Spotify Music Analysis

Team Fine Arts: Art and Music

12/7/22

Final Project Report

Authors

Kaushik Narasimha Bukkapatnam

Jonathan Olavarria

Katrina Arbogast

Varad Luktuke

Napathsawan Poopaiboon

Sudha Ramakrishnan

Introduction

Over the past few decades, there have been significant changes in the way the media arts are consumed. Everything from the projects that are produced, how they are produced, and the user experience has undergone an astonishing transformation in this new era of fast and limitless digital access. The media firms and individuals that have thrived and sustained are those that realized early on that usage trends and user attraction are the crucial elements to get a competitive edge over the competition.

Spotify is the industry leader in music streaming services, with over 100 million users and a market share of more than 31% among those who subscribe to online music. This motivated us to go further into the Spotify song database in order to uncover intriguing relationships between the songs and the artists as well as to gather insightful knowledge.

Problem Statement

If I were a struggling music producer and looking to produce my next music, what genres of music should I be targeting to produce based on the current market trends and are there any track features that have an impact the popularity of the track.

The approach

A two-fold approach is taken here :

- An overview of the genres and the artists
- An attempt to compute the track popularity based on its features

Data in scope

The data in scope is from the Spotify music platform that was extracted using spotify API. The Spotify Web API provides artist, album, and track data, as well as audio features and analysis, all easily accessible via the R package spotifyr. Each music track has 23 audio attributes recorded, including descriptors like duration, tempo, key, and mode as well as confidence indicators like acousticness, liveness, speechiness, and instrumentalness as well as perceptual indicators like energy, loudness, danceability, and valence (positiveness). The data used for this analysis is a dataset of over 30,000 songs obtained from GitHub. To learn more about the data, visit https://github.com/rfordatascience/tidytuesday/tree/master/data/2020/2020-01-21

Libraries and Data Input

```
suppressPackageStartupMessages(library(knitr))#Used to create a document that is a mixture
suppressPackageStartupMessages(library(tidyverse))#for Data manipulation and Wrangling tas
suppressPackageStartupMessages(library(dplyr))#for Data manipulation and Wrangling tasks
suppressPackageStartupMessages(library(ggplot2))#for effective visualizations
suppressPackageStartupMessages(library(gridExtra))#for effective visualizations
suppressPackageStartupMessages(library(reshape2))#used to melt the correlation matrix
```

Reading in the data

```
#Ceating a "spotify_songs" dataframe by reading the data
spotify_songs <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytu
```

```
Rows: 32833 Columns: 23
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (10): track_id, track_name, track_artist, track_album_id, track_album_na...
dbl (13): track_popularity, danceability, energy, key, loudness, mode, speec...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
spotify_songs
```

```
# A tibble: 32,833 x 23
   track_id      track~1 track~2 track~3 track~4 track~5 track~6 playl~7 playl~8
   <chr>         <chr>   <chr>     <dbl> <chr>   <chr>   <chr>   <chr>   <chr>
 1 6f807x0ima9a~ I Don'~ Ed She~      66 2oCs0D~ I Don'~ 2019-0~ Pop Re~ 37i9dQ~
 2 0r7CVbZTWZgb~ Memori~ Maroon~      67 63rPS0~ Memori~ 2019-1~ Pop Re~ 37i9dQ~
 3 1z1Hg7Vb0AhH~ All th~ Zara L~      70 1HoSmj~ All th~ 2019-0~ Pop Re~ 37i9dQ~
 4 75FpbthrwQmz~ Call Y~ The Ch~      60 1nqYsO~ Call Y~ 2019-0~ Pop Re~ 37i9dQ~
 5 1e8PAfcKUYoK~ Someon~ Lewis ~      69 7m7vv9~ Someon~ 2019-0~ Pop Re~ 37i9dQ~
 6 7fvUMiyapMsR~ Beauti~ Ed She~      67 2yiy9c~ Beauti~ 2019-0~ Pop Re~ 37i9dQ~
 7 2OAylPUDDfwR~ Never ~ Katy P~      62 7INHYS~ Never ~ 2019-0~ Pop Re~ 37i9dQ~
 8 6b1RNvAcJjQH~ Post M~ Sam Fe~      69 6703SR~ Post M~ 2019-0~ Pop Re~ 37i9dQ~
 9 7bF6tCO3gFb8~ Tough ~ Avicii       68 7CvAfG~ Tough ~ 2019-0~ Pop Re~ 37i9dQ~
10 1IXGILkPm0tO~ If I C~ Shawn ~      67 4Qxzbf~ If I C~ 2019-0~ Pop Re~ 37i9dQ~
# ... with 32,823 more rows, 14 more variables: playlist_genre <chr>,
#   playlist_subgenre <chr>, danceability <dbl>, energy <dbl>, key <dbl>,
#   loudness <dbl>, mode <dbl>, speechiness <dbl>, acousticness <dbl>,
#   instrumentalness <dbl>, liveness <dbl>, valence <dbl>, tempo <dbl>,
#   duration_ms <dbl>, and abbreviated variable names 1: track_name,
#   2: track_artist, 3: track_popularity, 4: track_album_id,
#   5: track_album_name, 6: track_album_release_date, 7: playlist_name, ...
# i Use `print(n = ...)` to see more rows, and `colnames()` to see all variable names
```

```
#getting the dimensions of the dataset
dim(spotify_songs)
```

```
[1] 32833    23
```

```
#take a quick look
summary(spotify_songs)
```

```
  track_id          track_name        track_artist       track_popularity
Length:32833       Length:32833       Length:32833       Min.   :  0.00
Class :character   Class :character   Class :character   1st Qu.: 24.00
Mode  :character   Mode  :character   Mode  :character   Median : 45.00
                                                         Mean   : 42.48
                                                         3rd Qu.: 62.00
                                                         Max.   :100.00
track_album_id     track_album_name   track_album_release_date
Length:32833       Length:32833       Length:32833
Class :character   Class :character   Class :character
Mode  :character   Mode  :character   Mode  :character


playlist_name      playlist_id        playlist_genre     playlist_subgenre
Length:32833       Length:32833       Length:32833       Length:32833
Class :character   Class :character   Class :character   Class :character
Mode  :character   Mode  :character   Mode  :character   Mode  :character


 danceability        energy               key            loudness
Min.   :0.0000   Min.   :0.000175   Min.   : 0.000   Min.   :-46.448
1st Qu.:0.5630   1st Qu.:0.581000   1st Qu.: 2.000   1st Qu.: -8.171
Median :0.6720   Median :0.721000   Median : 6.000   Median : -6.166
Mean   :0.6548   Mean   :0.698619   Mean   : 5.374   Mean   : -6.720
3rd Qu.:0.7610   3rd Qu.:0.840000   3rd Qu.: 9.000   3rd Qu.: -4.645
Max.   :0.9830   Max.   :1.000000   Max.   :11.000   Max.   :  1.275
     mode         speechiness      acousticness    instrumentalness
Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000000
1st Qu.:0.0000   1st Qu.:0.0410   1st Qu.:0.0151   1st Qu.:0.0000000
Median :1.0000   Median :0.0625   Median :0.0804   Median :0.0000161
Mean   :0.5657   Mean   :0.1071   Mean   :0.1753   Mean   :0.0847472
3rd Qu.:1.0000   3rd Qu.:0.1320   3rd Qu.:0.2550   3rd Qu.:0.0048300
Max.   :1.0000   Max.   :0.9180   Max.   :0.9940   Max.   :0.9940000
   liveness         valence           tempo         duration_ms
```

```
 Min.   :0.0000   Min.   :0.0000   Min.   :  0.00   Min.   :  4000
 1st Qu.:0.0927   1st Qu.:0.3310   1st Qu.: 99.96   1st Qu.:187819
 Median :0.1270   Median :0.5120   Median :121.98   Median :216000
 Mean   :0.1902   Mean   :0.5106   Mean   :120.88   Mean   :225800
 3rd Qu.:0.2480   3rd Qu.:0.6930   3rd Qu.:133.92   3rd Qu.:253585
 Max.   :0.9960   Max.   :0.9910   Max.   :239.44   Max.   :517810
```

Observation: There are a total of 32833 rows and 23 columns in the data

Lets look at what each attribute denotes-

```
#Creating a dataframe to hold the information about the attributes of the data
url <- 'https://raw.githubusercontent.com/vpcincin/DataWrangling/main/Data_Dictionary.csv'
spotify_attributes <- readr::read_csv(url)
```

```
Rows: 23 Columns: 3
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (3): variable, class, description

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
print(spotify_attributes, n = 23)
```

```
# A tibble: 23 x 3
   variable                 class     description
   <chr>                    <chr>     <chr>
 1 track_id                 character "Song unique ID"
 2 track_name               character "Song Name"
 3 track_artist             character "Song Artist"
 4 track_popularity         double    "Song Popularity (0-100) where higher is ~
 5 track_album_id           character "Album unique ID"
 6 track_album_name         character "Song album name"
 7 track_album_release_date character "Date when album released"
 8 playlist_name            character "Name of playlist"
 9 playlist_id              character "Playlist ID"
10 playlist_genre           character "Playlist genre"
11 playlist_subgenre        character "Playlist subgenre"
```

```
12 danceability            double    "Danceability describes how suitable a tr~
13 energy                  double    "Energy is a measure from 0.0 to 1.0 and ~
14 key                     double    "The estimated overall key of the track. ~
15 loudness                double    "The overall loudness of a track in decib~
16 mode                    double    "Mode indicates the modality (major or mi~
17 speechiness             double    "Speechiness detects the presence of spok~
18 acousticness            double    "A confidence measure from 0.0 to 1.0 of ~
19 instrumentalness        double    "Predicts whether a track contains no voc~
20 liveness                double    "Detects the presence of an audience in t~
21 valence                 double    "A measure from 0.0 to 1.0 describing the~
22 tempo                   double    "The overall estimated tempo of a track i~
23 duration_ms             double    "Duration of song in milliseconds"
```

We get the description of each variable and its class for understanding our data

Checking for null values

```
#getting the count of total null values in data
sum(is.na(spotify_songs))
```

```
[1] 15
```

```
#checking the null values by columns
colSums(is.na(spotify_songs))
```

```
               track_id              track_name            track_artist
                      0                       5                       5
      track_popularity          track_album_id        track_album_name
                      0                       0                       5
track_album_release_date           playlist_name             playlist_id
                      0                       0                       0
         playlist_genre       playlist_subgenre            danceability
                      0                       0                       0
                 energy                     key                loudness
                      0                       0                       0
                   mode             speechiness            acousticness
                      0                       0                       0
       instrumentalness                liveness                 valence
                      0                       0                       0
                  tempo             duration_ms
                      0                       0
```

Observation: We see a total of 15 null values in the dataset. When checked deeper on the column level, we see that there are a total of 5 missing values for each of the columns- 'track_name', 'track_artist', 'track_album_name'.

Checking for variable types

```
# checking variable types for consistencies
str(spotify_songs[])
```

```
tibble [32,833 x 23] (S3: tbl_df/tbl/data.frame)
 $ track_id                : chr [1:32833] "6f807x0ima9a1j3VPbc7VN" "0r7CVbZTWZgbTCYdfa2P31"
 $ track_name              : chr [1:32833] "I Don't Care (with Justin Bieber) - Loud Luxury
 $ track_artist            : chr [1:32833] "Ed Sheeran" "Maroon 5" "Zara Larsson" "The Chains
 $ track_popularity        : num [1:32833] 66 67 70 60 69 67 62 69 68 67 ...
 $ track_album_id          : chr [1:32833] "2oCs0DGTsRO98Gh5ZSl2Cx" "63rPSO264uRjW1X5E6cWv6"
 $ track_album_name        : chr [1:32833] "I Don't Care (with Justin Bieber) [Loud Luxury Re
 $ track_album_release_date: chr [1:32833] "2019-06-14" "2019-12-13" "2019-07-05" "2019-07-19
 $ playlist_name           : chr [1:32833] "Pop Remix" "Pop Remix" "Pop Remix" "Pop Remix" .
 $ playlist_id             : chr [1:32833] "37i9dQZF1DXcZDD7cfEKhW" "37i9dQZF1DXcZDD7cfEKhW"
 $ playlist_genre          : chr [1:32833] "pop" "pop" "pop" "pop" ...
 $ playlist_subgenre       : chr [1:32833] "dance pop" "dance pop" "dance pop" "dance pop" .
 $ danceability            : num [1:32833] 0.748 0.726 0.675 0.718 0.65 0.675 0.449 0.542 0.5
 $ energy                  : num [1:32833] 0.916 0.815 0.931 0.93 0.833 0.919 0.856 0.903 0.9
 $ key                     : num [1:32833] 6 11 1 7 1 8 5 4 8 2 ...
 $ loudness                : num [1:32833] -2.63 -4.97 -3.43 -3.78 -4.67 ...
 $ mode                    : num [1:32833] 1 1 0 1 1 1 0 0 1 1 ...
 $ speechiness             : num [1:32833] 0.0583 0.0373 0.0742 0.102 0.0359 0.127 0.0623 0.0
 $ acousticness            : num [1:32833] 0.102 0.0724 0.0794 0.0287 0.0803 0.0799 0.187 0.0
 $ instrumentalness        : num [1:32833] 0.00 4.21e-03 2.33e-05 9.43e-06 0.00 0.00 0.00 4.8
 $ liveness                : num [1:32833] 0.0653 0.357 0.11 0.204 0.0833 0.143 0.176 0.111 0
 $ valence                 : num [1:32833] 0.518 0.693 0.613 0.277 0.725 0.585 0.152 0.367 0
 $ tempo                   : num [1:32833] 122 100 124 122 124 ...
 $ duration_ms             : num [1:32833] 194754 162600 176616 169093 189052 ...
```

Observations: The attribute 'mode' currently has a numeric field, however it is supposed to be a Boolean/factor variable, as it has values{0,1} The attribute 'track_album_release_date' is currently a character column but it is supposed to be a field with date values

It is important for us to change the type of these variables as they may be important for our analysis.

Cleaning the data (dealing with null values and modifying datatypes)

We previously inspected a total of 5 missing values each for the 'track_name', 'track_artist' and 'track_album_name' attributes. Let us impute these missing values by a constant value 'unknown' as these parameters would not be impacting our analysis since we are not focusing on these three attributes and moreover it is a very small fraction of the dataset.

Also, we will be changing the datatypes of the two attributes to the required type that we observed to be incorrect

```
#Missing Value Treatment
spotify_songs$track_artist[is.na(spotify_songs$track_artist)] <- 'unknown'
spotify_songs$track_album_name[is.na(spotify_songs$track_album_name)] <- 'unknown'
spotify_songs$track_name[is.na(spotify_songs$track_name)] <- 'unknown'
#check
sum(is.na(spotify_songs))
```

```
[1] 0
```

```
#Modyfying Data types
spotify_songs$mode <- as.factor(spotify_songs$mode)
spotify_songs$track_album_release_date <- as.Date(spotify_songs$track_album_release_date)
#check
class(spotify_songs$mode)
```

```
[1] "factor"
```

```
class(spotify_songs$track_album_release_date)
```

```
[1] "Date"
```

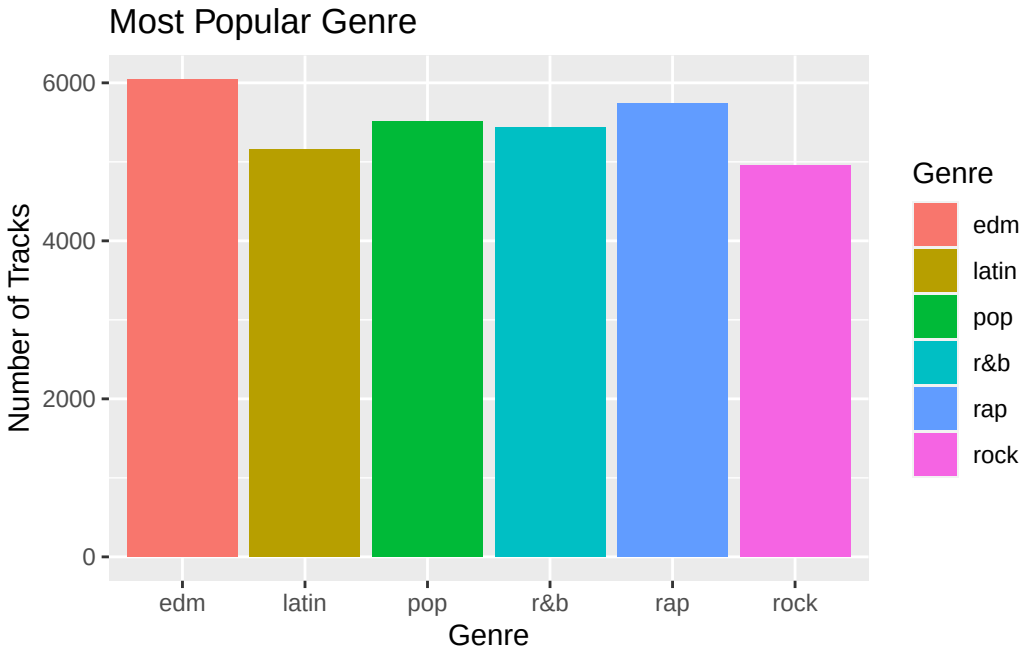
Exploratory Data Analysis

Firstly, we start by looking at the number of tracks in each genre to find out which genre has most number of releases.

```
spotify_songs %>% group_by(Genre = playlist_genre) %>%
  summarise(No_of_tracks = n()) %>% arrange(desc(No_of_tracks)) %>% knitr::kable()
```

Genre	No_of_tracks
edm	6043
rap	5746
pop	5507
r&b	5431
latin	5155
rock	4951

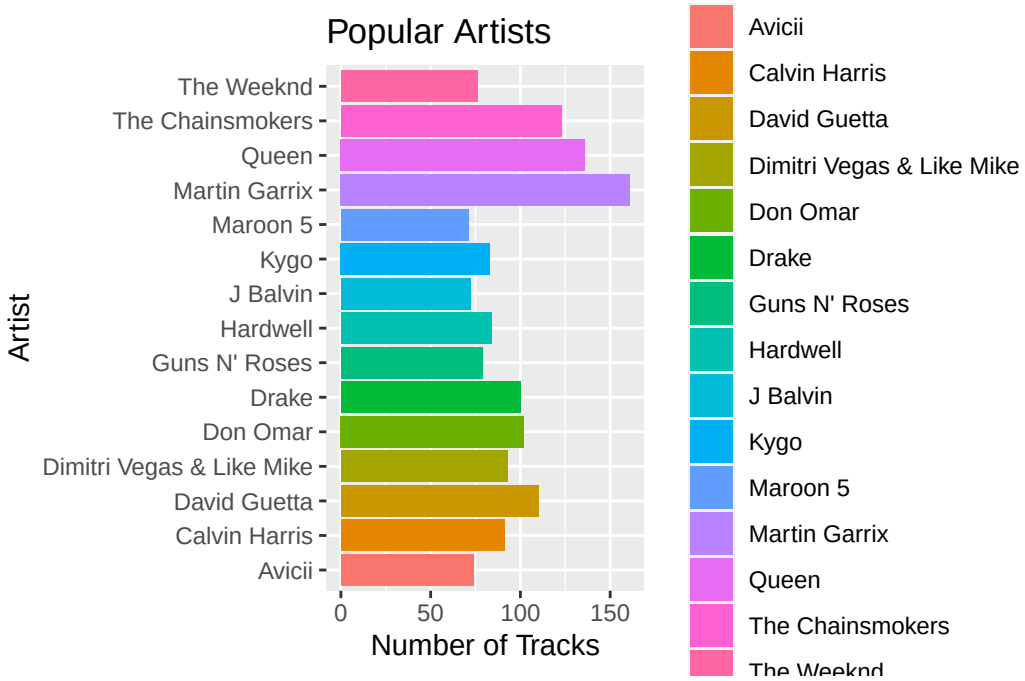
```
spotify_genres <- spotify_songs %>% group_by(Genre = playlist_genre) %>%
  summarise(No_of_tracks = n()) %>% arrange(desc(No_of_tracks)) %>%
  ggplot(aes(x = Genre, y = No_of_tracks, fill = Genre)) +
        geom_bar(stat = "identity") +
        labs(title = "Most Popular Genre", x = "Genre", y = "Number of Tracks")

spotify_genres
```



Observation: EDM songs quantify the most part of the data followed by rap and then pop depicting the listening trends for the respective genres. This indicates that most of the users listen to EDM music followed by Rap music.

Then we look at the number of songs released by each artist:

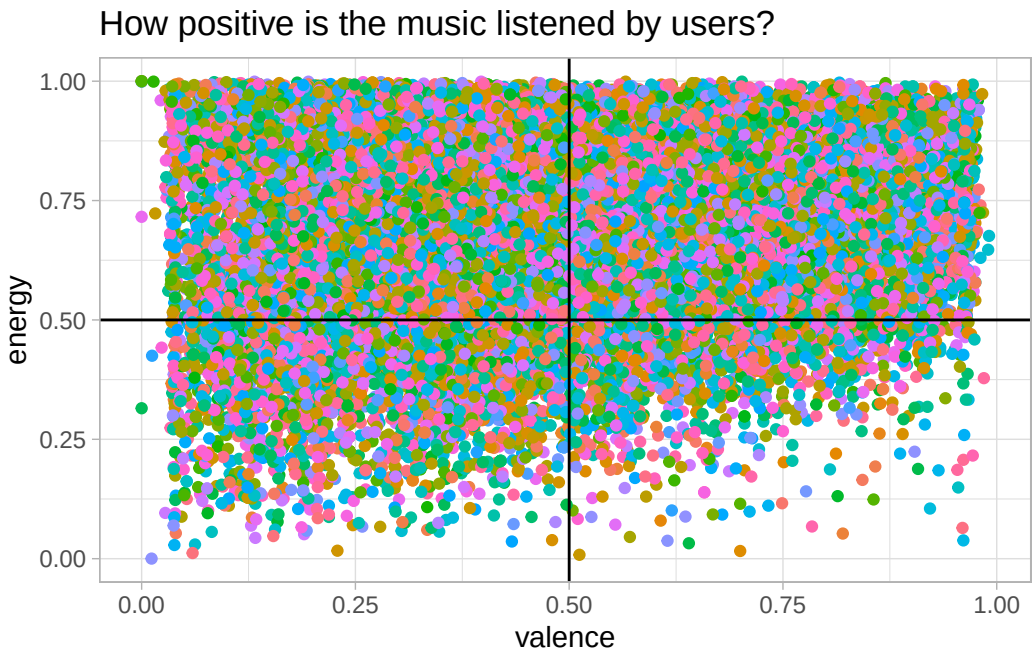
```r
# artists with most releases
most_releases <- spotify_songs %>% group_by(Artist = track_artist) %>%
  summarise(No_of_tracks = n()) %>%
  arrange(desc(No_of_tracks)) %>%
  top_n(15, wt = No_of_tracks) %>%
  ggplot(aes(x = Artist, y = No_of_tracks, fill = Artist)) +
        geom_bar(stat = "identity") +
        coord_flip() +
  labs(title = "Popular Artists", x = "Artist", y = "Number of Tracks")

most_releases
```



Observation: With more than 150 songs Martin Garrix has been the most listened artist in this span followed by the band Queen with around 130 songs.

One of the main parameters of a song would be valence. It is a parameter to indicate 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). We look into the general trend of musical positiveness of tracks by plotting them against energy.

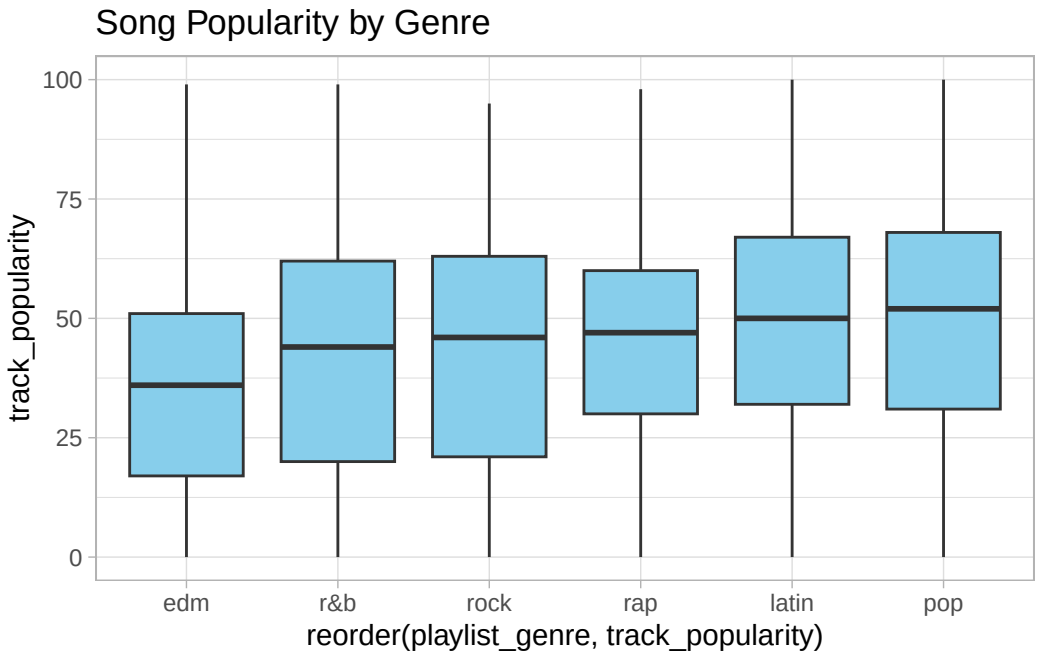
```r
valence <- spotify_songs %>%
  ggplot(aes(x= valence, y= energy, color= track_name)) +
  geom_jitter(show.legend = FALSE) +
  geom_vline(xintercept = 0.5) +
  geom_hline(yintercept = 0.5) +
  scale_x_continuous(breaks= seq(0, 1, 0.25)) +
  scale_y_continuous(breaks= seq(0, 1, 0.25)) +
  labs(title= "How positive is the music listened by users?") +
  theme_light()

valence
```



Observation: As seen from the plot, most of the songs are on the left side, which means they are sad, depressed, or angry. However, many of those tracks have high energy which can be interpreted as fast, loud, and noisy. Yet, there are still many tracks on the upper right, which represents the happy and positive.

Then we compared the popularity of songs by grouping them by genres to find out which genre would be more likely to gain popularity:

```
Pop_acc_genre <-spotify_songs%>%
  ggplot(aes(x=reorder(playlist_genre,track_popularity),y=track_popularity))+
  geom_boxplot(fill="skyblue")+
  ggtitle("Song Popularity by Genre")+
  theme_light()
Pop_acc_genre
```

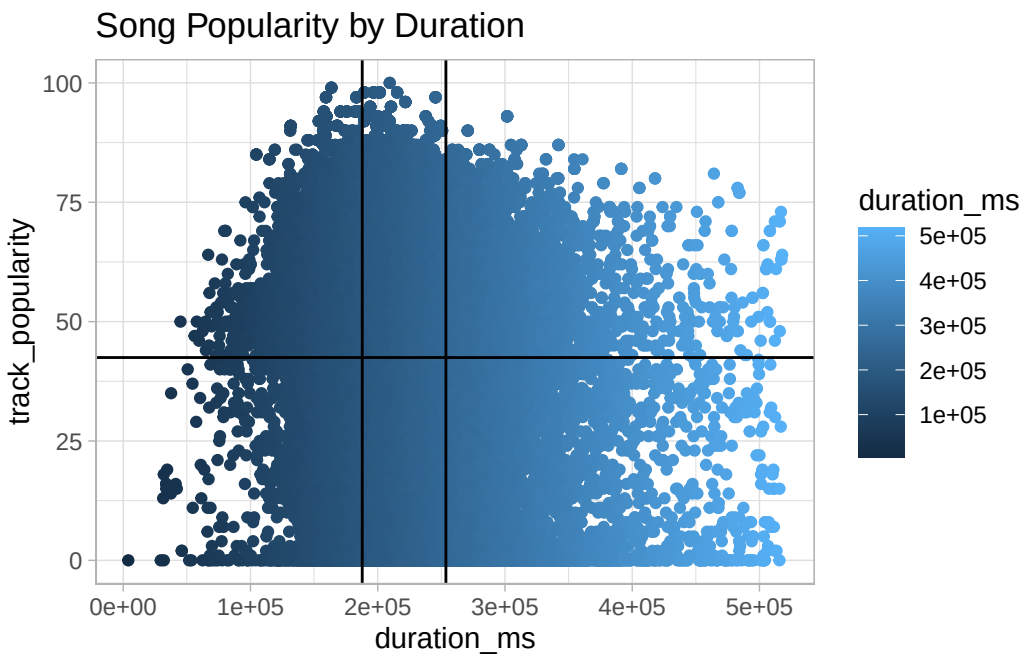


Observation: The average song popularity index of pop songs is higher compared to the rest of the genres. Latin songs seconds the list.

Next, we checked how duration of songs affected the popularity:

```
Popularity_duration <-spotify_songs%>%
  ggplot(aes(x=duration_ms,
             y=track_popularity, color = duration_ms))+
  geom_point()+
  geom_vline(xintercept = quantile(spotify_songs$duration_ms, 0.25)) +
  geom_vline(xintercept = quantile(spotify_songs$duration_ms, 0.75)) +
  geom_hline(yintercept = mean(spotify_songs$track_popularity)) +
  ggtitle("Song Popularity by Duration")+
  theme_light()
```

```
Popularity_duration
```

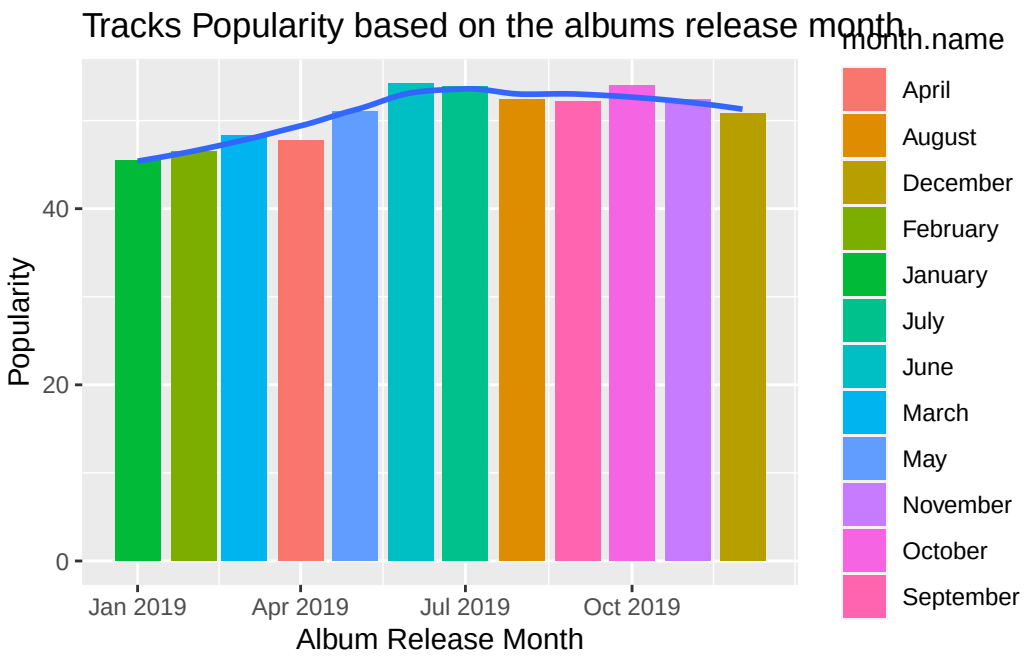


Observation: Songs that are in the upper 50% of popularity tend to localize around the center of song duration. We can see that between 187819 ms and 253585 ms most songs tend to accumulate.

Then we checked the correlation between track popularity and release date to find out whether the popularity of song is getting affected by the month it is released in:

```
top_dates_to_release <- spotify_songs %>%
  group_by(month = lubridate::floor_date(track_album_release_date, "month")) %>%
  summarise(avg_pop = mean(track_popularity))%>%
  filter(month >= '2019-01-01' & month < '2020-01-01')%>%
  ggplot(aes(x = month, y = avg_pop, fill = month.name)) +
        geom_bar(stat = "identity") +
        geom_smooth(se = FALSE, fill="blue")+
  labs(title = "Tracks Popularity based on the albums release month",
       x = 'Album Release Month',
       y = 'Popularity')

top_dates_to_release
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

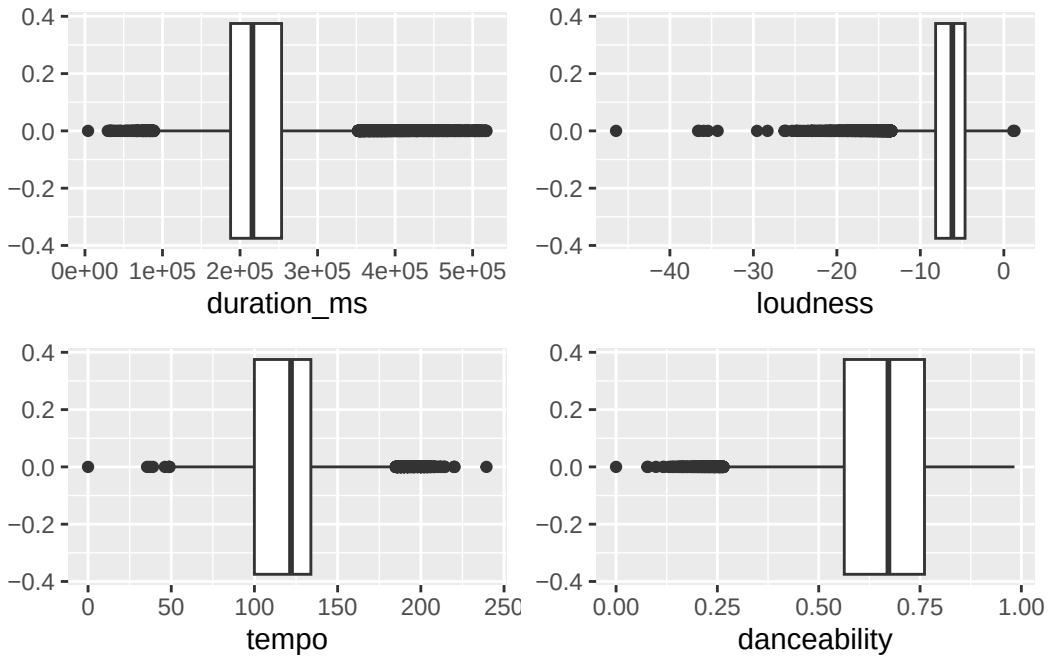


Observation: The average track popularity for a given release month (in the year 2019), shows higher averages in months June through December compared to that of months January through May.

Checking the outliers

```
duration_boxplot <- spotify_songs %>%
  ggplot()+
  geom_boxplot(aes(duration_ms))

loudness_boxplot <- spotify_songs %>%
  ggplot() +
  geom_boxplot(aes(loudness))

tempo_boxplot <- spotify_songs %>%
  ggplot() +
  geom_boxplot(aes(tempo))

danceability_boxplot <- spotify_songs %>%
```

```
  ggplot() +
  geom_boxplot(aes(danceability))

grid.arrange(duration_boxplot, loudness_boxplot, tempo_boxplot, danceability_boxplot, nrow
```




```
dq1 <- quantile(spotify_songs$duration_ms, 0.25)
dq3 <- quantile(spotify_songs$duration_ms, 0.75)

lq1 <- quantile(spotify_songs$loudness, 0.25)
lq3 <- quantile(spotify_songs$loudness, 0.75)

tq1 <- quantile(spotify_songs$tempo, 0.25)
tq3 <- quantile(spotify_songs$tempo, 0.75)

danceq1 <- quantile(spotify_songs$danceability, 0.25)
danceq3 <- quantile(spotify_songs$danceability, 0.75)
```

Observation: For analysis we choose to look at these four variables for possible outliers, each of them indeed do have outliers. Duration, loudness, and tempo have outliers at both extremes, while danceability only has lower bound outliers.

The First and Third Quartiles for the variables are:

Quantile	Duration (ms)	Loudness	Tempo	Danceability
25%	187819	-8.171	99.96	0.563
75%	253585	-4.645	133.918	0.761

Plotting correlation heatmap

We then proceeded to check the correlations between our key attributes in order to examine if any attribute in general has more effect on the track popularity

```
#In order to check for correlations, we need to first segregate the numerical attributes
spotify_songs_num <- spotify_songs[,c(4,12,13,15,17,18,19,20,21,22,23)]
summary(spotify_songs_num)
```

```
 track_popularity  danceability        energy            loudness
 Min.   :  0.00   Min.   :0.0000   Min.   :0.000175   Min.   :-46.448
 1st Qu.: 24.00   1st Qu.:0.5630   1st Qu.:0.581000   1st Qu.: -8.171
 Median : 45.00   Median :0.6720   Median :0.721000   Median : -6.166
 Mean   : 42.48   Mean   :0.6548   Mean   :0.698619   Mean   : -6.720
 3rd Qu.: 62.00   3rd Qu.:0.7610   3rd Qu.:0.840000   3rd Qu.: -4.645
 Max.   :100.00   Max.   :0.9830   Max.   :1.000000   Max.   :  1.275
  speechiness      acousticness    instrumentalness       liveness
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000000   Min.   :0.0000
 1st Qu.:0.0410   1st Qu.:0.0151   1st Qu.:0.0000000   1st Qu.:0.0927
 Median :0.0625   Median :0.0804   Median :0.0000161   Median :0.1270
 Mean   :0.1071   Mean   :0.1753   Mean   :0.0847472   Mean   :0.1902
 3rd Qu.:0.1320   3rd Qu.:0.2550   3rd Qu.:0.0048300   3rd Qu.:0.2480
 Max.   :0.9180   Max.   :0.9940   Max.   :0.9940000   Max.   :0.9960
    valence           tempo         duration_ms
 Min.   :0.0000   Min.   :  0.00   Min.   :  4000
 1st Qu.:0.3310   1st Qu.: 99.96   1st Qu.:187819
 Median :0.5120   Median :121.98   Median :216000
 Mean   :0.5106   Mean   :120.88   Mean   :225800
 3rd Qu.:0.6930   3rd Qu.:133.92   3rd Qu.:253585
 Max.   :0.9910   Max.   :239.44   Max.   :517810
```

```
head(spotify_songs_num, 5)
```

```
# A tibble: 5 x 11
  track_p~1 dance~2 energy loudn~3 speec~4 acous~5 instr~6 liven~7 valence tempo
```

```
      <dbl>   <dbl>  <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>
1        66   0.748  0.916   -2.63  0.0583  0.102  0        0.0653   0.518  122.
2        67   0.726  0.815   -4.97  0.0373  0.0724 4.21e-3  0.357    0.693  100.
3        70   0.675  0.931   -3.43  0.0742  0.0794 2.33e-5  0.11     0.613  124.
4        60   0.718  0.93    -3.78  0.102   0.0287 9.43e-6  0.204    0.277  122.
5        69   0.65   0.833   -4.67  0.0359  0.0803 0        0.0833   0.725  124.
# ... with 1 more variable: duration_ms <dbl>, and abbreviated variable names
#   1: track_popularity, 2: danceability, 3: loudness, 4: speechiness,
#   5: acousticness, 6: instrumentalness, 7: liveness
# i Use `colnames()` to see all variable names
```

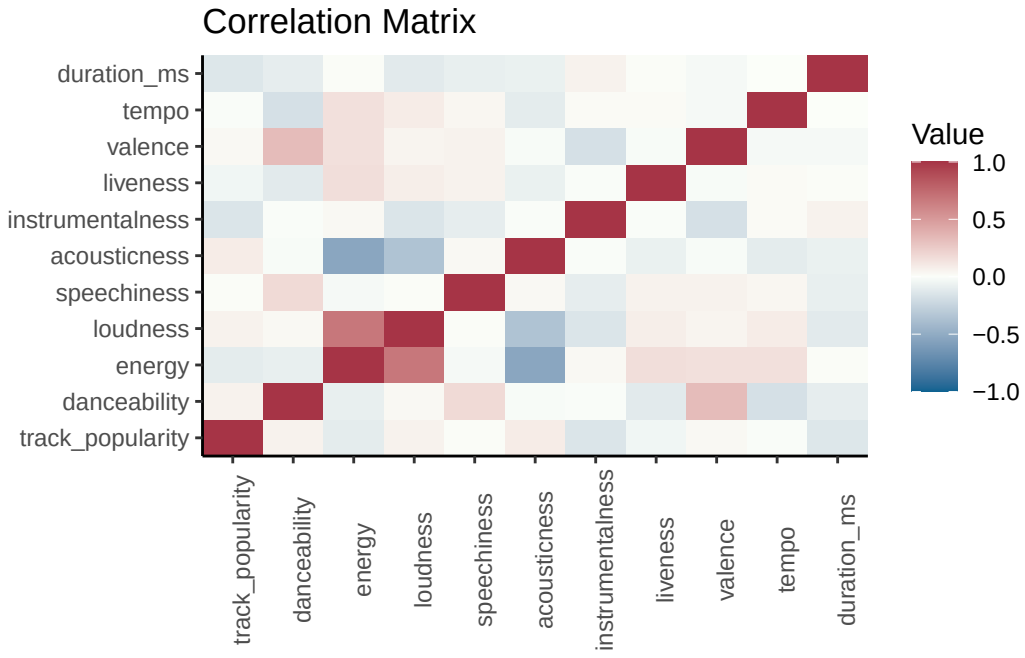
```
#creating a correlation matrix restricting upto two decimal places
cormat <- round(cor(spotify_songs_num),2)
head(cormat)
```

```
                  track_popularity danceability energy loudness speechiness
track_popularity              1.00         0.06  -0.11     0.06        0.01
danceability                  0.06         1.00  -0.09     0.03        0.18
energy                       -0.11        -0.09   1.00     0.68       -0.03
loudness                      0.06         0.03   0.68     1.00        0.01
speechiness                   0.01         0.18  -0.03     0.01        1.00
acousticness                  0.09        -0.02  -0.54    -0.36        0.03
                  acousticness instrumentalness liveness valence tempo
track_popularity          0.09            -0.15    -0.05    0.03 -0.01
danceability             -0.02            -0.01    -0.12    0.33 -0.18
energy                   -0.54             0.03     0.16    0.15  0.15
loudness                 -0.36            -0.15     0.08    0.05  0.09
speechiness               0.03            -0.10     0.06    0.06  0.04
acousticness              1.00            -0.01    -0.08   -0.02 -0.11
                  duration_ms
track_popularity        -0.14
danceability            -0.10
energy                   0.01
loudness                -0.12
speechiness             -0.09
acousticness            -0.08
```

```
#melting the correlation matrix for plotting purpose
melted_cormat <- melt(cormat)
head(melted_cormat)
```

```
              Var1             Var2 value
1 track_popularity track_popularity  1.00
2     danceability track_popularity  0.06
3           energy track_popularity -0.11
4         loudness track_popularity  0.06
5      speechiness track_popularity  0.01
6     acousticness track_popularity  0.09
```

```
#plotting the correlation matrix by using gradient fill
melted_cormat %>%
 ggplot(aes(Var1, Var2, fill=value)) +
 geom_tile() +
 labs(x = NULL, y = NULL, fill = "Value", title="Correlation Matrix") + scale_fill_gradien
 #geom_text() +
 theme_classic() +
 scale_x_discrete(expand=c(0,0)) +
 scale_y_discrete(expand=c(0,0)) +
 theme(axis.text.x = element_text(angle = 90))
```



Observation: From the correlation matrix we can see that there is no single attribute/feature of a track having a high correlation with the popularity of the track. There is a very moderate correlation between the track popularity and the acousticness of the track but since the

correlation is weak, we cannot have the acoustics as a measure for determining the popularity of a track.

Linear Regression Model

Creating dummy variable for Playlist Genre

```
spotify_songs$genre_rap <- ifelse(spotify_songs$playlist_genre == 'rap', 1, 0)
spotify_songs$genre_rnb <- ifelse(spotify_songs$playlist_genre == 'r&b', 1, 0)
spotify_songs$genre_pop <- ifelse(spotify_songs$playlist_genre == 'pop', 1, 0)
spotify_songs$genre_latin <- ifelse(spotify_songs$playlist_genre == 'latin', 1, 0)
spotify_songs$genre_edm <- ifelse(spotify_songs$playlist_genre == 'edm', 1, 0)
```

Build Linear Regression Model

Y is track popularity.

Xs are danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence, tempo, duration_ms, genre_rap, genre_rnb, genre_pop, genre_latin, and genre_edm

```
model <- lm(track_popularity ~ danceability + energy + loudness + speechiness + acousticne

#view regression model output
summary(model)
```

```
Call:
lm(formula = track_popularity ~ danceability + energy + loudness +
    speechiness + acousticness + instrumentalness + liveness +
    valence + tempo + duration_ms + genre_rap + genre_rnb + genre_pop +
    genre_latin + genre_edm, data = spotify_songs)

Residuals:
    Min      1Q  Median      3Q     Max
-61.433 -17.261   3.199  18.627  67.271

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       7.993e+01  1.682e+00  47.533  < 2e-16 ***
danceability      1.051e+01  1.137e+00   9.240  < 2e-16 ***
```

```
energy           -2.938e+01  1.224e+00 -24.009  < 2e-16 ***
loudness          1.626e+00  6.479e-02  25.088  < 2e-16 ***
speechiness      -2.161e+00  1.477e+00  -1.463 0.143358
acousticness      2.894e+00  7.303e-01   3.962 7.44e-05 ***
instrumentalness -9.095e+00  6.463e-01 -14.072  < 2e-16 ***
liveness         -3.289e+00  8.744e-01  -3.761 0.000169 ***
valence          -8.947e-01  6.658e-01  -1.344 0.178992
tempo             2.371e-02  5.074e-03   4.673 2.98e-06 ***
duration_ms      -4.564e-05  2.284e-06 -19.981  < 2e-16 ***
genre_rap        -5.138e+00  5.623e-01  -9.138  < 2e-16 ***
genre_rnb        -6.649e+00  5.194e-01 -12.800  < 2e-16 ***
genre_pop         2.803e-01  4.998e-01   0.561 0.574914
genre_latin      -1.166e+00  5.307e-01  -2.197 0.028045 *
genre_edm        -9.408e+00  5.277e-01 -17.827  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 23.83 on 32817 degrees of freedom
Multiple R-squared:  0.09053,   Adjusted R-squared:  0.09012
F-statistic: 217.8 on 15 and 32817 DF,  p-value: < 2.2e-16
```

Observation: A linear regression model describes the relationship between a *dependent variable*, y, and one or more *independent variables*, X. With 32817 degrees of freedom, we found that the f value is 217.8 and p value was found to be less than 2.2e-16. The regression model has an extremely low adjusted R-squared value suggesting that the input variables are not good indicators of song popularity. This makes sense when referencing the correlation heat map as we saw very weak correlations between all the variables and song popularity.

Conclusion

We would like to start off by acknowledging some potential sources of bias in our analysis. First of which is that our data only covers songs from six genres, it is common knowledge that there are far more than six main genres in music and therefore we must preface our conlusion's with the fact that they only apply to songs in the genres of pop, edm, latin, R&B, rock, and rap. We cannot extrapolate these findings to songs in other genres. As far as our personal biases for the research, our team went in to this project thinking that the trends and correlations within the music would be very strongly defined. However, as the project matured and our analysis became more defined we realized that the trends may not be as significant as we initially thought they would be.

In our exploratory analysis we investigated the distribution of our data as well as interactions between some of the variables. The data was relatively uniformly distributed accross the 6

genres. However, we did find that at the artist level we had a few artists with far more songs in the data than some of the others these were Martin Garrix, Queen, and The Chainsmokers. This means that across the playlists sampled for to create our data set these artists were the most popular. In addition to this we looked at the interactions between variables. We found that there is a "sweet spot" for song duration meaning that songs that lie within 187819 ms and 253585 ms tend to be more popular than songs that lie outside of this range. We also found that songs that came from albums in June and July are considerably more popular than songs that came from albums released earlier in the year. Lastly we created a correlation heat map to see how our variables are correlated with each other and this is where discovered that there are fairly weak correlations between most of the variables. There are no strong correlations between any of the song features and popularity however, there are some strong correlations between a few of the features themselves. For example energy is positively correlated with loudness and acousticness is negatively correlated with both energy and loudness. These correlations are not so surprising because as music listeners as we know that energetic songs tend to be louder and acoustic songs are generally more quiet and have much less energy.

Moving forward with our analysis we fit a regression model to see if we could reliably predict song popularity based off of the various features in our data set. The multiple regression model had an adjusted R-squared value of 0.09 which is extremely low and means that we were only able to account for 9% of the variability in song popularity using song features as predictors. Due to this we can conclude that a song's features are not a good indicator of the song's popularity. This result is expected as our correlation heat map shows that there are no strong correlations between song features and popularity. To wrap things up through our exploratory analysis we were able to find interesting trends in the data relating to release date, genre, and artist popularity however, when it came to predicting song popularity there were not strong enough correlations in the data and therefore our model was fairly innaccurate.

```
sessionInfo()
```

```
R version 4.2.1 (2022-06-23)
Platform: x86_64-apple-darwin17.0 (64-bit)
Running under: macOS Big Sur ... 10.16

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRlapack.dylib

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base
```

```
other attached packages:
 [1] reshape2_1.4.4  gridExtra_2.3   forcats_0.5.1   stringr_1.4.0
 [5] dplyr_1.0.10    purrr_0.3.4     readr_2.1.2     tidyr_1.2.0
 [9] tibble_3.1.8    ggplot2_3.4.0   tidyverse_1.3.2 knitr_1.41

loaded via a namespace (and not attached):
 [1] Rcpp_1.0.9          lattice_0.20-45     lubridate_1.8.0
 [4] assertthat_0.2.1    digest_0.6.29       utf8_1.2.2
 [7] R6_2.5.1            cellranger_1.1.0    plyr_1.8.7
[10] backports_1.4.1     reprex_2.0.1        evaluate_0.15
[13] highr_0.9           httr_1.4.3          pillar_1.8.0
[16] rlang_1.0.6         curl_4.3.2          googlesheets4_1.0.0
[19] readxl_1.4.0        rstudioapi_0.13     Matrix_1.4-1
[22] rmarkdown_2.14      splines_4.2.1       labeling_0.4.2
[25] googledrive_2.0.0   bit_4.0.4           munsell_0.5.0
[28] broom_1.0.0         compiler_4.2.1      modelr_0.1.8
[31] xfun_0.35           pkgconfig_2.0.3     mgcv_1.8-40
[34] htmltools_0.5.3     tidyselect_1.1.2    fansi_1.0.3
[37] crayon_1.5.1        tzdb_0.3.0          dbplyr_2.2.1
[40] withr_2.5.0         grid_4.2.1          nlme_3.1-157
[43] jsonlite_1.8.0      gtable_0.3.0        lifecycle_1.0.3
[46] DBI_1.1.3           magrittr_2.0.3      scales_1.2.0
[49] cli_3.4.1           stringi_1.7.8       vroom_1.5.7
[52] farver_2.1.1        fs_1.5.2            xml2_1.3.3
[55] ellipsis_0.3.2      generics_0.1.3      vctrs_0.5.1
[58] tools_4.2.1         bit64_4.0.5         glue_1.6.2
[61] hms_1.1.1           parallel_4.2.1      fastmap_1.1.0
[64] yaml_2.3.5          colorspace_2.0-3    gargle_1.2.0
[67] rvest_1.0.2         haven_2.5.0
```