000 0.258 0.757 0.000 0.000 0.066 0.000 0.911 0.000 0.010
## [30,] 0.003 1.000 0.009 1.000 0.452 0.016 0.001 0.017 0.042 0.356 0.000 0.317
## [31,] 1.000 0.013 0.330 0.014 0.000 1.000 0.797 0.000 0.074 0.000 0.000 0.065
## [32,] 0.000 0.830 0.006 1.000 0.862 0.002 0.000 0.014 0.029 0.305 0.000 0.285
## [33,] 0.044 0.101 0.627 0.005 0.006 0.062 0.036 0.000 0.666 0.001 0.000 0.537
## [34,] 0.052 0.373 0.523 0.438 0.205 0.105 0.169 0.016 0.715 0.115 0.000 0.856
## [35,] 0.003 0.784 0.129 0.668 0.476 0.007 0.003 0.065 0.254 0.440 0.000 0.649
## [36,] 0.205 0.000 0.121 0.001 0.000 0.444 0.531 0.000 0.030 0.000 0.000 0.066
## [37,] 0.013 0.000 0.058 0.000 0.000 0.015 0.046 0.000 0.009 0.000 0.000 0.042
## [38,] 0.075 0.000 0.113 0.001 0.000 0.260 0.285 0.000 0.018 0.000 0.000 0.063
## [39,] 0.018 0.285 0.102 0.294 0.134 0.022 0.008 0.001 0.352 0.079 0.000 1.000
## [40,] 0.000 0.000 0.000 0.008 0.000 0.000 0.000 0.011 0.000 0.000 0.702 0.000
##       [,25] [,26] [,27] [,28] [,29] [,30] [,31] [,32] [,33] [,34] [,35] [,36]
##  [1,] 0.513 0.642 0.113 0.093 0.325 0.849 0.013 1.000 0.123 0.610 0.669 0.000
##  [2,] 0.943 0.825 0.436 0.069 0.191 0.825 0.033 0.423 0.099 0.794 0.946 0.001
##  [3,] 0.231 0.089 0.008 0.337 0.602 0.283 0.010 0.239 0.012 0.259 0.274 0.000
##  [4,] 0.019 0.005 0.005 0.003 0.000 0.002 0.539 0.000 0.002 0.074 0.001 1.000
##  [5,] 0.046 0.132 0.259 0.002 0.000 0.094 0.177 0.034 0.613 0.370 0.229 0.098
##  [6,] 0.654 0.723 0.804 0.002 0.086 0.284 0.038 0.282 0.175 1.000 0.257 0.004
##  [7,] 0.239 0.035 0.436 0.004 0.009 0.046 0.507 0.030 0.496 0.183 0.103 0.076
##  [8,] 0.008 0.059 0.000 0.538 0.403 0.092 0.000 0.098 0.001 0.091 0.157 0.000
##  [9,] 0.535 0.178 1.000 0.017 0.023 0.301 0.212 0.182 0.688 1.000 0.281 0.067
## [10,] 0.174 0.100 1.000 0.003 0.001 0.050 0.030 0.036 0.369 1.000 0.200 0.015
## [11,] 0.000 0.000 0.000 0.038 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
## [12,] 0.070 0.053 0.116 0.013 0.000 0.034 0.783 0.006 0.074 0.389 0.112 0.777
## [13,] 0.008 0.009 0.057 0.000 0.000 0.003 1.000 0.000 0.044 0.052 0.003 0.205
## [14,] 0.873 0.843 0.130 0.027 0.162 1.000 0.013 0.830 0.101 0.373 0.784 0.000
## [15,] 0.007 0.055 0.228 0.001 0.000 0.009 0.330 0.006 0.627 0.523 0.129 0.121
## [16,] 0.865 0.813 0.024 0.188 0.258 1.000 0.014 1.000 0.005 0.438 0.668 0.001
## [17,] 0.241 0.301 0.007 0.184 0.757 0.452 0.000 0.862 0.006 0.205 0.476 0.000
## [18,] 0.078 0.010 0.097 0.001 0.000 0.016 1.000 0.002 0.062 0.105 0.007 0.444
## [19,] 0.004 0.005 0.050 0.002 0.000 0.001 0.797 0.000 0.036 0.169 0.003 0.531
## [20,] 0.000 0.021 0.000 0.646 0.066 0.017 0.000 0.014 0.000 0.016 0.065 0.000
## [21,] 0.052 0.113 0.513 0.005 0.000 0.042 0.074 0.029 0.666 0.715 0.254 0.030
## [22,] 0.078 0.194 0.005 0.553 0.911 0.356 0.000 0.305 0.001 0.115 0.440 0.000
## [23,] 0.000 0.000 0.000 0.090 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
## [24,] 0.454 0.531 0.870 0.053 0.010 0.317 0.065 0.285 0.537 0.856 0.649 0.066
## [25,] 1.000 1.000 0.170 0.116 0.080 0.848 0.022 0.375 0.197 0.630 1.000 0.015
## [26,] 1.000 1.000 0.110 0.023 0.213 0.793 0.049 0.399 0.017 0.252 1.000 0.002
## [27,] 0.170 0.110 1.000 0.000 0.002 0.071 0.088 0.033 0.543 1.000 0.265 0.001
## [28,] 0.116 0.023 0.000 1.000 0.253 0.107 0.000 0.054 0.006 0.084 0.046 0.000
## [29,] 0.080 0.213 0.002 0.253 1.000 0.285 0.000 0.433 0.001 0.074 0.333 0.000
## [30,] 0.848 0.793 0.071 0.107 0.285 1.000 0.018 0.817 0.052 0.389 1.000 0.000
## [31,] 0.022 0.049 0.088 0.000 0.000 0.018 1.000 0.006 0.146 0.168 0.053 0.361
## [32,] 0.375 0.399 0.033 0.054 0.433 0.817 0.006 1.000 0.030 0.216 0.418 0.000
## [33,] 0.197 0.017 0.543 0.006 0.001 0.052 0.146 0.030 1.000 0.718 0.082 0.000
## [34,] 0.630 0.252 1.000 0.084 0.074 0.389 0.168 0.216 0.718 1.000 0.410 0.041
## [35,] 1.000 1.000 0.265 0.046 0.333 1.000 0.053 0.418 0.082 0.410 1.000 0.006
## [36,] 0.015 0.002 0.001 0.000 0.000 0.000 0.361 0.000 0.000 0.041 0.006 1.000
## [37,] 0.011 0.000 0.001 0.001 0.000 0.002 0.019 0.000 0.000 0.011 0.004 0.076
## [38,] 0.018 0.000 0.002 0.000 0.000 0.002 0.109 0.000 0.000 0.027 0.000 0.774
## [39,] 0.653 0.603 0.897 0.054 0.023 0.322 0.044 0.277 0.166 1.000 0.487 0.006
## [40,] 0.000 0.002 0.000 0.354 0.000 0.000 0.000 0.000 0.000 0.004 0.005 0.000
##       [,37] [,38] [,39] [,40]
##  [1,] 0.000 0.000 0.521 0.001
##  [2,] 0.000 0.000 0.764 0.001
##  [3,] 0.003 0.000 0.127 0.280
##  [4,] 0.060 0.638 0.001 0.000
##  [5,] 0.044 0.076 0.103 0.000
##  [6,] 0.003 0.000 1.000 0.004
##  [7,] 0.007 0.019 0.167 0.000
##  [8,] 0.000 0.000 0.017 0.002
##  [9,] 0.057 0.064 1.000 0.000
## [10,] 0.003 0.019 0.698 0.000
## [11,] 0.000 0.000 0.000 0.182
## [12,] 0.039 0.560 0.120 0.000
## [13,] 0.013 0.075 0.018 0.000
## [14,] 0.000 0.000 0.285 0.000
## [15,] 0.058 0.113 0.102 0.000
## [16,] 0.000 0.001 0.294 0.008
## [17,] 0.000 0.000 0.134 0.000
## [18,] 0.015 0.260 0.022 0.000
## [19,] 0.046 0.285 0.008 0.000
## [20,] 0.000 0.000 0.001 0.011
## [21,] 0.009 0.018 0.352 0.000
## [22,] 0.000 0.000 0.079 0.000
## [23,] 0.000 0.000 0.000 0.702
## [24,] 0.042 0.063 1.000 0.000
## [25,] 0.011 0.018 0.653 0.000
## [26,] 0.000 0.000 0.603 0.002
## [27,] 0.001 0.002 0.897 0.000
## [28,] 0.001 0.000 0.054 0.354
## [29,] 0.000 0.000 0.023 0.000
## [30,] 0.002 0.002 0.322 0.000
## [31,] 0.019 0.109 0.044 0.000
## [32,] 0.000 0.000 0.277 0.000
## [33,] 0.000 0.000 0.166 0.000
## [34,] 0.011 0.027 1.000 0.004
## [35,] 0.004 0.000 0.487 0.005
## [36,] 0.076 0.774 0.006 0.000
## [37,] 1.000 0.449 0.006 0.000
## [38,] 0.449 1.000 0.003 0.000
## [39,] 0.006 0.003 1.000 0.000
## [40,] 0.000 0.000 0.000 1.000
```

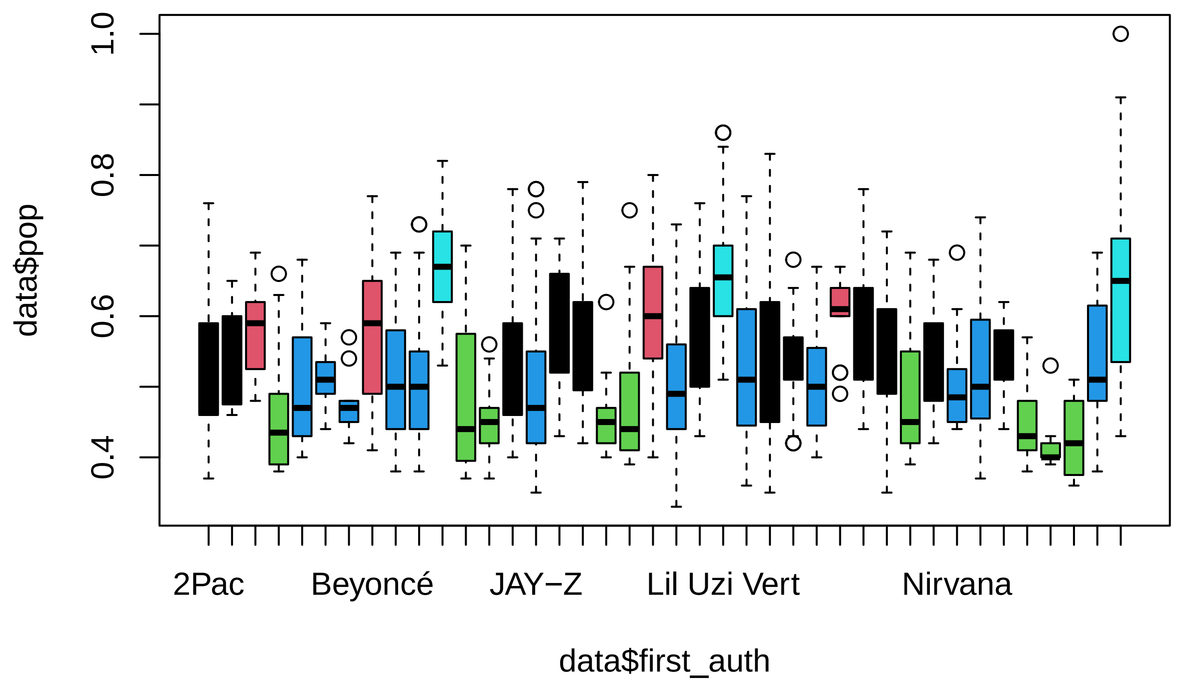
```
rownames(p_values) <- u_art
colnames(p_values) <- u_art

ggplot(melt(p_values),aes(Var1,Var2))+
  geom_tile(aes(fill = value), colour = "white")  +
  scale_fill_gradient(low = "white", high = "red")+
  theme(axis.text.x = element_text(angle=90,hjust = 1,siz=5),
        axis.text.y = element_text(hjust = 1,siz=5))
```



Point d

The cutree() cut a hierarchical clustering tree into a 5 clusters. The clustering is performed on a distance matrix derived from the p-value matrix, pval_mat, by subtracting each value from 1 and converting it to a distance using as.dist(). The resulting clusters are used to assign colors to the boxes in the boxplot.

```
boxplot( data$pop ~ data$first_auth,
         col= cutree( hclust(as.dist(1-p_values)),5))
```



Point e

Now, we apply the scale() function to "pop" and perform, for each artist, a single-sample t-test to assess:

$$H_0 : \mu = 0$$

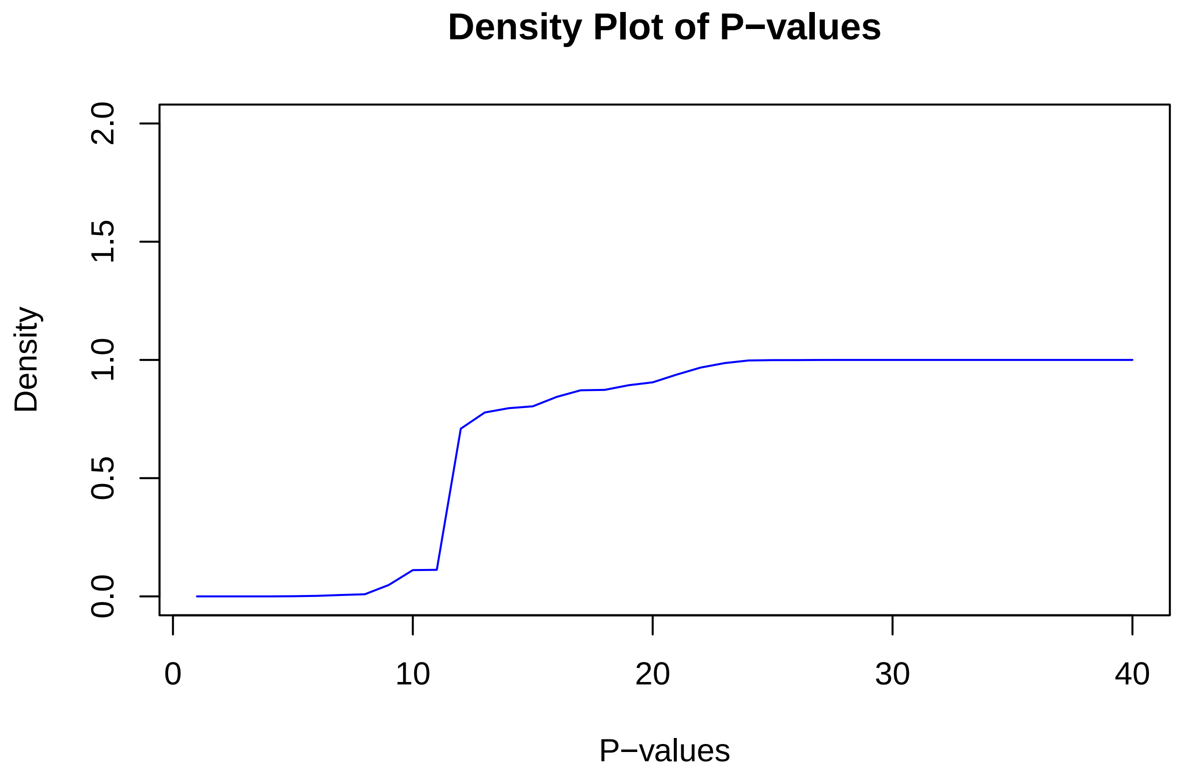
$$H_1 : \mu > 0$$

```r
pop_scaled <- scale(data$pop)
df<- data.frame(Artist = data$first_auth,pop_scaled)


t_test<- function(df,artist){
  x<- df[df$Artist==artist,"pop_scaled"]
  t_test<- t.test(x,mu=0,alternative = "greater")
  return(t_test$p.value)
}

p_values<- sapply(unique(df$Artist),t_test,df=df)

prob_violation <- sum(p_values<0.10)/length(p_values)
prob_violation #probability of the global null to be violated
```

```
## [1] 0.225
```

The plot below shows that the lies in the two extremes of probability i.e. 0 and 1. The ones with smaller values indicates a higher popularity index than the others.

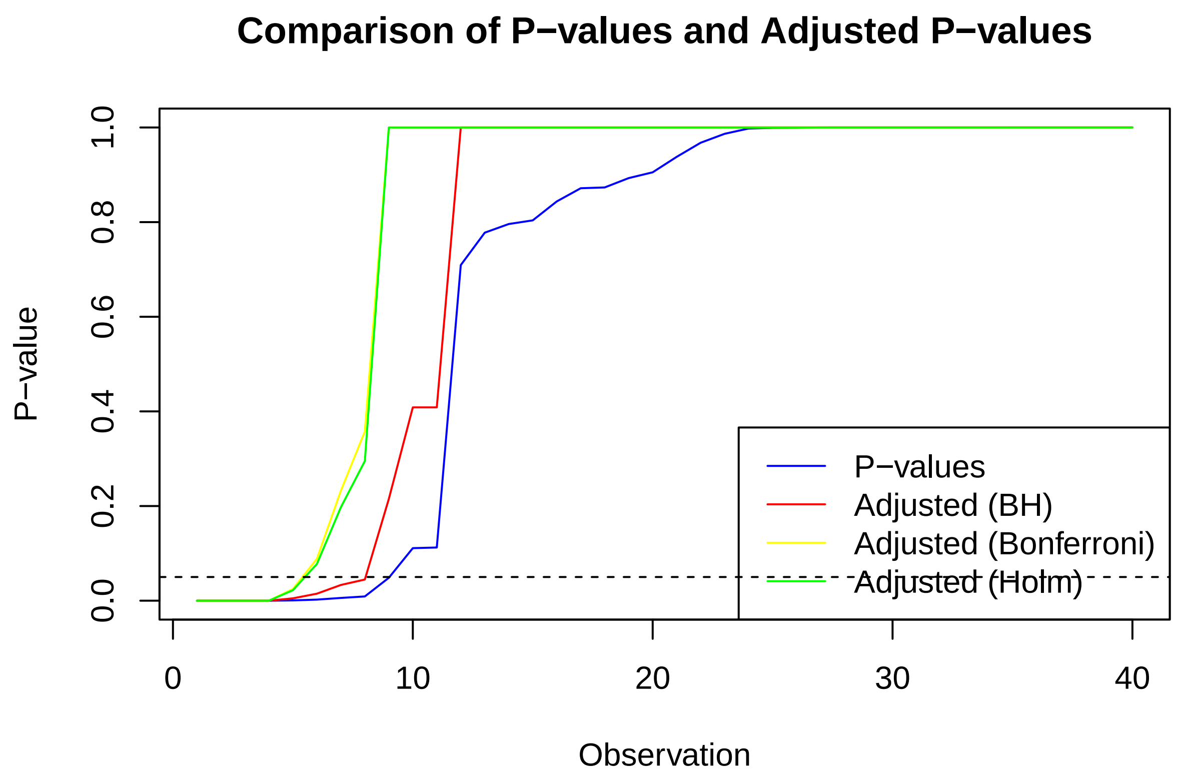
```
plot(sort(p_values), type="l",col = "blue", main = "Density Plot of P-values", xlab = "P-values", ylab =
```



Point f

The p.adjust function is used to adjust p-values for multiple comparisons. When conducting multiple statistical tests or comparing multiple groups, the probability of obtaining a significant result by chance increases. Multiple comparison procedures, such as adjusting p-values, are used to control for this increased risk of false positives (Type I errors).The p.adjust function takes a vector of p-values as input and applies various methods to adjust these p-values: Bonferroni adjustment,Benjamini-Hochberg and Holm. In summary, the p.adjust function helps in minimizing Type I errors by applying appropriate adjustments to p-values, depending on the chosen method.

```
adjusted_bh <- p.adjust(p_values, method = "BH")
adjusted_bonferroni <- p.adjust(p_values, method = "bonferroni")
adjusted_holm <- p.adjust(p_values, method = "holm")
results <- data.frame(p_values, adjusted_bh, adjusted_bonferroni,adjusted_holm)

# Plotting the p-values and adjusted p-values
plot(sort(p_values), type="l", col = "blue", xlab = "Observation", ylab = "P-value",
     main = "Comparison of P-values and Adjusted P-values")
lines(sort(adjusted_bh), col = "red")
lines(sort(adjusted_bonferroni),  col = "yellow")
lines(sort(adjusted_holm), col = "green")
abline(h = 0.05, lwd = 1, lty = 2)
legend("bottomright", legend = c("P-values", "Adjusted (BH)", "Adjusted (Bonferroni)","Adjusted (Holm)"
       col = c("blue", "red","yellow","green"), lwd=1,)
```



Point g

Holm's procedure

When the n hypotheses are tested seperately using tests with the level

$$\frac{\alpha}{n}$$

it follows immediately from the Boole inequality that the probability of rejecting any true hypothesis is smaller then or equal to alpha. This constitutes a multiple test procedure with the multiple level of signficance alpha for free combination, the classical Bonferroni mupltiple test procedure.

As written in the paper "A simple sequentially rejective multiple test procedure" of Sture Holm, the sequentially rejective Bonferroni test the obtained levels are compared to the numbers

$$\frac{\alpha}{n}, \frac{\alpha}{n-1}, \ldots, \frac{\alpha}{1}$$

, whereas in the classical Bonferroni test they are compared to

$$\frac{\alpha}{n}$$

. This means that the probability of rejecting any set of (false) hypotheses using the CLASSICAL Bonferroni test is SMALLER than or equal to the same probability using the sequantially rejective Bonferroni test based on the same test statistics. So the classical Bonferroni test can be replaced by the corresponding sequentially rejective Bonferroni test without loosing any probability of rejecting false hypotheses. The power gain obtained by using a sequentially rejective Bonferroni test dependes very much upon the alternative. The power of a hypothesis test is the probability of rejecting the null hypothesis when the alternative hypothesis is the hypothesis that is true. As a consequence, the power gain is small if all the hypothese are "almost true". It may be considerable if a number of hypotheses are "completely wrong".