

ANUJATESTASS2

Anuja Saira Abraham

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
```

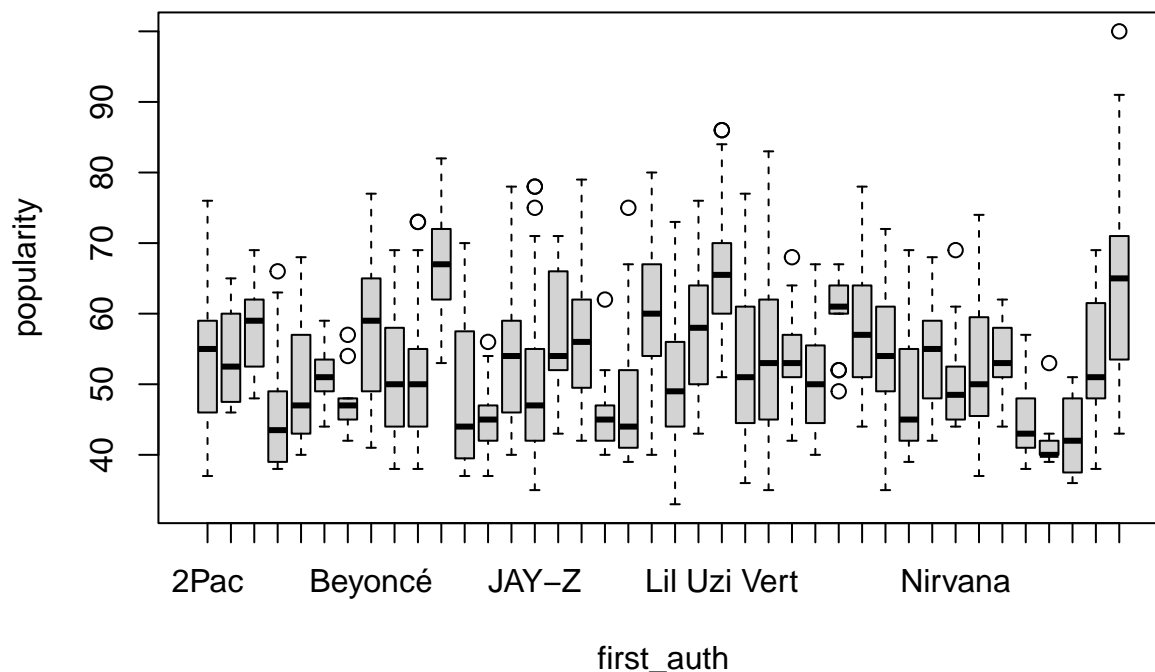
DESCRIPTION OF THE VARIABLES AND EXPLORATORY ANALYSIS: Considering the data saved in spot.RDS, a description of the variables can be found on Kaggle: - Acousticness: a confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic. - Danceability: 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.

```
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)
```



Focusing on “pop” and given two artists, we want to measure their difference 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)

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_auth == u_art[y]]))
    })
  })
}
```

```

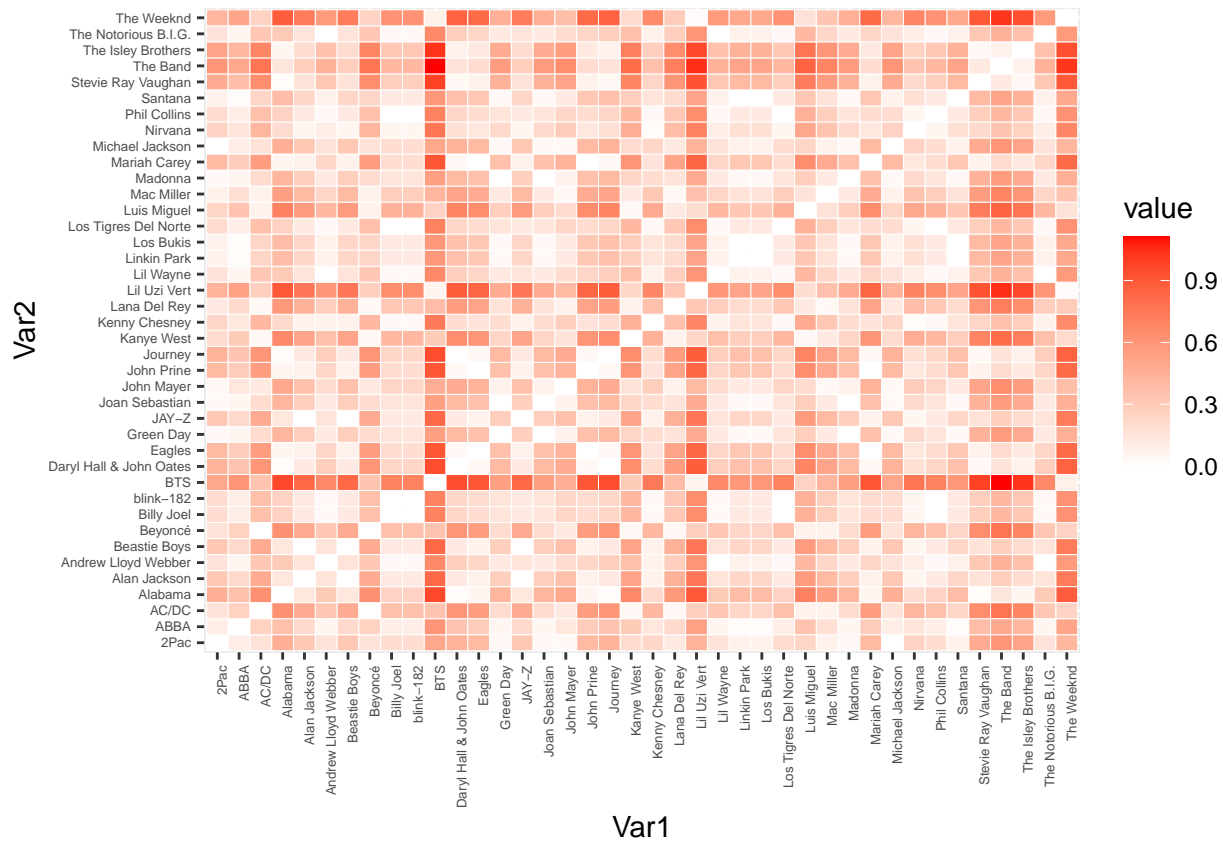
    })
  })
  return(res)
}

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))

```



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.

```

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 0.762 0.426 0.001 0.054 0.552 0.170 0.043 0.481 0.044 0.000 0.019
## [2,] 0.762 1.000 0.369 0.003 0.128 0.396 0.150 0.087 0.641 0.241 0.000 0.202
## [3,] 0.426 0.369 1.000 0.001 0.017 0.045 0.004 1.000 0.160 0.027 0.005 0.041
## [4,] 0.001 0.003 0.001 1.000 0.093 0.001 0.311 0.000 0.104 0.010 0.000 0.999
## [5,] 0.054 0.128 0.017 0.093 1.000 0.196 1.000 0.000 0.497 0.075 0.000 0.285
## [6,] 0.552 0.396 0.045 0.001 0.196 1.000 0.035 0.026 1.000 0.748 0.000 0.038
## [7,] 0.170 0.150 0.004 0.311 1.000 0.035 1.000 0.002 0.390 0.197 0.000 0.529
## [8,] 0.043 0.087 1.000 0.000 0.000 0.026 0.002 1.000 0.062 0.000 0.000 0.000
## [9,] 0.481 0.641 0.160 0.104 0.497 1.000 0.390 0.062 1.000 1.000 0.000 0.473
## [10,] 0.044 0.241 0.027 0.010 0.075 0.748 0.197 0.000 1.000 1.000 0.000 0.062
## [11,] 0.000 0.000 0.005 0.000 0.000 0.000 0.000 0.000 0.000 0.000 1.000 0.000
## [12,] 0.019 0.202 0.041 0.999 0.285 0.038 0.529 0.000 0.473 0.062 0.000 1.000
## [13,] 0.001 0.000 0.004 0.298 0.208 0.020 0.353 0.000 0.085 0.030 0.000 1.000
## [14,] 0.844 0.617 0.118 0.000 0.038 0.423 0.090 0.014 0.302 0.104 0.000 0.008
## [15,] 0.002 0.075 0.003 0.126 1.000 0.223 1.000 0.000 0.471 0.096 0.000 0.315
## [16,] 0.863 0.832 0.389 0.000 0.035 0.204 0.011 0.161 0.336 0.037 0.000 0.020
## [17,] 0.726 0.364 0.415 0.000 0.001 0.109 0.018 0.078 0.112 0.001 0.000 0.000
## [18,] 0.018 0.007 0.004 0.657 0.233 0.006 0.297 0.000 0.192 0.070 0.000 0.729
## [19,] 0.002 0.002 0.003 0.703 0.132 0.010 0.218 0.000 0.188 0.013 0.000 1.000
## [20,] 0.005 0.017 0.879 0.000 0.000 0.013 0.002 0.801 0.008 0.000 0.000 0.000
## [21,] 0.012 0.147 0.012 0.033 0.315 0.509 0.468 0.000 0.705 0.448 0.000 0.123
## [22,] 0.299 0.263 0.877 0.000 0.000 0.084 0.001 0.882 0.083 0.001 0.000 0.000
## [23,] 0.000 0.000 0.035 0.000 0.000 0.000 0.000 0.001 0.000 0.000 0.177 0.000

```

```

## [24,] 0.222 0.735 0.153 0.056 0.208 1.000 0.399 0.012 0.860 0.806 0.000 0.174
## [25,] 0.529 0.947 0.229 0.016 0.055 0.669 0.246 0.008 0.538 0.168 0.000 0.076
## [26,] 0.621 0.851 0.087 0.007 0.134 0.742 0.040 0.051 0.185 0.099 0.000 0.048
## [27,] 0.081 0.396 0.011 0.002 0.288 0.818 0.445 0.001 1.000 1.000 0.000 0.116
## [28,] 0.122 0.058 0.308 0.001 0.002 0.001 0.002 0.531 0.024 0.005 0.038 0.015
## [29,] 0.342 0.168 0.575 0.000 0.000 0.094 0.005 0.437 0.027 0.000 0.000 0.000
## [30,] 0.865 0.783 0.289 0.001 0.062 0.283 0.060 0.113 0.285 0.052 0.000 0.037
## [31,] 0.012 0.036 0.010 0.485 0.151 0.053 0.542 0.000 0.199 0.022 0.000 0.778
## [32,] 1.000 0.429 0.256 0.000 0.020 0.257 0.024 0.095 0.215 0.022 0.000 0.002
## [33,] 0.136 0.110 0.010 0.001 0.596 0.194 0.469 0.001 0.710 0.376 0.000 0.073
## [34,] 0.573 0.801 0.242 0.073 0.374 1.000 0.147 0.094 1.000 1.000 0.000 0.357
## [35,] 0.651 0.944 0.257 0.000 0.221 0.245 0.108 0.131 0.302 0.206 0.000 0.111
## [36,] 0.000 0.000 0.000 1.000 0.092 0.002 0.099 0.000 0.067 0.009 0.000 0.770
## [37,] 0.000 0.001 0.004 0.036 0.043 0.001 0.004 0.000 0.050 0.001 0.000 0.029
## [38,] 0.001 0.000 0.001 0.651 0.067 0.001 0.026 0.000 0.076 0.021 0.000 0.562
## [39,] 0.465 0.737 0.124 0.002 0.119 1.000 0.153 0.032 1.000 0.703 0.000 0.117
## [40,] 0.000 0.002 0.283 0.000 0.000 0.001 0.000 0.003 0.000 0.000 0.188 0.000
##      [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24]
## [1,] 0.001 0.844 0.002 0.863 0.726 0.018 0.002 0.005 0.012 0.299 0.000 0.222
## [2,] 0.000 0.617 0.075 0.832 0.364 0.007 0.002 0.017 0.147 0.263 0.000 0.735
## [3,] 0.004 0.118 0.003 0.389 0.415 0.004 0.003 0.879 0.012 0.877 0.035 0.153
## [4,] 0.298 0.000 0.126 0.000 0.000 0.657 0.703 0.000 0.033 0.000 0.000 0.056
## [5,] 0.208 0.038 1.000 0.035 0.001 0.233 0.132 0.000 0.315 0.000 0.000 0.208
## [6,] 0.020 0.423 0.223 0.204 0.109 0.006 0.010 0.013 0.509 0.084 0.000 1.000
## [7,] 0.353 0.090 1.000 0.011 0.018 0.297 0.218 0.002 0.468 0.001 0.000 0.399
## [8,] 0.000 0.014 0.000 0.161 0.078 0.000 0.000 0.801 0.000 0.882 0.001 0.012
## [9,] 0.085 0.302 0.471 0.336 0.112 0.192 0.188 0.008 0.705 0.083 0.000 0.860
## [10,] 0.030 0.104 0.096 0.037 0.001 0.070 0.013 0.000 0.448 0.001 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 0.177 0.000
## [12,] 1.000 0.008 0.315 0.020 0.000 0.729 1.000 0.000 0.123 0.000 0.000 0.174
## [13,] 1.000 0.001 0.459 0.001 0.000 1.000 0.807 0.000 0.146 0.000 0.000 0.064
## [14,] 0.001 1.000 0.003 1.000 0.320 0.011 0.000 0.000 0.029 0.087 0.000 0.293
## [15,] 0.459 0.003 1.000 0.024 0.000 0.448 0.201 0.000 0.359 0.000 0.000 0.073
## [16,] 0.001 1.000 0.024 1.000 0.607 0.004 0.001 0.044 0.033 0.441 0.000 0.275
## [17,] 0.000 0.320 0.000 0.607 1.000 0.002 0.000 0.016 0.001 0.392 0.000 0.092
## [18,] 1.000 0.011 0.448 0.004 0.002 1.000 1.000 0.000 0.187 0.001 0.000 0.183
## [19,] 0.807 0.000 0.201 0.001 0.000 1.000 1.000 0.000 0.023 0.000 0.000 0.028
## [20,] 0.000 0.000 0.000 0.044 0.016 0.000 0.000 1.000 0.000 0.322 0.003 0.000
## [21,] 0.146 0.029 0.359 0.033 0.001 0.187 0.023 0.000 1.000 0.000 0.000 0.384
## [22,] 0.000 0.087 0.000 0.441 0.392 0.001 0.000 0.322 0.000 1.000 0.000 0.019
## [23,] 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.003 0.000 0.000 1.000 0.000
## [24,] 0.064 0.293 0.073 0.275 0.092 0.183 0.028 0.000 0.384 0.019 0.000 1.000
## [25,] 0.010 0.882 0.008 0.854 0.222 0.084 0.004 0.000 0.046 0.080 0.000 0.417
## [26,] 0.009 0.822 0.064 0.782 0.240 0.009 0.006 0.020 0.122 0.220 0.000 0.548
## [27,] 0.075 0.143 0.244 0.017 0.002 0.095 0.051 0.000 0.530 0.006 0.000 0.862
## [28,] 0.000 0.030 0.000 0.169 0.185 0.000 0.001 0.627 0.001 0.581 0.100 0.035
## [29,] 0.000 0.150 0.000 0.275 0.786 0.000 0.000 0.061 0.000 0.900 0.000 0.004
## [30,] 0.004 1.000 0.014 1.000 0.446 0.022 0.002 0.021 0.041 0.327 0.000 0.337
## [31,] 1.000 0.016 0.368 0.014 0.000 1.000 0.785 0.000 0.093 0.000 0.000 0.074
## [32,] 0.001 0.805 0.002 1.000 0.880 0.007 0.000 0.024 0.013 0.297 0.000 0.272
## [33,] 0.046 0.099 0.636 0.010 0.011 0.056 0.037 0.000 0.671 0.006 0.000 0.524
## [34,] 0.047 0.325 0.489 0.410 0.186 0.088 0.144 0.016 0.715 0.105 0.001 0.883
## [35,] 0.003 0.776 0.117 0.694 0.486 0.008 0.005 0.067 0.232 0.417 0.000 0.679
## [36,] 0.224 0.001 0.128 0.000 0.000 0.445 0.511 0.000 0.015 0.001 0.000 0.073

```

```

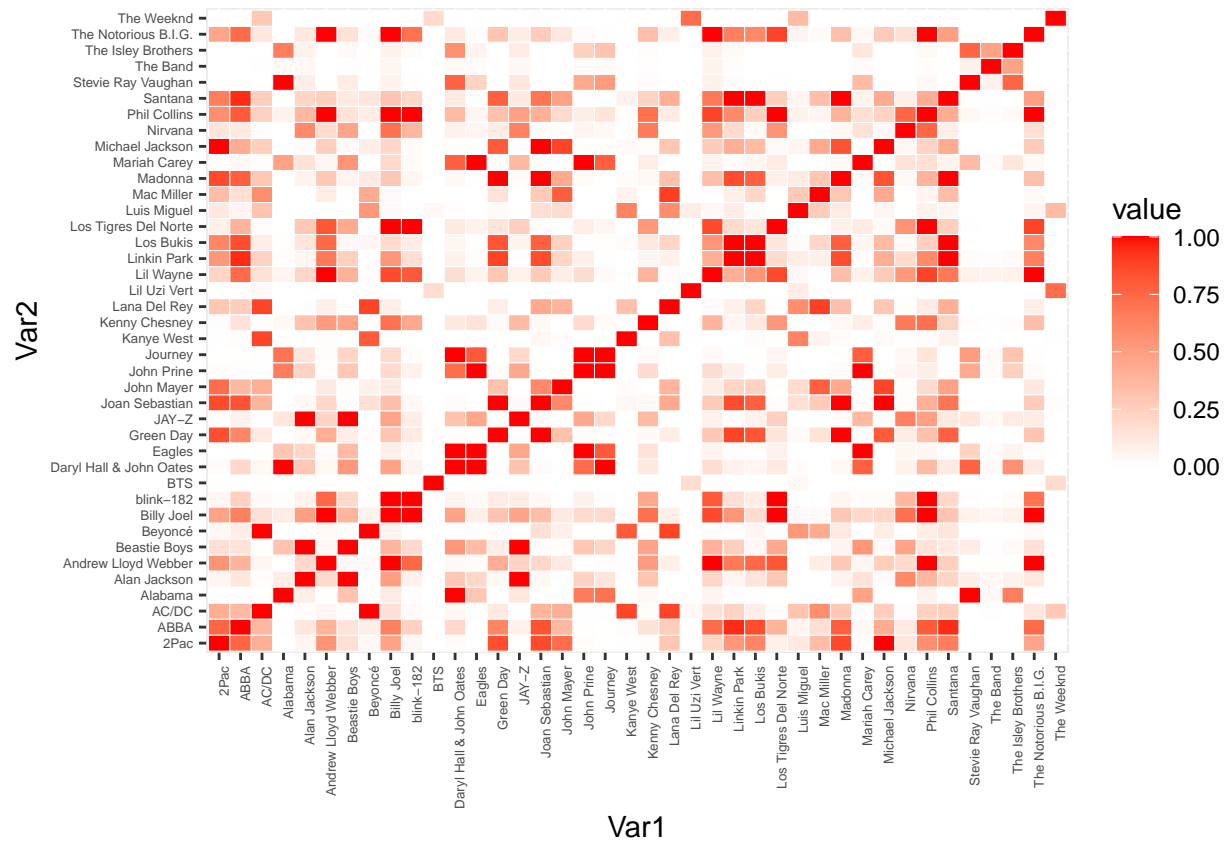
## [37,] 0.019 0.000 0.039 0.000 0.000 0.021 0.027 0.000 0.008 0.000 0.000 0.064
## [38,] 0.060 0.000 0.106 0.001 0.000 0.247 0.308 0.000 0.012 0.000 0.000 0.068
## [39,] 0.015 0.302 0.101 0.270 0.119 0.020 0.007 0.000 0.337 0.086 0.000 1.000
## [40,] 0.000 0.000 0.000 0.010 0.000 0.000 0.000 0.008 0.000 0.002 0.727 0.000
##      [,25] [,26] [,27] [,28] [,29] [,30] [,31] [,32] [,33] [,34] [,35] [,36]
## [1,] 0.529 0.621 0.081 0.122 0.342 0.865 0.012 1.000 0.136 0.573 0.651 0.000
## [2,] 0.947 0.851 0.396 0.058 0.168 0.783 0.036 0.429 0.110 0.801 0.944 0.000
## [3,] 0.229 0.087 0.011 0.308 0.575 0.289 0.010 0.256 0.010 0.242 0.257 0.000
## [4,] 0.016 0.007 0.002 0.001 0.000 0.001 0.485 0.000 0.001 0.073 0.000 1.000
## [5,] 0.055 0.134 0.288 0.002 0.000 0.062 0.151 0.020 0.596 0.374 0.221 0.092
## [6,] 0.669 0.742 0.818 0.001 0.094 0.283 0.053 0.257 0.194 1.000 0.245 0.002
## [7,] 0.246 0.040 0.445 0.002 0.005 0.060 0.542 0.024 0.469 0.147 0.108 0.099
## [8,] 0.008 0.051 0.001 0.531 0.437 0.113 0.000 0.095 0.001 0.094 0.131 0.000
## [9,] 0.538 0.185 1.000 0.024 0.027 0.285 0.199 0.215 0.710 1.000 0.302 0.067
## [10,] 0.168 0.099 1.000 0.005 0.000 0.052 0.022 0.022 0.376 1.000 0.206 0.009
## [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,] 0.076 0.048 0.116 0.015 0.000 0.037 0.778 0.002 0.073 0.357 0.111 0.770
## [13,] 0.010 0.009 0.075 0.000 0.000 0.004 1.000 0.001 0.046 0.047 0.003 0.224
## [14,] 0.882 0.822 0.143 0.030 0.150 1.000 0.016 0.805 0.099 0.325 0.776 0.001
## [15,] 0.008 0.064 0.244 0.000 0.000 0.014 0.368 0.002 0.636 0.489 0.117 0.128
## [16,] 0.854 0.782 0.017 0.169 0.275 1.000 0.014 1.000 0.010 0.410 0.694 0.000
## [17,] 0.222 0.240 0.002 0.185 0.786 0.446 0.000 0.880 0.011 0.186 0.486 0.000
## [18,] 0.084 0.009 0.095 0.000 0.000 0.022 1.000 0.007 0.056 0.088 0.008 0.445
## [19,] 0.004 0.006 0.051 0.001 0.000 0.002 0.785 0.000 0.037 0.144 0.005 0.511
## [20,] 0.000 0.020 0.000 0.627 0.061 0.021 0.000 0.024 0.000 0.016 0.067 0.000
## [21,] 0.046 0.122 0.530 0.001 0.000 0.041 0.093 0.013 0.671 0.715 0.232 0.015
## [22,] 0.080 0.220 0.006 0.581 0.900 0.327 0.000 0.297 0.006 0.105 0.417 0.001
## [23,] 0.000 0.000 0.000 0.100 0.000 0.000 0.000 0.000 0.000 0.001 0.000 0.000
## [24,] 0.417 0.548 0.862 0.035 0.004 0.337 0.074 0.272 0.524 0.883 0.679 0.073
## [25,] 1.000 1.000 0.190 0.097 0.083 0.852 0.017 0.422 0.190 0.597 1.000 0.019
## [26,] 1.000 1.000 0.131 0.021 0.216 0.784 0.048 0.356 0.030 0.247 1.000 0.002
## [27,] 0.190 0.131 1.000 0.000 0.002 0.077 0.106 0.028 0.551 1.000 0.253 0.005
## [28,] 0.097 0.021 0.000 1.000 0.273 0.105 0.001 0.045 0.000 0.072 0.049 0.000
## [29,] 0.083 0.216 0.002 0.273 1.000 0.295 0.000 0.440 0.002 0.056 0.337 0.000
## [30,] 0.852 0.784 0.077 0.105 0.295 1.000 0.030 0.826 0.043 0.395 1.000 0.000
## [31,] 0.017 0.048 0.106 0.001 0.000 0.030 1.000 0.002 0.145 0.168 0.064 0.354
## [32,] 0.422 0.356 0.028 0.045 0.440 0.826 0.002 1.000 0.043 0.234 0.436 0.000
## [33,] 0.190 0.030 0.551 0.000 0.002 0.043 0.145 0.043 1.000 0.757 0.083 0.000
## [34,] 0.597 0.247 1.000 0.072 0.056 0.395 0.168 0.234 0.757 1.000 0.432 0.039
## [35,] 1.000 1.000 0.253 0.049 0.337 1.000 0.064 0.436 0.083 0.432 1.000 0.003
## [36,] 0.019 0.002 0.005 0.000 0.000 0.000 0.354 0.000 0.000 0.039 0.003 1.000
## [37,] 0.011 0.000 0.001 0.002 0.000 0.000 0.019 0.000 0.001 0.010 0.004 0.069
## [38,] 0.028 0.000 0.001 0.000 0.000 0.000 0.128 0.000 0.000 0.029 0.000 0.765
## [39,] 0.644 0.604 0.883 0.059 0.031 0.330 0.034 0.273 0.166 1.000 0.495 0.002
## [40,] 0.000 0.002 0.000 0.352 0.000 0.001 0.000 0.001 0.000 0.001 0.009 0.000
##      [,37] [,38] [,39] [,40]
## [1,] 0.000 0.001 0.465 0.000
## [2,] 0.001 0.000 0.737 0.002
## [3,] 0.004 0.001 0.124 0.283
## [4,] 0.036 0.651 0.002 0.000
## [5,] 0.043 0.067 0.119 0.000
## [6,] 0.001 0.001 1.000 0.001
## [7,] 0.004 0.026 0.153 0.000
## [8,] 0.000 0.000 0.032 0.003

```

```
## [9,] 0.050 0.076 1.000 0.000
## [10,] 0.001 0.021 0.703 0.000
## [11,] 0.000 0.000 0.000 0.188
## [12,] 0.029 0.562 0.117 0.000
## [13,] 0.019 0.060 0.015 0.000
## [14,] 0.000 0.000 0.302 0.000
## [15,] 0.039 0.106 0.101 0.000
## [16,] 0.000 0.001 0.270 0.010
## [17,] 0.000 0.000 0.119 0.000
## [18,] 0.021 0.247 0.020 0.000
## [19,] 0.027 0.308 0.007 0.000
## [20,] 0.000 0.000 0.000 0.008
## [21,] 0.008 0.012 0.337 0.000
## [22,] 0.000 0.000 0.086 0.002
## [23,] 0.000 0.000 0.000 0.727
## [24,] 0.064 0.068 1.000 0.000
## [25,] 0.011 0.028 0.644 0.000
## [26,] 0.000 0.000 0.604 0.002
## [27,] 0.001 0.001 0.883 0.000
## [28,] 0.002 0.000 0.059 0.352
## [29,] 0.000 0.000 0.031 0.000
## [30,] 0.000 0.000 0.330 0.001
## [31,] 0.019 0.128 0.034 0.000
## [32,] 0.000 0.000 0.273 0.001
## [33,] 0.001 0.000 0.166 0.000
## [34,] 0.010 0.029 1.000 0.001
## [35,] 0.004 0.000 0.495 0.009
## [36,] 0.069 0.765 0.002 0.000
## [37,] 1.000 0.477 0.002 0.000
## [38,] 0.477 1.000 0.001 0.000
## [39,] 0.002 0.001 1.000 0.000
## [40,] 0.000 0.000 0.000 1.000
```

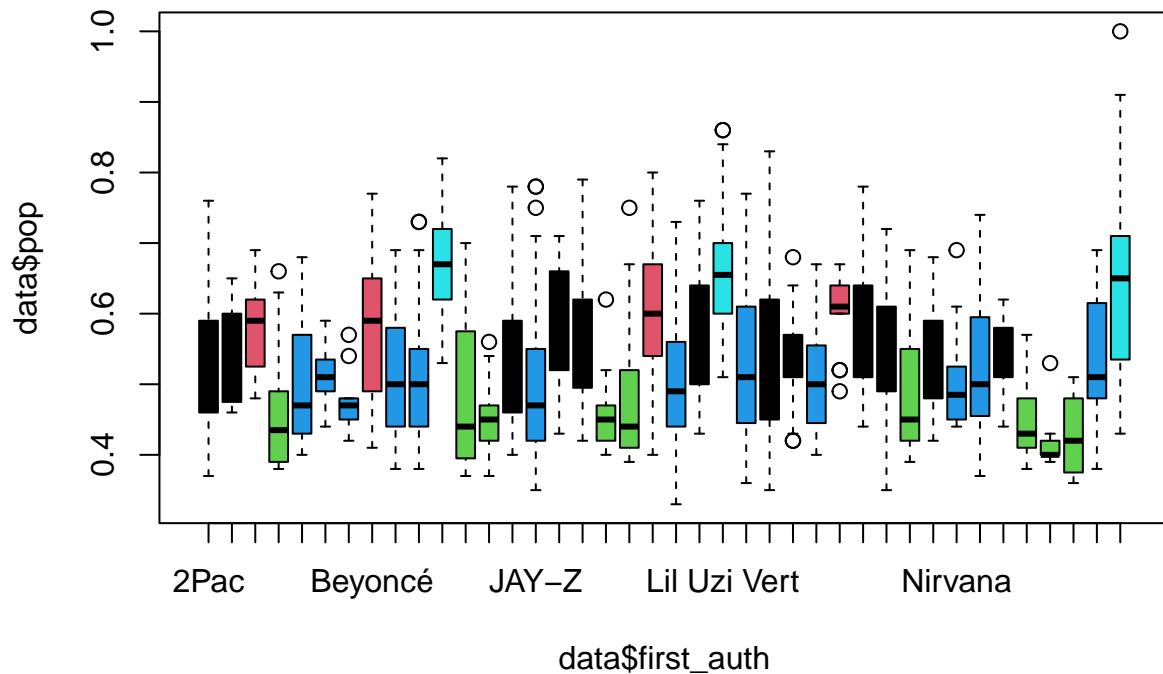
```
rownames(p_values) <- u_art
colnames(p_values) <- u_art

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))
```

??

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



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

$$H_0 : \mu = 0$$

$$H_1 : \mu > 0$$

```
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)
}

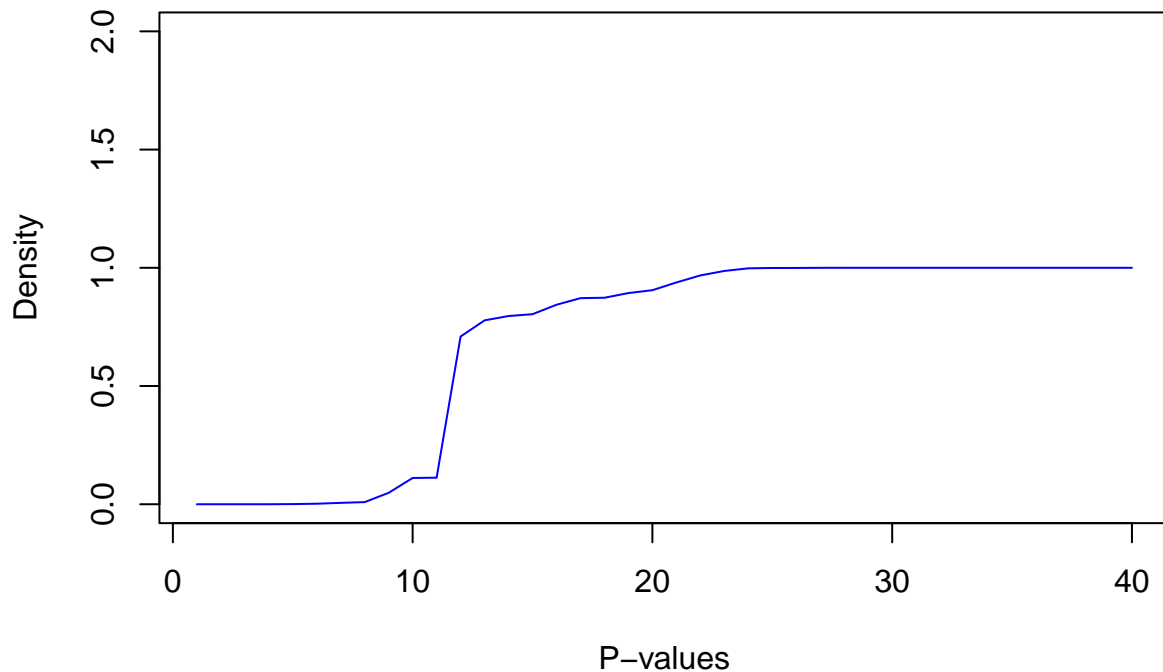
p_values<- sapply(unique(df$Artist),t_test,df=df)

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

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

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

Density Plot of P-values

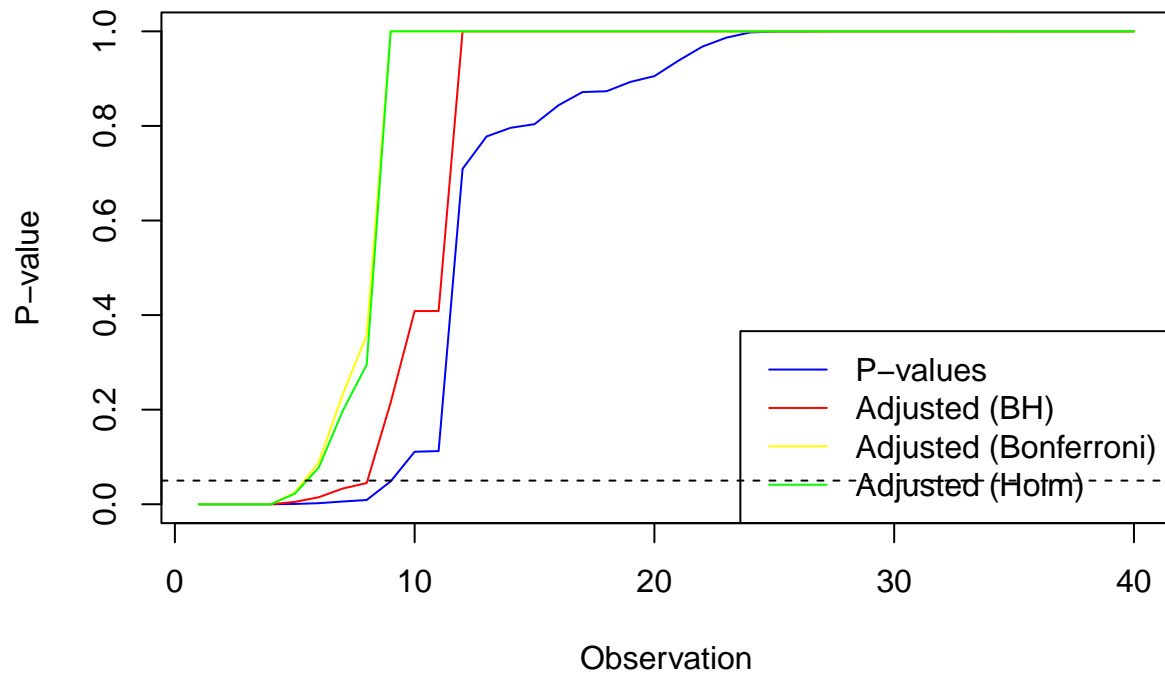


The `p.adjust` function is used to adjust p-values for multiple comparisons. When conducting multiple statistical tests or comparing multiple groups, the probability of obtaining a significant result by chance increases. Multiple comparison procedures, such as adjusting p-values, are used to control 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,)
```

Comparison of P-values and Adjusted P-values



Holm procedure (will do is soon!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!)