Project 1 by Ana Cardenas

To me, music is an essential part of my everyday life. Music offers an opportunity to escape from the stress that accompanies my academic schedule and workload. Spotify has been my music source for years, and I knew the data collected from my Spotify would be an accurate representation of the songs and genre I most often listen to. During my data search, I found a data set from Kaggle that had collected the top fifty songs by country that were most played on Spotify in the year 2019. I was interested to learn which songs I had listened to that also appeared in the top 50 songs of a specific country. The data set from Kaggle also contained variables regarding the beats per minute of the song, energy of the song, danceability of the song, loudness of the song, liveness of the long, valence of the song, duration of the long, acousticness of the song, and popularity of the song. Comparing these two data sets gave me a chance to learn more about the songs and genre that I tend to listen to. My dataset contained the date, song title, artist name, and the number of minutes I listened to the specific song. Combining both data resulted in the necessary amount of numeric and total variables needed. Within this report, different R-Studio functions that were learned in class will be used to manipulate the data in order to learn more about specific songs. Different statistics will be applied to specific numerical variables as well. Plots will then be made to show relationships among different numerical variables and categorical variables. Lastly, the data will be manipulated using clustering functions in order to determine more relationships among the numerical and categorical variables. It is expected that the danceability of a song would also have high energy and liveness. It is also expected that when clustering, the same artist with a different song will be close in distance to themselves.

```
library(dplyr)
library(tidyverse)
ana_spotify <- mydataset
top50spotifysongs <- mydataset2
ana_spotify2 <- ana_spotify %>% separate(endTime, into = c("Year", "Month", "Day", "EndTimeHour", "EndTi
Widerana_spotify2 <- ana_spotify2 %>% pivot_wider(names_from = "artistName", values_from = "Year")
longana_spotify2 <-  Widerana_spotify2 %>% pivot_longer(cols = 7:915, names_to = "artistName", values_to
BothData <- left_join(ana_spotify2,top50spotifysongs, by = c("trackName"= "title"))

anti_join(ana_spotify2, top50spotifysongs, by = c("trackName" ="title"))
```

```
##    Year Month  Day EndTimeHour EndTimeMinute     artistName
## 1     2    28 2019          19            13       James TW
## 2     3     1 2019          15            54       James TW
## 3     3     1 2019          15            58   Nathan Sykes
## 4     3     1 2019          16            01     Ed Sheeran
## 5     3     1 2019          17            20     Ed Sheeran
## 6     3     1 2019          17            24    Calum Scott
## 7     3     2 2019           4            14   James Arthur
## 8     3     2 2019           6            53  Noah Schnacky
## 9     3     2 2019           6            56    Brett Young
## 10    3     2 2019           7            00 Brett Eldredge
## 11    3     2 2019           7            04          LANCO
## 12    3     2 2019           7            06    Brett Young
##                            trackName msPlayed
## 1              When You Love Someone     5590
```

```
## 2              When You Love Someone   214741
## 3                Over And Over Again   246893
## 4                            Happier   156804
## 5                            Happier   205343
## 6  You Are The Reason - Duet Version   190760
## 7                       At My Weakest    42724
## 8                    Hello Beautiful   216992
## 9             In Case You Didn't Know   226200
## 10                       The Long Way   208720
## 11               Greatest Love Story   222573
## 12                             Mercy    21803
##  [ reached 'max' / getOption("max.print") -- omitted 5785 rows ]
```

```
anti_join(top50spotifysongs, ana_spotify2, by = c("title" = "trackName"))
```

```
##    X                                title        artist       top.genre year
## 1  5       All I Want for Christmas Is You  Mariah Carey       dance pop 1994
## 2 13                          Santa Tell Me Ariana Grande       dance pop 2014
## 3 14     Rockin' Around The Christmas Tree    Brenda Lee adult standards 1964
## 4 15                       Jingle Bell Rock   Bobby Helms adult standards 1992
## 5 17 Happy Xmas (War Is Over) - Remastered   John Lennon      album rock 2010
##        added bpm nrgy dnce  dB live val dur acous spch pop country
## 1 1969-12-31 150   63   34  -7    7  35 241    16    4  98   world
## 2 1969-12-31 192   62   53  -7   29  59 204     5   12  95   world
## 3 1969-12-31  67   47   59  -9   51  90 126    61    5  93   world
## 4 1969-12-31 120   42   75  -8    7  81 131    64    4  92   world
## 5 1969-12-31 147   61   33 -11   77  40 214    32    3  92   world
##  [ reached 'max' / getOption("max.print") -- omitted 788 rows ]
```

To start, my dataset was made tidy by using the function "separate" to make the date the song was played into its own variables rather than being combined with the time the song stopped playing. The date was separated even more into "Year", "Day", and "Month". This was done to produce a clearer way to read the date. The "wider" function and long function were used to demonstrate how those functions work. Artist name and year were made longer and then wider. When they were made wider, every artist had a column with a year found in one of their rows. After, the artist and year were reverted to their own distinct columns using the "longer" function.There are 5,807 tracks that are in Ana's Spotify data set that are not found in the Top 50 Spotify data set. There are 803 tracks that are in the Top 50 Spotify data set that are not found in Ana's Spotify data set.In order to keep minutes played for every track on Ana's Spotify data set, the "left_join" function was used to combine both data sets by the common variable "track" and "title" of the song. After doing this, the variables that were found in the Top 50 Spotify data were combined with Ana's Spotify data. Rows that matched tracks found in both data sets contained a result. Any track that was in Ana's Spotify data that was not found in the Top 50 Spotify data set contained rows with "NA" under the new variables added.

```
BothData %>% group_by(artistName, trackName) %>% summarize(mean_minsplayed = mean(msPlayed)) %>% arrang
```

```
## # A tibble: 2,105 x 3
## # Groups:   artistName [909]
##    artistName     trackName                                    mean_minsplayed
##    <chr>          <chr>                                                  <dbl>
##  1 Housefires     Iâ€ll Give Thanks - Live                              536653
##  2 Hillsong UNIT~ As You Find Me - Live                                 522120
```

```
##  3 Bob Dylan      Like a Rolling Stone - Live at Royal Albert H~         505240
##  4 Jesus Culture  Your Love Never Fails - Live                         474217
##  5 Hillsong Wors~ Broken Vessels (Amazing Grace) - Live                 450170.
##  6 Michael W. Sm~ Waymaker                                             437131
##  7 Paul Cardall   Letting Go                                           412266
##  8 Eagles         Hotel California - Live at The Forum, Los Ang~       409520
##  9 Piano Prayer   So Will I (100 Billion X)                           409445
## 10 Hillsong UNIT~ So Will I (100 Billion X)                           407725
## # ... with 2,095 more rows
```

```
BothData %>% filter(artistName == "Maroon 5")
```

```
##   Year Month  Day EndTimeHour EndTimeMinute artistName trackName msPlayed  X
## 1    3     5 2019          14            02   Maroon 5       Sad   194253 NA
## 2    3    13 2019          14            56   Maroon 5       Sad    65387 NA
## 3    4     4 2019          14            05   Maroon 5       Sad   194253 NA
## 4    4    10 2019          17            18   Maroon 5       Sad   194253 NA
##   artist top.genre year added bpm nrgy dnce dB live val dur acous spch pop
## 1   <NA>      <NA>   NA  <NA>  NA   NA   NA NA   NA  NA  NA    NA   NA  NA
## 2   <NA>      <NA>   NA  <NA>  NA   NA   NA NA   NA  NA  NA    NA   NA  NA
## 3   <NA>      <NA>   NA  <NA>  NA   NA   NA NA   NA  NA  NA    NA   NA  NA
## 4   <NA>      <NA>   NA  <NA>  NA   NA   NA NA   NA  NA  NA    NA   NA  NA
##   country
## 1    <NA>
## 2    <NA>
## 3    <NA>
## 4    <NA>
##  [ reached 'max' / getOption("max.print") -- omitted 122 rows ]
```

```
BothData %>% select(artistName, trackName)
```

```
##            artistName                                  trackName
## 1            James TW                When You Love Someone
## 2            James TW                When You Love Someone
## 3        Nathan Sykes                  Over And Over Again
## 4          Ed Sheeran                              Happier
## 5          Ed Sheeran                              Happier
## 6         Calum Scott    You Are The Reason - Duet Version
## 7        James Arthur                         At My Weakest
## 8       Noah Schnacky                       Hello Beautiful
## 9         Brett Young                In Case You Didn't Know
## 10     Brett Eldredge                           The Long Way
## 11              LANCO                  Greatest Love Story
## 12        Brett Young                               Mercy
## 13        Brett Young                               Mercy
## 14       Thomas Rhett                             Marry Me
## 15  Russell Dickerson                                Yours
## 16         Dan + Shay                              Tequila
## 17         Dan + Shay   From the Ground Up - Single Version
## 18       Thomas Rhett                       Die A Happy Man
## 19    Imagine Dragons                              Thunder
## 20   Hailee Steinfeld      Back to Life - from "Bumblebee"
## 21  Russell Dickerson                                Yours
```

```
## 22  Russell Dickerson                                     Yours
## 23       Thomas Rhett                                  Marry Me
## 24         Dan + Shay        From the Ground Up - Single Version
## 25        Brett Young                    In Case You Didn't Know
## 26       Maren Morris                    I Could Use a Love Song
## 27      Noah Schnacky                             Hello Beautiful
## 28      Noah Schnacky                             Hello Beautiful
## 29     Brett Eldredge                                The Long Way
## 30     Brett Eldredge                                The Long Way
## 31         Dan + Shay                                     Tequila
## 32        Brett Young                                      Mercy
## 33       Thomas Rhett                             Die A Happy Man
## 34       Thomas Rhett                             Die A Happy Man
## 35              LANCO                         Greatest Love Story
## 36        Keith Urban                                    Female
## 37         Dan + Shay                                 Speechless
## 38         Matt Maher                          Burning In My Soul
## 39           Maroon 5                                       Sad
## 40         Matt Maher Your Grace Is Enough/Here I Am Lord (Live)
## 41        Leona Lewis                                      Happy
## 42 David Crowder Band                                How He Loves
## 43       Thomas Rhett                                   Marry Me
## 44         Dan + Shay                                 Speechless
## 45         Dan + Shay        From the Ground Up - Single Version
## 46       Thomas Rhett                             Die A Happy Man
## 47        Brett Young                    In Case You Didn't Know
## 48       Maren Morris                    I Could Use a Love Song
## 49  Russell Dickerson                                     Yours
## 50     Brett Eldredge                                The Long Way
##  [ reached 'max' / getOption("max.print") -- omitted 6627 rows ]
```

```
BothData%>% select(artistName, trackName, msPlayed) %>% arrange(desc(msPlayed))
```

```
##                    artistName
## 1             Hillsong Worship
## 2                   Housefires
## 3              Hillsong UNITED
## 4              Hillsong UNITED
## 5              Hillsong UNITED
## 6              Hillsong UNITED
## 7              Hillsong UNITED
## 8                    Bob Dylan
## 9              Hillsong UNITED
## 10               Jesus Culture
## 11            Michael W. Smith
## 12                Paul Cardall
## 13                      Eagles
## 14               Piano Prayer
## 15             Hillsong UNITED
## 16                    Coldplay
## 17               Taylor Swift
## 18               Taylor Swift
## 19                    Futures
## 20            Meredith Andrews
```

```
## 21                    Coldplay
## 22               Hillsong UNITED
## 23                  Brad Paisley
## 24               Hillsong UNITED
## 25               Hillsong UNITED
## 26                    Matt Maher
## 27                    Matt Maher
## 28                  Chris Tomlin
## 29                  Chris Tomlin
## 30                  Chris Tomlin
## 31 Hot Country: Original Videos
## 32                  Harry Styles
## 33                 NEEDTOBREATHE
##                                                                    trackName
## 1                                          Broken Vessels (Amazing Grace) - Live
## 2                                                Iâ\200\231ll Give Thanks - Live
## 3                                                Oceans (Where Feet May Fail)
## 4                                                Oceans (Where Feet May Fail)
## 5                                                Oceans (Where Feet May Fail)
## 6                                                Oceans (Where Feet May Fail)
## 7                                                     As You Find Me - Live
## 8      Like a Rolling Stone - Live at Royal Albert Hall, London, UK - May 26, 1966
## 9                                                Oceans (Where Feet May Fail)
## 10                                               Your Love Never Fails - Live
## 11                                                                 Waymaker
## 12                                                               Letting Go
## 13           Hotel California - Live at The Forum, Los Angeles, CA, 10/20-22/1976
## 14                                                 So Will I (100 Billion X)
## 15                                                 So Will I (100 Billion X)
## 16                                                                     Up&Up
## 17                                                               Dear John
## 18                                                               Dear John
## 19                                                 â\200  (just the cross) - Live
## 20                                                        Open Over Us (Live)
## 21                                                        Hypnotised - EP Mix
## 22                                                 Here Now (Madness) - Live
## 23                                                               New Again
## 24                                           Even When It Hurts (Praise Song)
## 25                                               Oceans (Where Feet May Fail)
## 26                                 Your Grace Is Enough/Here I Am Lord (Live)
## 27                                 Your Grace Is Enough/Here I Am Lord (Live)
## 28                                       How Great Is Our God - World Edition
## 29                                       How Great Is Our God - World Edition
## 30                                       How Great Is Our God - World Edition
## 31                                              Thomas Rhett: My Center Point
## 32                                                                        She
## 33 Stones Under Rushing Water (feat. Drew Holcomb & Ellie Holcomb) [Acoustic Live]
##    msPlayed
## 1    568786
## 2    536653
## 3    536000
## 4    536000
## 5    536000
## 6    530032
```

```
## 7    522120
## 8    505240
## 9    497787
## 10   474217
## 11   437131
## 12   412266
## 13   409520
## 14   409445
## 15   407725
## 16   405320
## 17   403920
## 18   403920
## 19   398333
## 20   396064
## 21   391413
## 22   389496
## 23   382400
## 24   378400
## 25   374955
## 26   365880
## 27   365880
## 28   365134
## 29   365134
## 30   365134
## 31   364384
## 32   362653
## 33   361322
##  [ reached 'max' / getOption("max.print") -- omitted 6644 rows ]
```

```
BothData %>% mutate(hoursplayed = msPlayed/60)
```

```
##   Year Month  Day EndTimeHour EndTimeMinute   artistName             trackName
## 1    2    28 2019          19            13     James TW When You Love Someone
## 2    3     1 2019          15            54     James TW When You Love Someone
## 3    3     1 2019          15            58 Nathan Sykes   Over And Over Again
## 4    3     1 2019          16            01   Ed Sheeran               Happier
##   msPlayed  X artist top.genre year added bpm nrgy dnce dB live val dur acous
## 1     5590 NA   <NA>      <NA>   NA  <NA>  NA   NA   NA NA   NA  NA  NA    NA
## 2   214741 NA   <NA>      <NA>   NA  <NA>  NA   NA   NA NA   NA  NA  NA    NA
## 3   246893 NA   <NA>      <NA>   NA  <NA>  NA   NA   NA NA   NA  NA  NA    NA
## 4   156804 NA   <NA>      <NA>   NA  <NA>  NA   NA   NA NA   NA  NA  NA    NA
##   spch pop country hoursplayed
## 1   NA  NA    <NA>    93.16667
## 2   NA  NA    <NA>  3579.01667
## 3   NA  NA    <NA>  4114.88333
## 4   NA  NA    <NA>  2613.40000
##  [ reached 'max' / getOption("max.print") -- omitted 6673 rows ]
```

The following dplry functions were used on the new combined data set "BothData": filter, select, arrange, group_by, mutate, and summarize. The function "group_by", "summarize", and "arrange" were first used to get the mean minutes played of a specific track in descending order. If a song was listened to more than once, the group_by function ensured that the result would produce the mean minutes played of the track. The results will be further discussed in the summary statistics below. The next function, "filter", was used to

obtain only the rows containing information about the band Maroon 5. It can be observed that the only track by Maroon 5 that I listened to that was also on the Top 50 charts was their song "Memories". The "select" function was used to obtain only the columns "artistName" and "trackName". This function allowed me to look over all the artists and songs I listened to. The final dplyr function used was "mutate". This function was used to convert the minutes the song was played into the hours the song was played.

```
BothData%>% summarize_at(c("msPlayed", "bpm", "nrgy", "dnce", "live", "pop"), mean, na.rm= T)
```

```
##   msPlayed      bpm     nrgy     dnce     live      pop
## 1 157720.7 110.2818 54.90682 70.07841 14.46705 94.91591
```

```
BothData %>% group_by(artistName, trackName) %>% summarize(mean_minsplayed = mean(msPlayed)) %>% arrang
```

```
## # A tibble: 2,105 x 3
## # Groups:   artistName [909]
##    artistName     trackName                                        mean_minsplayed
##    <chr>          <chr>                                                      <dbl>
##  1 Housefires     Iâ€ll Give Thanks - Live                                  536653
##  2 Hillsong UNIT~ As You Find Me - Live                                     522120
##  3 Bob Dylan      Like a Rolling Stone - Live at Royal Albert H~            505240
##  4 Jesus Culture  Your Love Never Fails - Live                              474217
##  5 Hillsong Wors~ Broken Vessels (Amazing Grace) - Live                     450170.
##  6 Michael W. Sm~ Waymaker                                                  437131
##  7 Paul Cardall   Letting Go                                                412266
##  8 Eagles         Hotel California - Live at The Forum, Los Ang~            409520
##  9 Piano Prayer   So Will I (100 Billion X)                                 409445
## 10 Hillsong UNIT~ So Will I (100 Billion X)                                 407725
## # ... with 2,095 more rows
```

```
BothData %>% summarize_at(c("msPlayed", "bpm", "nrgy", "dnce", "live", "pop"), sd, na.rm= T)
```

```
##   msPlayed      bpm     nrgy     dnce     live      pop
## 1 85605.09 22.83014 16.72206 11.24551 10.46534 5.869434
```

```
BothData %>% group_by(trackName) %>% summarize(sd_minsplayed = sd(msPlayed)) %>% arrange(desc(sd_minspla
```

```
## # A tibble: 2,044 x 2
##    trackName                                                    sd_minsplayed
##    <chr>                                                                <dbl>
##  1 Oceans (Where Feet May Fail)                                        233043.
##  2 Here Now (Madness) - Live                                           230001.
##  3 Go Loko                                                             192448.
##  4 I Am Free - Live                                                    189582.
##  5 Bed Of Lies                                                         187753.
##  6 Mary Jane's Last Dance                                              184508.
##  7 Stones Under Rushing Water (feat. Drew Holcomb & Ellie Holcomb~     181626.
##  8 Not Today                                                           179440.
##  9 Believe                                                             172853.
## 10 Holding On to You                                                   170866.
## # ... with 2,034 more rows
```

```
BothData %>% summarize_at(c("msPlayed", "bpm", "nrgy", "dnce", "live", "pop"), var, na.rm= T)
```

```
##     msPlayed      bpm     nrgy     dnce     live      pop
## 1 7328231410 521.2151 279.6273 126.4614 109.5234 34.45026
```

```
BothData %>% group_by(trackName) %>% summarize(var_minsplayed = var(msPlayed)) %>% arrange(desc(var_mins
```

```
## # A tibble: 2,044 x 2
##    trackName                                                  var_minsplayed
##    <chr>                                                               <dbl>
##  1 Oceans (Where Feet May Fail)                                54309168912.
##  2 Here Now (Madness) - Live                                   52900286450
##  3 Go Loko                                                     37036349284.
##  4 I Am Free - Live                                            35941486050
##  5 Bed Of Lies                                                 35251231764.
##  6 Mary Jane's Last Dance                                      34043276178
##  7 Stones Under Rushing Water (feat. Drew Holcomb & Ellie Holcom~   32988158597.
##  8 Not Today                                                   32198591378
##  9 Believe                                                     29878264650.
## 10 Holding On to You                                           29195186440.
## # ... with 2,034 more rows
```

```
BothData %>% summarize_at(c("msPlayed", "bpm", "nrgy", "dnce", "live", "pop"), n_distinct, na.rm= T)
```

```
##   msPlayed bpm nrgy dnce live pop
## 1     3682  39   37   37   27  23
```

```
BothData %>% summarize_at(c("msPlayed", "bpm", "nrgy", "dnce", "live", "pop"), min, na.rm= T)
```

```
##   msPlayed bpm nrgy dnce live pop
## 1        0  76   18   24    5  55
```

```
BothData %>% summarize_at(c("msPlayed", "bpm", "nrgy", "dnce", "live"), max, na.rm= T)
```

```
##   msPlayed bpm nrgy dnce live
## 1   568786 205   90   90   79
```

```
BothData %>% summarize_at(c("msPlayed", "bpm", "nrgy", "dnce", "live", "pop"), median, na.rm= T)
```

```
##   msPlayed bpm nrgy dnce live pop
## 1   185706 102   59   72   10  97
```

```
BothData %>% count(msPlayed, bpm, nrgy, dnce, live, pop, sort = TRUE)
```

```
## # A tibble: 3,699 x 7
##    msPlayed   bpm  nrgy  dnce  live   pop     n
##       <int> <int> <int> <int> <int> <int> <int>
##  1   209438    98    59    82    15   100    85
```

```
##  2   189486    91    32    76     8    99    50
##  3        0    NA    NA    NA    NA    NA    42
##  4   197866    93    65    64     8    86    40
##  5   215280   120    76    70     9    99    40
##  6   197436    NA    NA    NA    NA    NA    36
##  7   230266    NA    NA    NA    NA    NA    32
##  8   182160   110    41    50    11    96    30
##  9   211466    NA    NA    NA    NA    NA    25
## 10   210240    NA    NA    NA    NA    NA    21
## # ... with 3,689 more rows
```

```
df2 <- BothData %>% na.omit %>% select_if(is.numeric)
cor(df2)
```

```
##              msPlayed             X         year            bpm         nrgy
## msPlayed  1.000000000 -0.0343855105 -0.009958563  0.0157395473  0.045262857
## X        -0.034385510  1.0000000000  0.069458115  0.0003019433 -0.113128476
## year     -0.009958563  0.0694581152  1.000000000 -0.3899771586  0.019270399
## bpm       0.015739547  0.0003019433 -0.389977159  1.0000000000  0.166744919
## nrgy      0.045262857 -0.1131284757  0.019270399  0.1667449188  1.000000000
## dnce     -0.037894235 -0.1812888776  0.385382805 -0.4185664137 -0.003113058
## dB        0.030209990 -0.1264744361  0.158973055 -0.1165061411  0.680832455
##                 dnce          dB        live           val         dur
## msPlayed -0.037894235  0.03020999 -0.16858692 -0.1033908876  0.34476754
## X        -0.181288878 -0.12647444  0.01846943 -0.2005572382 -0.05006105
## year      0.385382805  0.15897306 -0.05916682 -0.2138727317  0.19324458
## bpm      -0.418566414 -0.11650614  0.06245344  0.0003922164 -0.11665668
## nrgy     -0.003113058  0.68083246  0.08787348  0.3372810777  0.20351155
## dnce      1.000000000  0.07728414 -0.11182669  0.3022735742  0.12540671
## dB        0.077284138  1.00000000 -0.05129385  0.3003267285  0.01514346
##                acous         spch         pop
## msPlayed  0.11394478 -0.179626381 -0.08336579
## X        -0.08423954  0.040349291 -0.18074348
## year     -0.14028392  0.085870782  0.16876230
## bpm      -0.24085610  0.181438006 -0.16327193
## nrgy     -0.71804493  0.060624337 -0.27452270
## dnce      0.10821232  0.009830399  0.28764778
## dB       -0.28894885 -0.353240570 -0.03729488
##  [ reached getOption("max.print") -- omitted 6 rows ]
```

```
df3 <- BothData %>% na.omit %>% select(msPlayed, bpm, nrgy, dnce, live, pop)
cor(df3)
```

```
##              msPlayed         bpm         nrgy         dnce        live
## msPlayed  1.00000000  0.01573955  0.045262857 -0.037894235 -0.16858692
## bpm       0.01573955  1.00000000  0.166744919 -0.418566414  0.06245344
## nrgy      0.04526286  0.16674492  1.000000000 -0.003113058  0.08787348
## dnce     -0.03789424 -0.41856641 -0.003113058  1.000000000 -0.11182669
## live     -0.16858692  0.06245344  0.087873485 -0.111826687  1.00000000
## pop      -0.08336579 -0.16327193 -0.274522703  0.287647776  0.01188228
##                 pop
## msPlayed -0.08336579
## bpm      -0.16327193
```

```
## nrgy     -0.27452270
## dnce      0.28764778
## live      0.01188228
## pop       1.00000000
```

```
summarystatsofdata <- BothData %>% select(msPlayed, bpm, nrgy, dnce, live, pop)
summary(summarystatsofdata, digits = 2)
```

```
##     msPlayed           bpm            nrgy           dnce           live
##  Min.   :     0   Min.   : 76    Min.   :18     Min.   :24     Min.   : 5
##  1st Qu.: 95373   1st Qu.: 93    1st Qu.:41     1st Qu.:64     1st Qu.: 8
##  Median :185706   Median :102    Median :59     Median :72     Median :10
##  Mean   :157721   Mean   :110    Mean   :55     Mean   :70     Mean   :14
##  3rd Qu.:215853   3rd Qu.:120    3rd Qu.:67     3rd Qu.:78     3rd Qu.:15
##  Max.   :568786   Max.   :205    Max.   :90     Max.   :90     Max.   :79
##                   NA's   :5797   NA's   :5797   NA's   :5797   NA's   :5797
##       pop
##  Min.   : 55
##  1st Qu.: 93
##  Median : 97
##  Mean   : 95
##  3rd Qu.: 99
##  Max.   :100
##  NA's   :5797
```

```
CleanData1 <- BothData[!duplicated(BothData$trackName),]
BothDataClean <- CleanData1%>% na.omit()

summarystats2 <- BothDataClean %>% na.omit %>% select(msPlayed, bpm, nrgy, dnce, live, pop)
cor(summarystats2)
```

```
##             msPlayed         bpm        nrgy        dnce        live
## msPlayed  1.00000000  0.19088604  0.02723828 -0.31748040 -0.15036434
## bpm       0.19088604  1.00000000  0.07331165 -0.39709526 -0.07431167
## nrgy      0.02723828  0.07331165  1.00000000 -0.02071731 -0.13723650
## dnce     -0.31748040 -0.39709526 -0.02071731  1.00000000  0.05002308
## live     -0.15036434 -0.07431167 -0.13723650  0.05002308  1.00000000
## pop      -0.20810634 -0.01250169 -0.22091835  0.15395993 -0.18134883
##                  pop
## msPlayed -0.20810634
## bpm      -0.01250169
## nrgy     -0.22091835
## dnce      0.15395993
## live     -0.18134883
## pop       1.00000000
```

```
BothDataClean %>% group_by(top.genre) %>% summarise(n=n()) %>% mutate(frequency = n/sum(n)) %>% arrange
```

```
## # A tibble: 26 x 3
##    top.genre                    n frequency
##    <chr>                    <int>     <dbl>
##  1 pop                         12    0.194
```

```
##  2 dance pop                     9    0.145
##  3 latin                         6    0.0968
##  4 dfw rap                       3    0.0484
##  5 electropop                    3    0.0484
##  6 pop rap                       3    0.0484
##  7 adult standards               2    0.0323
##  8 australian pop                2    0.0323
##  9 canadian contemporary r&b     2    0.0323
## 10 chicago rap                   2    0.0323
## # ... with 16 more rows
```

```
BothData%>% group_by(trackName) %>% summarise(n=n()) %>% mutate(frequency = n/sum(n)) %>% arrange(desc(
```

```
## # A tibble: 2,044 x 3
##    trackName                           n frequency
##    <chr>                           <int>     <dbl>
##  1 Memories                          130   0.0195
##  2 Dance Monkey                      119   0.0178
##  3 Circles                            56   0.00839
##  4 Beautiful People (feat. Khalid)    52   0.00779
##  5 bad guy                            50   0.00749
##  6 Say You Won't Let Go               48   0.00719
##  7 I Like Me Better                   43   0.00644
##  8 ROXANNE                            42   0.00629
##  9 Praying                            40   0.00599
## 10 Someone You Loved                  40   0.00599
## # ... with 2,034 more rows
```

```
BothData%>% na.omit %>% group_by(country) %>% summarise(n=n()) %>% mutate(frequency = n/sum(n)) %>% arra
```

```
## # A tibble: 20 x 3
##    country       n frequency
##    <chr>     <int>     <dbl>
##  1 malasya     155   0.176
##  2 israel       97   0.110
##  3 indonesia    92   0.105
##  4 belgium      63   0.0716
##  5 world        58   0.0659
##  6 india        56   0.0636
##  7 africa       45   0.0511
##  8 australia    44   0.05
##  9 france       40   0.0455
## 10 italy        33   0.0375
## 11 argentina    32   0.0364
## 12 germany      27   0.0307
## 13 bolivia      21   0.0239
## 14 colombia     21   0.0239
## 15 spain        21   0.0239
## 16 usa          21   0.0239
## 17 chile        20   0.0227
## 18 japan        14   0.0159
## 19 canada       13   0.0148
## 20 brazil        7   0.00795
```

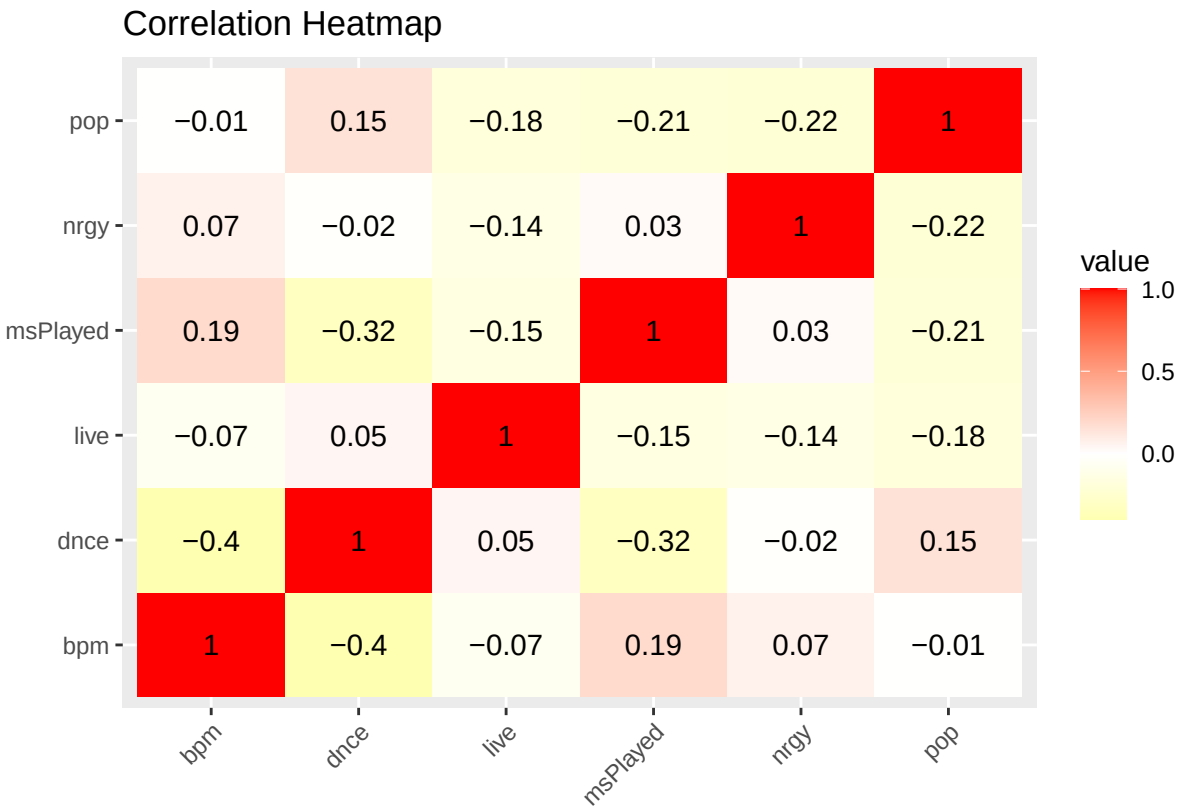
```
BothData%>% group_by(artistName) %>% summarise(n=n()) %>% mutate(frequency = n/sum(n)) %>% arrange(desc
```

```
## # A tibble: 909 x 3
##    artistName         n frequency
##    <chr>          <int>     <dbl>
##  1 Taylor Swift     543    0.0813
##  2 Jonas Brothers   343    0.0514
##  3 Post Malone      151    0.0226
##  4 Ed Sheeran       128    0.0192
##  5 Maroon 5         126    0.0189
##  6 Thomas Rhett     122    0.0183
##  7 Tones and I      121    0.0181
##  8 Shawn Mendes     102    0.0153
##  9 Selena Gomez      96    0.0144
## 10 Dan + Shay        94    0.0141
## # ... with 899 more rows
```

Since the "BothData" contained many numerical variables, only specific numerical variables were chosen to run summary statistics on. The following numerical variables were chosen to run summary statistics on: danceability level (dnce), energy level (ngry), liveness of the song (live), popularity level (pop), and minutes the song was played (msPlayed). The first statistic run on the numerical variables was mean. Mean calculates the central average of each numerical variable. To determine which song had the highest mean minutes played, the "group_by" and "arrange" functions were used. It was observed that the song, "I'll Give Thanks" had the highest mean minutes played. Standard deviation of the numerical variables was than obtained. Popularity level and liveliness had the lowest standard deviation. This indicated that those two variables variations from their mean was low. When grouped by track name and arranged in descending order, it was determined that the song "Oceans" had the highest standard deviation of minutes played. Variance of the numerical variables was then taken, and it was found that "Oceans" had the highest variance of minutes played which was expected. The amount of each unique value within each numerical variable was obtained using the "n_distinct" function. Minutes played had the most unique values out of the numerical variables. Energy level had the smallest numerical value which was determined using the function "min". The "max" function resulted in minutes played variable having the largest numerical value. To determine the middle numerical value of each numerical variable, the function "median" was used. The "count" function obtained the number of values within each row. A correlation matrix was then created for the numerical variables. There were no significant correlations among the numerical variables. To see if there were higher results among the numerical variables if there were no duplicates or NAs among the data, the duplicates of track names were removed from the dataset. The NAs were also removed from the combined dataset. After doing this, higher values were obtained. A table was then created to display the original combined data's min, max, median, mean, 1st quartile, 3rd quartile, and the number of NA's. among each numerical variable. This provided a clearer way to compare results. Frequency was determined for categorical variables. It was found that the "pop" genre had the highest frequency among the dataset. The song "Memories" by Marron 5 had the highest frequency among the tracks in the dataset. Malaysia was the country that had the highest frequency among the tracks in the dataset. Taylor Swift was the most played artist in the dataset.

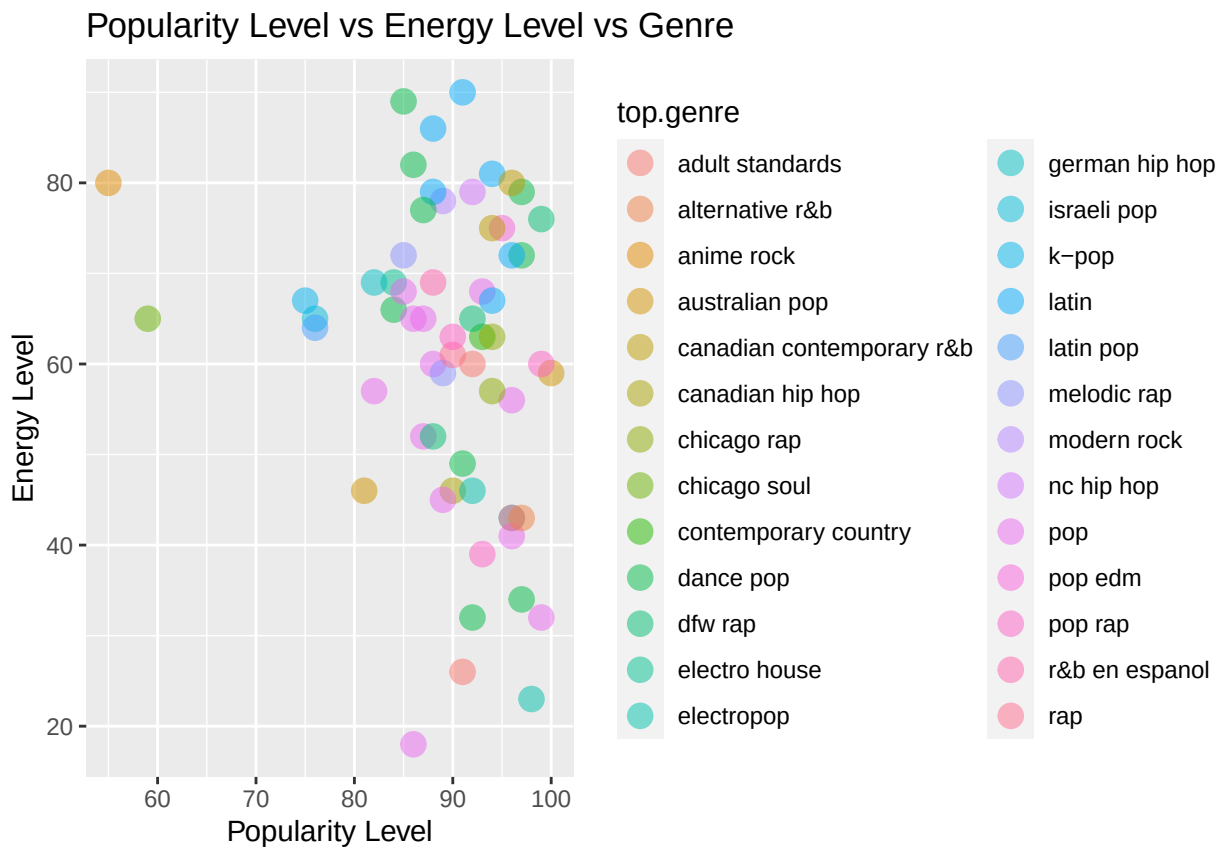
```
library(tidyverse)
library(ggplot2)
library(dplyr)

#Plot 1:
BothDataClean %>% select(msPlayed, bpm, nrgy, dnce, live, pop) %>% cor() %>% as.data.frame() %>% rowname
```



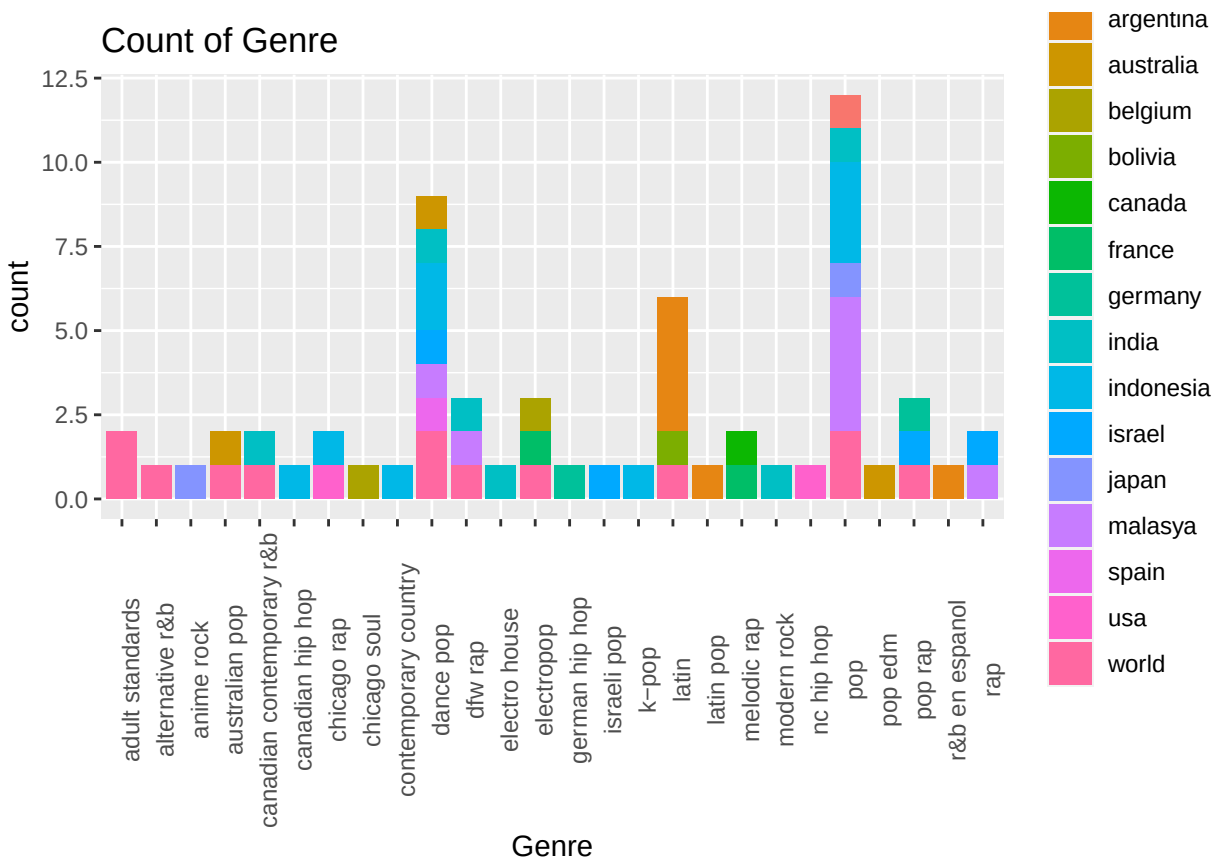
The combined data without duplicates nor NAs was used for the rest of the project. The first plot made was a correlation heatmap. Based on the correlation heatmap, it can be determined that there were no significant correlations between the numerical variables chosen. It was observed that danceability level and popularity level had a slightly high potential relationship. This could also be observed between beats per minutes and minutes played. Energy level and beats per minute had a low potential relationship between the two.

```
#Plot 2:
BothDataClean%>% ggplot(aes(x=pop, y=nrgy, color = top.genre)) + geom_point(size = 4, alpha = .5) + ggt
```

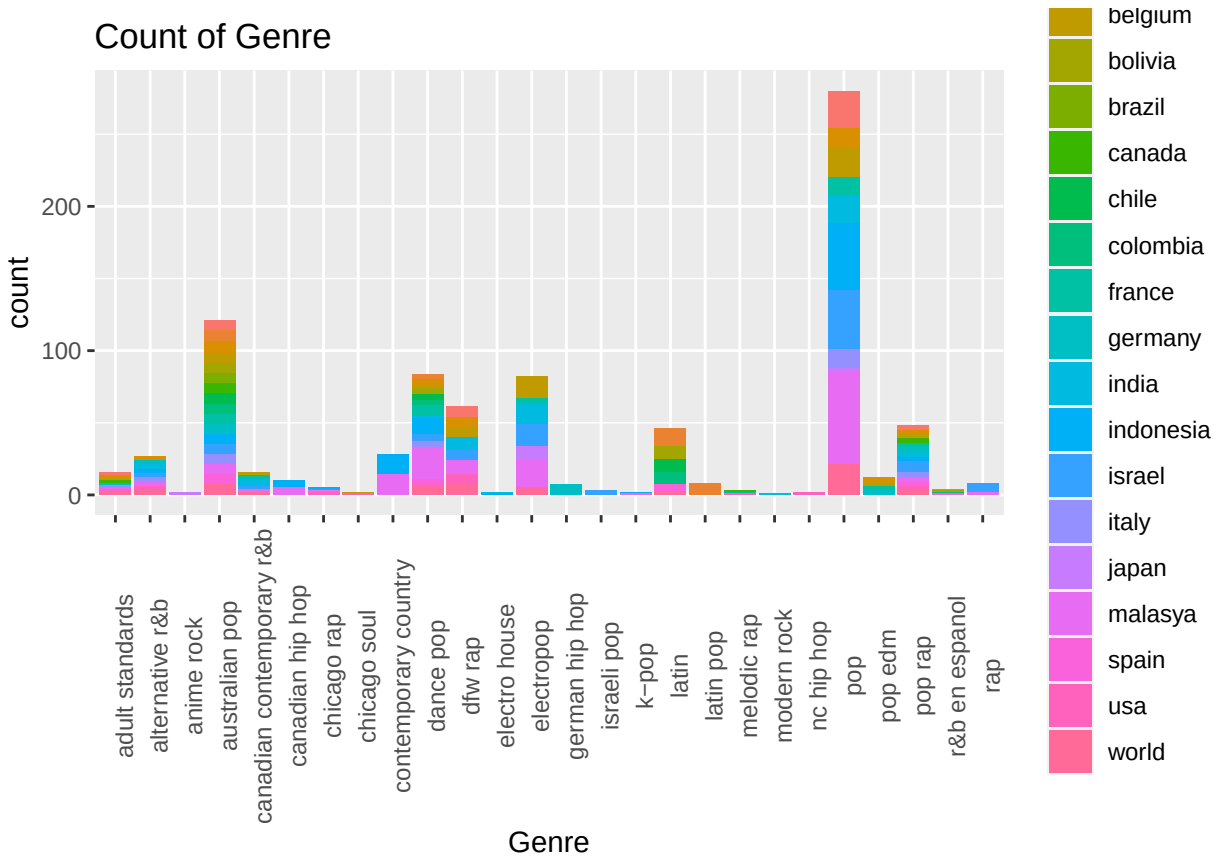


The second plot made was a scatter plot comparing the energy level of a track with the beats per minute of the track while also visualizing the different genres in the dataset. From this plot, it can be observed that the popularity level and energy level was high for most tracks. It can also be observed that one of the least popular tracks was under the genre "anime rock".

```
#Plot 3:
ggplot(BothDataClean, aes(x=top.genre, fill= country)) + geom_bar(stat = "count") + theme(axis.text.x =
```

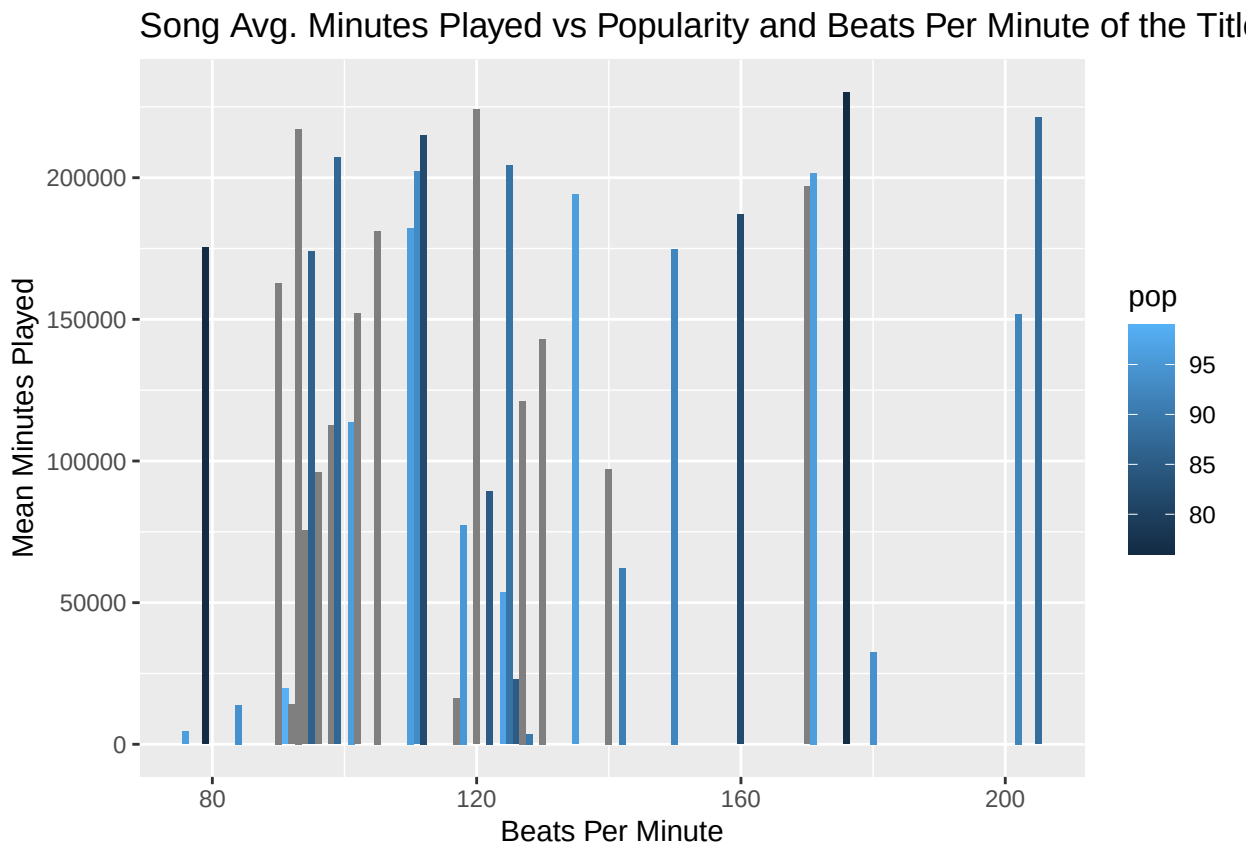



```
BothData %>% na.omit %>% ggplot(aes(x= top.genre, fill= country)) + geom_bar(stat = "count") + theme(ax
```



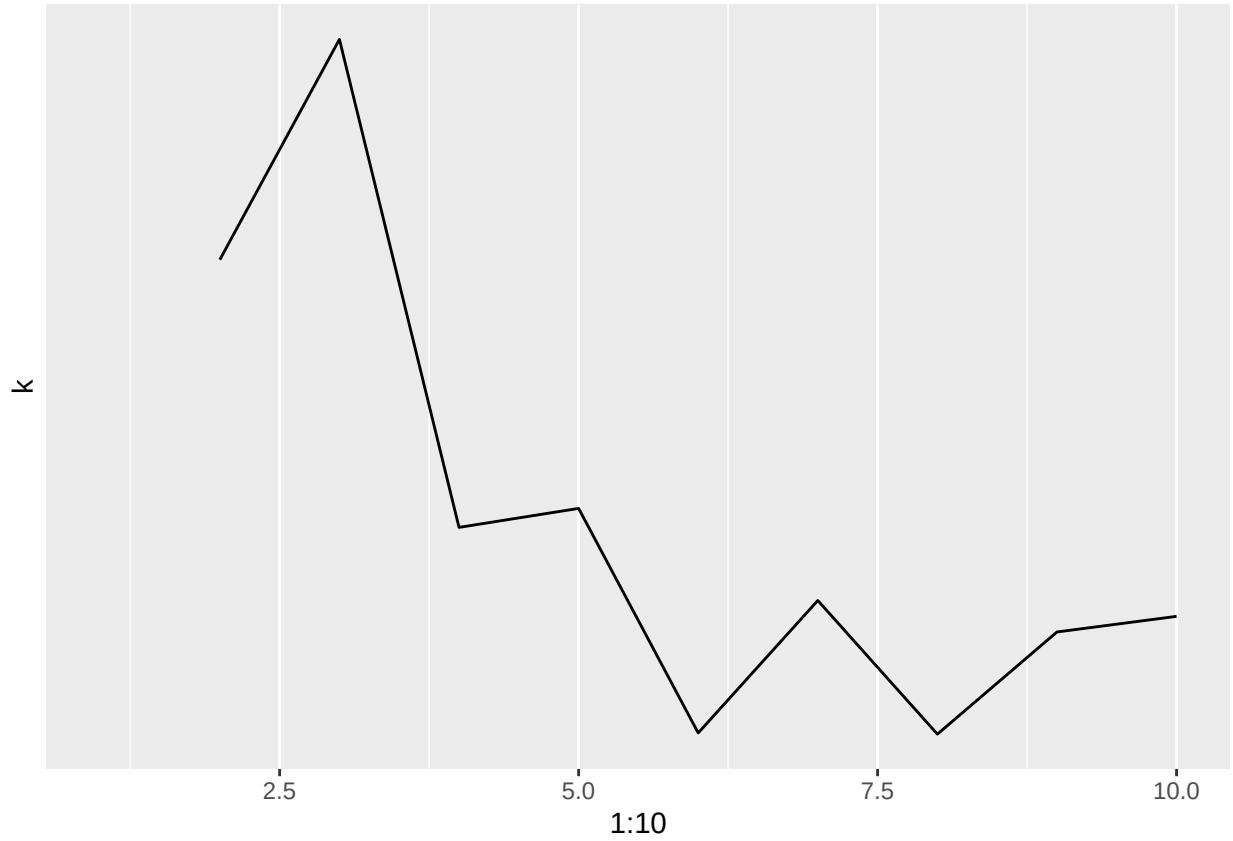
The third plot was used to determine what type of genre I most often listen to. This was done by creating a plot with the stat function "count". The plot also included which country that genre was popular in. It can be observed that the Pop Genre and Dance Pop Genre are the two genres that I listen to most often. It can also be determined that those two genres are genres that are the most popular in more than three countries. To ensure this was accurate the plot was made again with the original data that had the duplicates present. When this was done, it was found that Australian Pop Genre was among the highest genres that I listen to.

```
#Plot 4:
BothDataClean %>% ggplot(aes(x = bpm, fill = pop)) + geom_bar(aes(y = msPlayed, width = 1), stat = "sum
```



The last plot made was to determine if there was a relationship with the mean minutes played and beats per minute that was potentially visible in the correlation matrix. In this final plot, popularity was also considered. The bar graph produced showed that most beats per minute values had a high mean minutes played. It didn't appear to increase with increasing beats per minute which could be because there were not a lot of songs with high beats per minute.

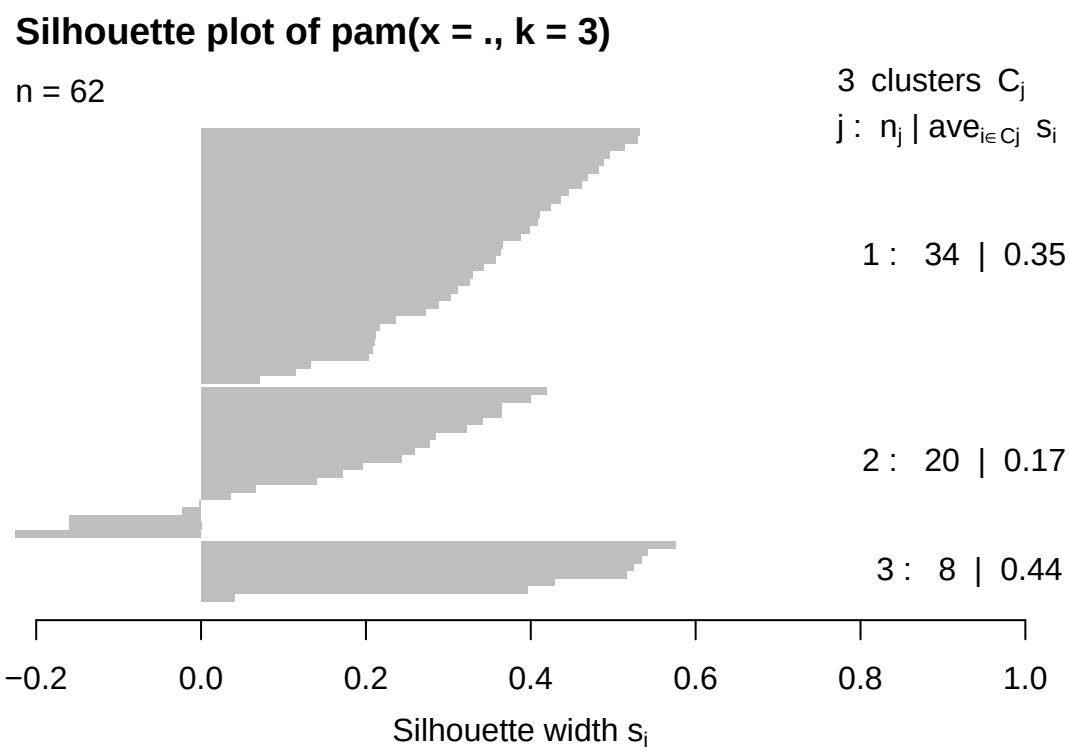
```
library(ggplot2)
library(tidyverse)
library(cluster)
sil_width2<- vector()
clusteringdata1 <- BothDataClean %>% select(nrgy, bpm, pop,live, acous, pop, spch, everything())
sil_width1 <- vector()
for(i in 2:10){
  pam_fit2 <- clusteringdata1 %>% select(nrgy, bpm, pop, dnce, live, acous, spch, pop) %>% pam(i)
  sil_width1[i] <- pam_fit2$silinfo$avg.width
}
ggplot() + geom_line(aes(x=1:10, y= sil_width1))+ scale_y_continuous(name="k", breaks = 1:10)
```



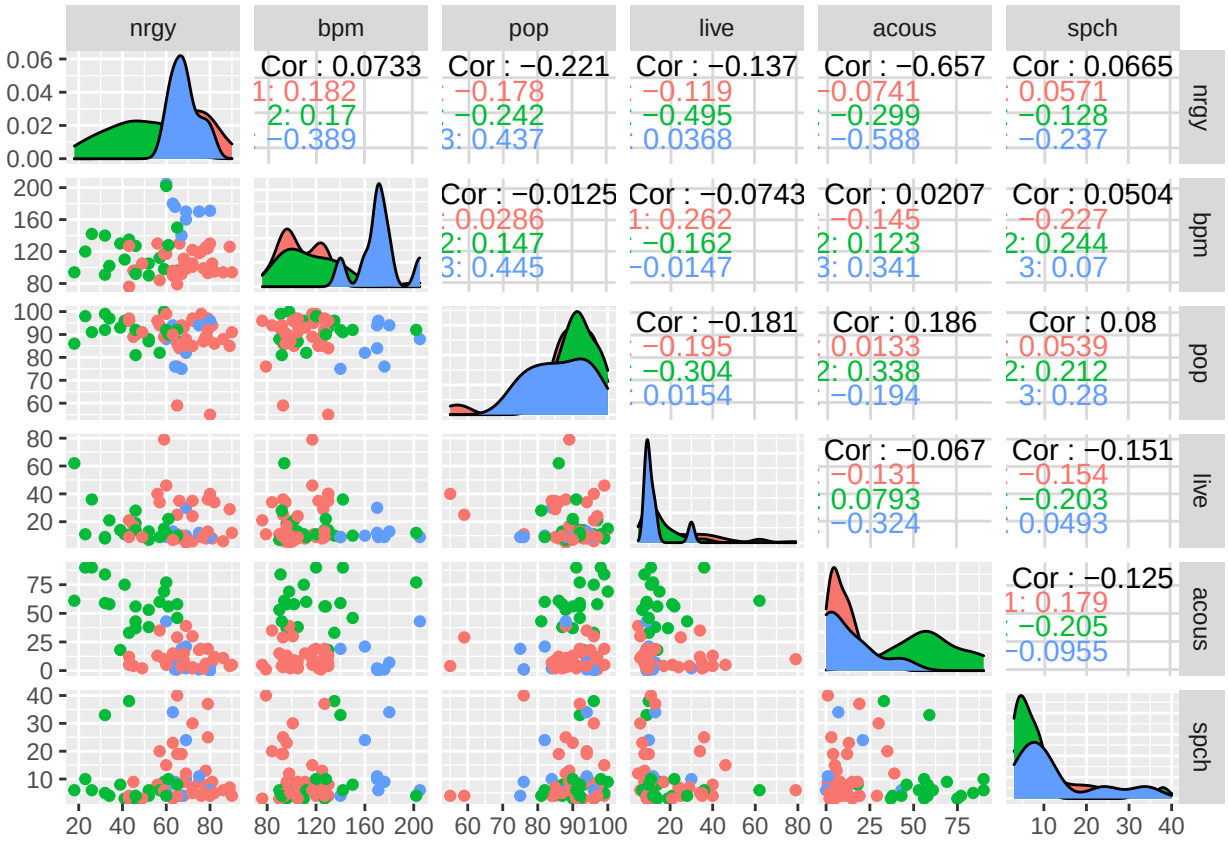

```
pam1 <- clusteringdata1 %>% select(nrgy, bpm, pop, dnce, live, acous, spch, pop) %>% pam(3)
final1 <- clusteringdata1 %>% mutate(cluster=as.factor(pam1$clustering))
confmat1 <- final1%>% group_by(top.genre) %>% count(cluster) %>% arrange(desc(n)) %>% pivot_wider(names_
confmat1
```

```
## # A tibble: 26 x 4
## # Groups:   top.genre [26]
##    top.genre                   `1`   `2`   `3`
##    <chr>                     <int> <int> <int>
##  1 dance pop                     7     2     0
##  2 latin                         6     0     0
##  3 pop                           5     6     1
##  4 electropop                    0     3     0
##  5 adult standards               0     2     0
##  6 australian pop                0     2     0
##  7 canadian contemporary r&b     0     0     2
##  8 dfw rap                       1     2     0
##  9 melodic rap                   2     0     0
## 10 pop rap                       2     1     0
## # ... with 16 more rows
```

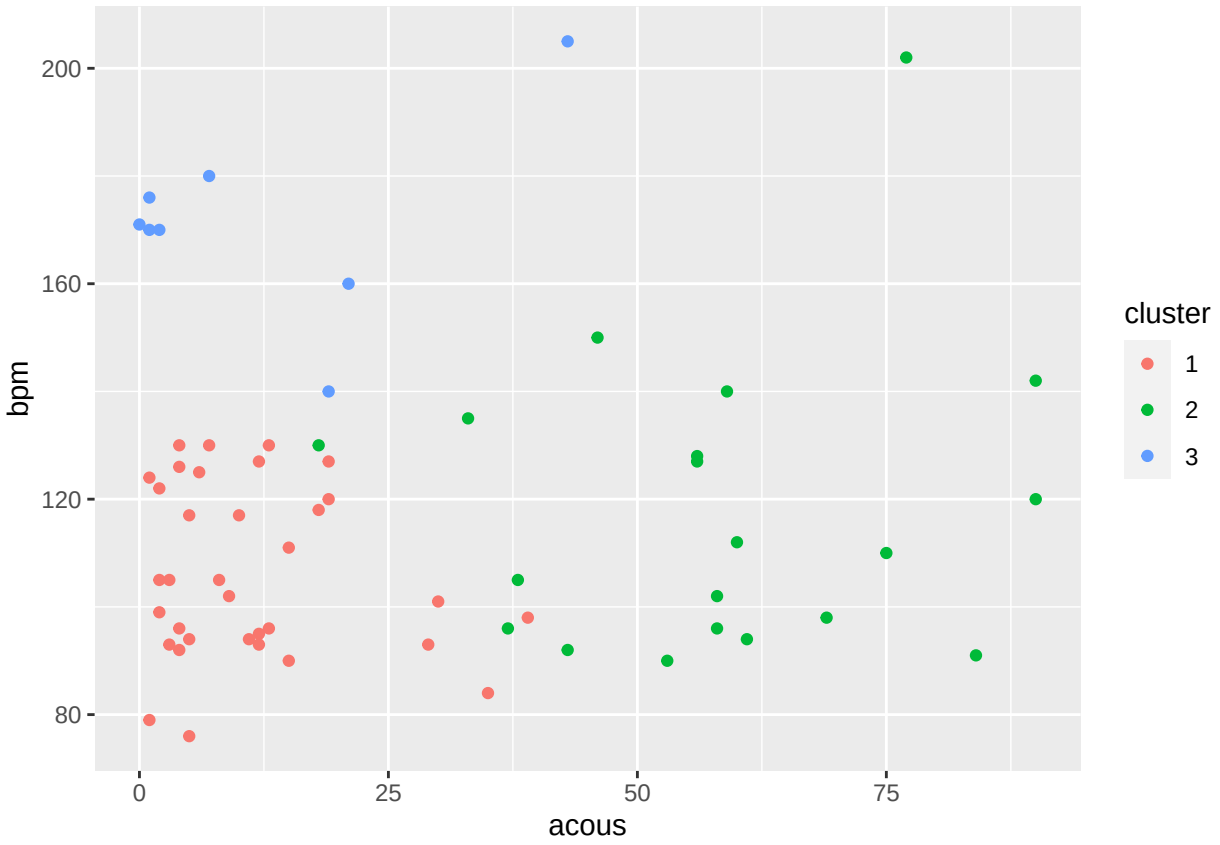
```
plot(pam1, which = 2)
```




```
library(GGally)
ggpairs(final1, columns = 1:6, aes(color=cluster))
```




```
ggplot(final1, aes(x=acous, y=bpm, color= cluster))+geom_point()
```



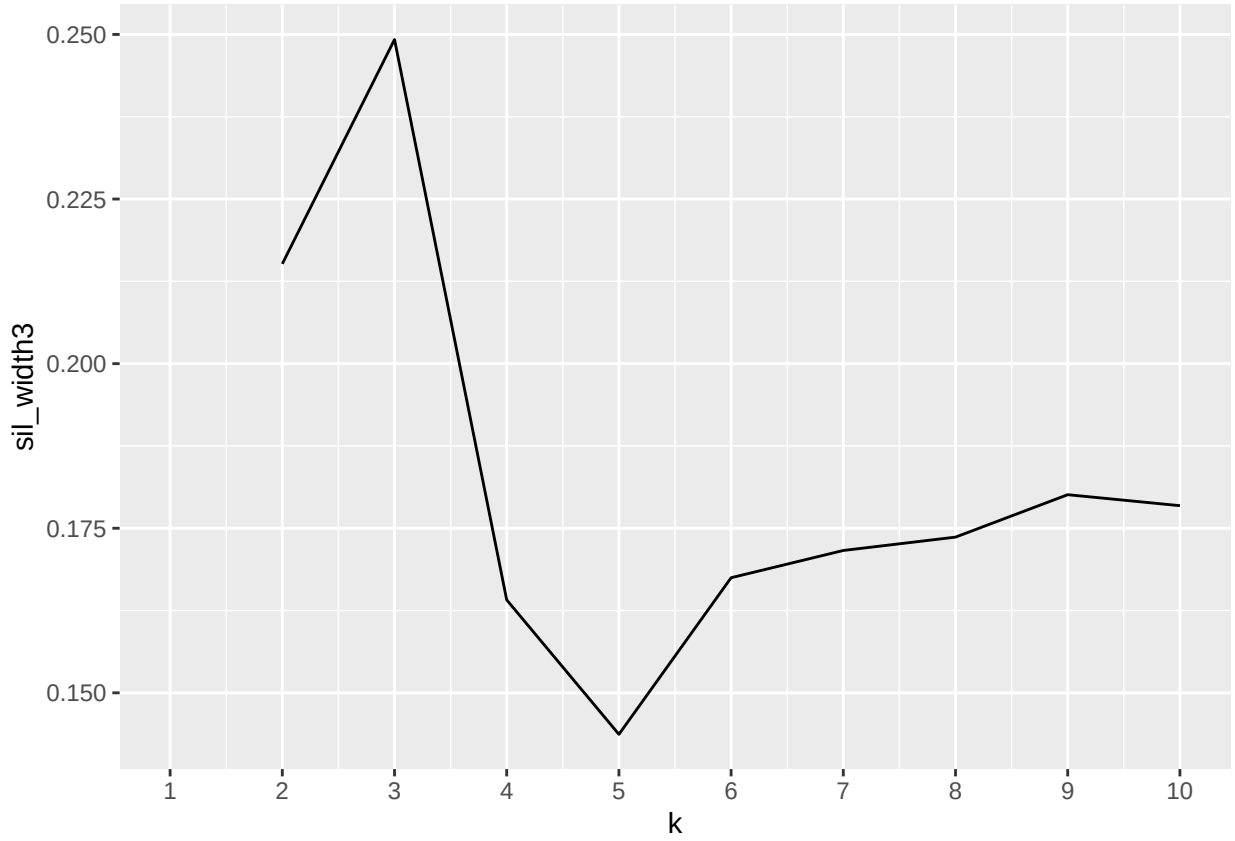

```
clusteringdata1%>%mutate(cluster=pam1$clustering)%>%rename_all(function(x)sub("_", ".", x))%>%
group_by(cluster)%>%mutate(n=n())%>%group_by(cluster,n)%>%
summarize_at(1:6,.funs = list("mean"=mean,"median"=median,"sd"=sd),na.rm=T)%>%
pivot_longer(contains("_"))%>%separate(name,sep="_",into=c("variable","stat"))%>%
pivot_wider(names_from = "variable",values_from="value")%>%arrange
```

```
## # A tibble: 9 x 9
## # Groups:   cluster [3]
##   cluster     n stat    nrgy   bpm   pop  live acous  spch
##     <int> <int> <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1       1    34 mean   68.8  106.  88.9  19.9  11.1  11.3
## 2       1    34 median 68.5  104.  90.5  11     8.5   7.5
## 3       1    34 sd     12.4   15.9  9.57 16.2   9.84  9.85
## 4       2    20 mean   44.8  118   91.4  17    58.0   8.9
## 5       2    20 median 46    111   92    12.5  58     6
## 6       2    20 sd     14.2   27.5  5.30 12.9  19.2   9.41
## 7       3     8 mean   68.4  172.  86.1  12.6  11.8  13
## 8       3     8 median 68    170.  86     9.5   4.5   9.5
## 9       3     8 sd      6.55  18.3  8.22  7.19 15.1  10.5
```

The final step in the project was to cluster numerical variables. The variables chosen to cluster were beats per minute of the song, energy of the song, danceability of the song, liveness of the long, and acousticness of the song. To determine the cluster size, the average silhouette width was plotted. A cluster size of 3 was suggested and chosen. It can be observed that the cluster solution had an average silhouette width of 0.3. Thus, the structure was weak. The clusters did not map nicely to most of the variables. The plot of beats

per minute vs acousticness showed the clusters most distinctly. That plot was observed closely, it could be determined that cluster 3 had higher beats per minute and lower acousticness, cluster 1 had lower beats per minute and lower acousticness, and cluster 2 had higher acousticness and lower beats per minute. To further characterize the clusters, summary statistics was performed on the clusters. Cluster 1 had tracks with the most energy. Cluster 2 had tracks with the highest popularity and acousticness. Cluster 3 had tracks with the highest beats per minute.

```
clusteringdata2 <- BothDataClean %>% select(top.genre, country, trackName, artistName, pop, live, nrgy,
sil_width3 <- vector()
for(i in 2:10){
  pam_fit3 <- clusteringdata2%>% select(top.genre, country, trackName, artistName, pop, live, nrgy) %>%
  sil_width3[i] <- pam_fit3$silinfo$avg.width
}
ggplot() + geom_line(aes(x=1:10, y = sil_width3))+scale_x_continuous(name= "k", breaks = 1:10)
```




```
dat2 <- clusteringdata2 %>% select(top.genre, country, trackName, artistName, pop, live, nrgy) %>% muta
gower1 <- daisy(dat2, metric = "gower")
pam2 <- pam(gower1, k = 2, diss = T)
pam2
```

```
## Medoids:
##      ID
## [1,] 33 33
## [2,] 12 12
## Clustering vector:
```

```
##  [1] 1 1 2 2 1 2 2 2 1 1 2 2 2 2 2 2 2 1 2 1 2 1 2 2 1 1 1 2 2 2 1 2 1 1 1 1 2 2
## [39] 1 2 1 2 1 1 1 1 2 1 1 2 1 1 2 1 2 2 2 1 1 2 2 2
## Objective function:
##     build      swap
## 0.5301503 0.5301503
##
## Available components:
## [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"
## [6] "clusinfo"   "silinfo"    "diss"       "call"
```

```
gower1%>%as.matrix%>%as.data.frame%>%rownames_to_column%>%pivot_longer(-1,values_to="distance")%>%
  filter(rowname!=name)%>%distinct(distance,.keep_all = T)%>%filter(distance%in%c(min(distance),max(dis
```

```
## # A tibble: 2 x 3
##   rowname name  distance
##   <chr>   <chr>    <dbl>
## 1 1       39       0.172
## 2 11      45       0.877
```

```
clusteringdata2%>% slice(1,39)
```

```
##   top.genre   country         trackName   artistName pop live nrgy Year Month
## 1     latin argentina         Con Calma Daddy Yankee  88    6   86    3    25
## 2     latin argentina Que Tire Pa Lante Daddy Yankee  91   12   90   11    18
##    Day EndTimeHour EndTimeMinute msPlayed   X       artist year      added bpm
## 1 2019           2            19    41795 124 Daddy Yankee 2019 1969-12-31  94
## 2 2019           3            05     2773 110 Daddy Yankee 2019 1969-12-31  94
##   dnce dB val dur acous spch
## 1   74 -3  66 193    11    6
## 2   66 -3  71 211     5    4
```

```
clusteringdata2%>% slice(11,45)
```

```
##    top.genre country           trackName      artistName pop live nrgy Year
## 1 anime rock   japan           Yesterday Imagine Dragons  55   40   80    5
## 2 electropop   world everything i wanted   Billie Eilish  98   11   23   11
##   Month  Day EndTimeHour EndTimeMinute msPlayed   X                artist year
## 1    18 2019           1            57   205113 938 Official HIGE DANdism 2019
## 2    28 2019           5            38   233140   6         Billie Eilish 2019
##        added bpm dnce  dB val dur acous spch
## 1 1969-12-31 130   54  -4  56 299     4    4
## 2 1969-12-31 120   70 -14  24 245    90   10
```

```
dat2%>%mutate(cluster=pam2$clustering)%>%group_by(cluster)%>%
filter(!is.na(top.genre))%>%count(top.genre)%>%mutate(prop=n/sum(n))%>%
pivot_wider(-n,names_from=top.genre,values_from=prop,values_fill = list(prop=0))
```

```
## # A tibble: 2 x 27
## # Groups:   cluster [2]
##   cluster `adult standard~ `alternative r&~ `australian pop` `canadian conte~
##     <int>            <dbl>            <dbl>            <dbl>            <dbl>
```

```
## 1       1           0.0690           0.0345           0.0345           0.0690
## 2       2           0                0                0.0303           0
## # ... with 22 more variables: `dance pop` <dbl>, `dfw rap` <dbl>,
## #   electropop <dbl>, latin <dbl>, `modern rock` <dbl>, `nc hip hop` <dbl>,
## #   pop <dbl>, `pop edm` <dbl>, `pop rap` <dbl>, rap <dbl>, `anime rock` <dbl>,
## #   `canadian hip hop` <dbl>, `chicago rap` <dbl>, `chicago soul` <dbl>,
## #   `contemporary country` <dbl>, `electro house` <dbl>, `german hip
## #   hop` <dbl>, `israeli pop` <dbl>, `k-pop` <dbl>, `latin pop` <dbl>, `melodic
## #   rap` <dbl>, `r&b en espanol` <dbl>
```

Clustering by categorical variables was then produced by the "PAM" and "gower" functions. Two clusters were suggested and therefore chosen. After clustering country, artist name, track name, top genre, popularity level, liveness level, and energy level, tracks most similar and most dissimilar were determined. It was found that the tracks most like each other were performed by the same artist. The two tracks found most dissimilar were not performed by the same artist which was expected. To characterize the clusters more, proportions were determined based on the top genre. Cluster 1 had majority of tracks in the Latin genre. Cluster 2 had tracks in multiple genres. Cluster 3 had tracks that were majority among rap genres.

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.