Genre Distinctions in Spotify (PDF)

May 31, 2020

Terra Angela Hurtado

Cornell University

1 I. Introduction

1.1 Context

Music categorization is a crucial step in organizing rankings. Spotify, Billboard Charts, and the Recording Academy (GRAMMYs) are examples of groups with their own respective methods in labeling and classifying works. Though music is a highly subjective field, award counts and chart rankings are not.

Oftentimes, the classification of a work speaks volumes to its hit potential. It's common to find the Top 40 (from the Billboard Charts) entirely populated by just that: Pop, or popular, music. Less common genres to appear in the Top 40 include Rap, Rock, EDM, and R&B, but they do make appearances, oftentimes as mixed-genre music co-tagged with Pop.

Mixed-genre music is a common point of contention. One song could have a theoretically infinite number of genres: Old Town Road, as an example, fits the criteria for both Country and Rap classifications. The difficulty in music labeling comes in when organizations rely heavily on these labels for award show criteria, or for determining chart placement.

There are recurring problems in music classification. Old Town Road's classification played a huge part in its popularity. Since the song was published as a Country song, it rose on the Country charts; but in April 2019, the song was removed from Billboard's Country charts due to a perceived classification error. As part of an effort to re-establish Old Town Road as a Country piece, Billy Ray Cyrus was added to the track's re-release as a feature, which brought the song back up to a chart-topping position on Billboard's main charting system for individual songs, the Hot 100. Due to its highly publicized status as a fusion of Rap and Country, Old Town Road began gaining more traction, and topped Billboard's Hot 100 for a record-breaking nineteen consecutive weeks.

Another problem lies in the nature of the Latin classification. As a member of GRAMMY U, the Recording Academy's University chapter, I attended the 2019 GRAMMY Awards and noticed that the Latin artists attending were treated differently — almost as if they were artists outside of the mainstream. There are separate categories for Latin artists, including an entirely separate award ceremony called the Latin GRAMMYs, for this one category of music.

My hope in completing this project is that I can use what I know as a data scientist, combined with what I know as a musician, to advocate for cleaner classifications methods within the industry. There are too many examples of blurry genre boundaries having real consequences on musicians.

Latin musicians are already largely shoved to the side; language plays a big part, but by proving that Latin music does deserve to qualify for Pop awards, hopefully effort can be made towards the translation and inclusion of Latin music. R&B artists, who are predominantly black, are similarly disadvantaged in mainstream award shows. Tyler the Creator, who won a Grammy this year in the category of Best Rap Album, described his win as a “backhanded compliment” due to the misclassification of his genre. According to him, “It sucks that whenever we — and I mean guys that look like me — do anything that’s genre-bending or that’s anything they always put it in a rap or urban category” (https://www.cnn.com/2020/01/27/entertainment/tyler-the-creator-grammys-intl-scli/index.html). Song genres are messy, and often undefined, but if boundaries between song genres grow to be too undefined then discriminatory organizations will be able to take advantage of this.

1.2 Research Questions

How well could this data be used to classify songs by genre? Music classification is extremely subjective, but hopefully, enough patterns exist in that significant conclusions can be made about one genre vs. another (like Latin vs. Pop).

What classifies a genre? In award shows, if a song is ascribed the “Latin” category — even if it’s co-tagged — this takes precedence over other qualifiers. Latin songs are almost exclusively segregated into the “Latin” category (with some exceptions, like “Despacito” in 2018.) I’m looking to see if this is a fair decision, or if Latin music should instead be defined by musical qualities (i.e., tempo, energy, valence).

What separates the “Latin” music genre from all others? Spotify’s extensive tagging system should show whether Latin music exhibits technical distinctions from other categories, and if so, which ones. I want to see whether the significant segregation of Latin music is justified by these numbers, as per Spotify’s Web API.

What do you do with genre benders? When a song conforms less easily to a genre, it runs the risk of falling into an obscure category, and thereby remaining un-recommended to Spotify’s audiences through genre playlists. How do we classify these songs, then, if Spotify’s algorithm cannot decisively put it anywhere?

1.3 Main Findings

By examining objective classifications, we can see that the industry is very standardized. This is evidenced by examining categories of facts. The duration of a song, in milliseconds, cannot be argued with. According to this data, the shorter a song within a range of 3 to 2 minutes, the more popular. This speaks towards trends in the music industry, which are leaning more and more toward soundbite-style streaming. Due to use of the subscription model, minimum track length is less of a concern, allowing shorter singles to be streamed more for shorter durations. Similarly, out of all the songs in this dataset, there is a significant avoidance of pitch class 3 (or, the key of E flat). This has to be cultural, because the key of E flat holds no outlier harmonic significance.

Latin isn’t a less popular genre, nor is it particularly different, as would be implied by mainstream award shows. If anything, that genre would be EDM, whose graph of track popularity is significantly left-leaning when compared the graphs of the other genres sampled in this project. EDM has high energy and low acousticness, making it the clearest genre boundary out of the genres sampled (second perhaps to Rap, with significantly high speechiness values). And if EDM artists are eligible for awards in multiple categories, including Pop (like Skrillex, deadmau5, Avicii, etc.), then Latin artists should be as well.

Subjective classifications — like danceability, valence, or energy — are what make up a song's genre, and different genres are defined by breakages from the normal pattern. This sounds obvious, but it becomes very clear that Spotify chose these attributes to target specific differences. Unlike the objective values, subjective values conform to a Gaussian distribution; these subjective values tend to cluster around the same numbers, around 75%, which makes it more significant for a song to lie outside two standard deviations. When this happens, it usually indicates that it belongs to a separate genre.

No genre differs from any other by substantial means. Put simply, music is music. Even when examining genre boundaries, no genre of music, or cluster of Spotify data points, particularly stands out as an outlier genre.

2 II. Data Description

2.1 1. Sources

More detailed citations can be found in the final cells of the notebook.

For this project, I'll be looking at the methods of categorization used by Spotify's developers in particular, because each attribute is explained in-depth at https://developer.spotify.com/documentation/web-api/reference/object-model/#audio-features-object. The data was previously extracted by Kaylin Pavlik (https://www.kaylinpavlik.com/classifying-songs-genres/), who published the data in the blog post linked, which later made its way to the GitHub repository for tidytuesdayR (https://github.com/rfordatascience/tidytuesday/blob/master/data/2020/2020-01-21/readme.md). TidytuesdayR is a community for practicing data science with weekly prompts in R.

2.2 2. Datasheet as specified by Gebru et. al (https://arxiv.org/pdf/1803.09010.pdf)

The data consists of n = 32833, or, 32,833 song entries randomly sampled from Spotify's catalog.

Spotify uses metrics in their categorization which are classified by Spotify themselves. These metrics include ratings for a track's danceability, its instrumentalness, and its valence, among others. Respectively, these labels help indicate how easy it is to dance to a track, how much prevalence instruments hold in a track, and how sad the track sounds.

Some of these labels (i.e., "energy") may sound arbitrary, but they are the key to understanding Spotify's functioning in music recommendation. Spotify's recommendation has the power to push an artist out of obscurity, and into public awareness, just by the sheer number of subscribers on the platform. Notably, Spotify came up with these categories themselves, rather than sticking to recommendation based solely on genre.

The correlation between categories, ratings, and genres may provide insight into one of Spotify's most widely-used features: recommendation. This feature works wonders for a song's popularity, and similarly, the way it's categorized may make or break the artist's chances come award season.

What are the observations (rows) and the attributes (columns)? The observations in this dataset are individual tracks from Spotify. The attributes are as follows: track id, track name, track artist, track popularity, track album id, track album name, track album release date, playlist name, playlist id, playlist genre, playlist subgenre, danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, and duration (in ms).

Why was this dataset created? The data was taken from Spotify's Web API (https://developer.spotify.com/documentation/web-api/), which was made in order to give users a method of tracking their own Spotify data, as well as encouraging community engagement with Spotify.

Who funded the creation of the dataset? The Spotify Developer Platform was funded by and made for use in the United States by Spotify USA Inc.

What processes might have influenced what data was observed and recorded and what was not? Since many artists choose not to upload their works on Spotify, this dataset doesn't list them or their tracks even though they may be popular. Similarly, this data is only reflective of Spotify USA, since the licensing for individual tracks may differ by country, and Spotify itself struggles with this in terms of localizing the platform by country. So as a result, this dataset includes tracks from Spotify's catalog within the United States.

What preprocessing was done, and how did the data come to be in the form that you are using? The data includes Spotify's own method of classification, for use by recommendation apps. It indexes songs by their track id, which is a unique identifier for that track which was assigned to it by Spotify Inc. The data comes from Spotify, and was put in the form of a .csv by Kaylin Pavlik and uploaded by the mods of the tidytuesdayR repo.

If people are involved, were they aware of the data collection and if so, what purpose did they expect the data to be used for? There is no user-specific data in this dataset, only classification data for individual tracks as they are assigned by Spotify. Only track_popularity takes information directly from Spotify's user base, and this data is anonymized. One thing to note is that while Spotify doesn't explicitly cite a use for their API, there exist tokens for accessing the data of specific users. Spotify doesn't grant non-developers these privileges. While there exist many playlists which are not curated by Spotify, for example, public playlists (public = TRUE grants access without a token), these user-curated playlists are not included in this dataset.

Was any preprocessing/cleaning/labeling of the data done? The dataset contains the data as Spotify prepared it, but I do go through the dataset myself later in this project to very perform minimal processing. The raw data is what exists at the beginning point of this notebook. Specifically, later, I go through listwise and delete missing entries for model clustering, and create subsets of genres or qualities to make some graphs more clear.

2.3 Potential Problems with the Dataset

This data only observes popular songs from the six most popular categories, and specifically from Spotify USA. For licensing reasons, many popular artists such as Beyonce and Taylor Swift have chosen not to make their music accessible on Spotify. The reasoning for this can vary: Beyonce, for example, publishes her solo/main work exclusively to Tidal. Taylor Swift is on Spotify as of 2017, but boycotted the platform for years because of their unfair payment distribution for artists (Swift's reasoning being that she will boycott, since many small artists don't have this option).

This means the catalog of songs sampled to create this dataset was not exhaustive, as it excluded many artists with popular tracks that just aren't on Spotify.

3 III. Data Analysis

I'll be installing two packages, including tidyverse, which consists of its own graphing and data-managing packages: ggplot2 and dplyr, respectively. Mclust is a package used for model-based approaches to data science — it generates the log likelihood that data fits to a certain model, and prints that model in its summary.

[1]:
```
# Initialize
install.packages("tidyverse") # from here i want ggplot2 and dplyr specifically
                                # dplyr for management/pipe operator %>%, and
↪ggplot2 for graphing

install.packages("mclust") # this is to be used later for model clustering

install.packages("pastecs") # for detailed summary statistics
```

```
Updating HTML index of packages in '.Library'
Making 'packages.html' … done
Updating HTML index of packages in '.Library'
Making 'packages.html' … done
Updating HTML index of packages in '.Library'
Making 'packages.html' … done
```

Similarly, I'm loading the libraries here, as well as turning off scientific notation.

[2]:
```
# initialize (load libraries)
library(tidyverse)
library(mclust)
library(pastecs)

# turn off scientific notation
options(scipen=999)
```

```
 Attaching packages                         tidyverse 1.3.0
 ggplot2 3.3.0      purrr   0.3.4
 tibble  3.0.1      dplyr   0.8.5
 tidyr   1.1.0      stringr 1.4.0
 readr   1.3.1      forcats 0.4.0
 Conflicts                            tidyverse_conflicts()
 dplyr::filter() masks stats::filter()
 dplyr::lag()    masks stats::lag()
Package 'mclust' version 5.4.6
Type 'citation("mclust")' for citing this R package in publications.

Attaching package: 'mclust'
```

```
The following object is masked from ‘package:purrr’:

    map


Attaching package: ‘pastecs’

The following objects are masked from ‘package:dplyr’:

    first, last

The following object is masked from ‘package:tidyr’:

    extract
```

Below, I'll grab the data from the repo for tinytuesdayR, a community which comes up with weekly data science prompts in R. This data was from the week of 2020-01-21. This grab should specify delimiter ",".

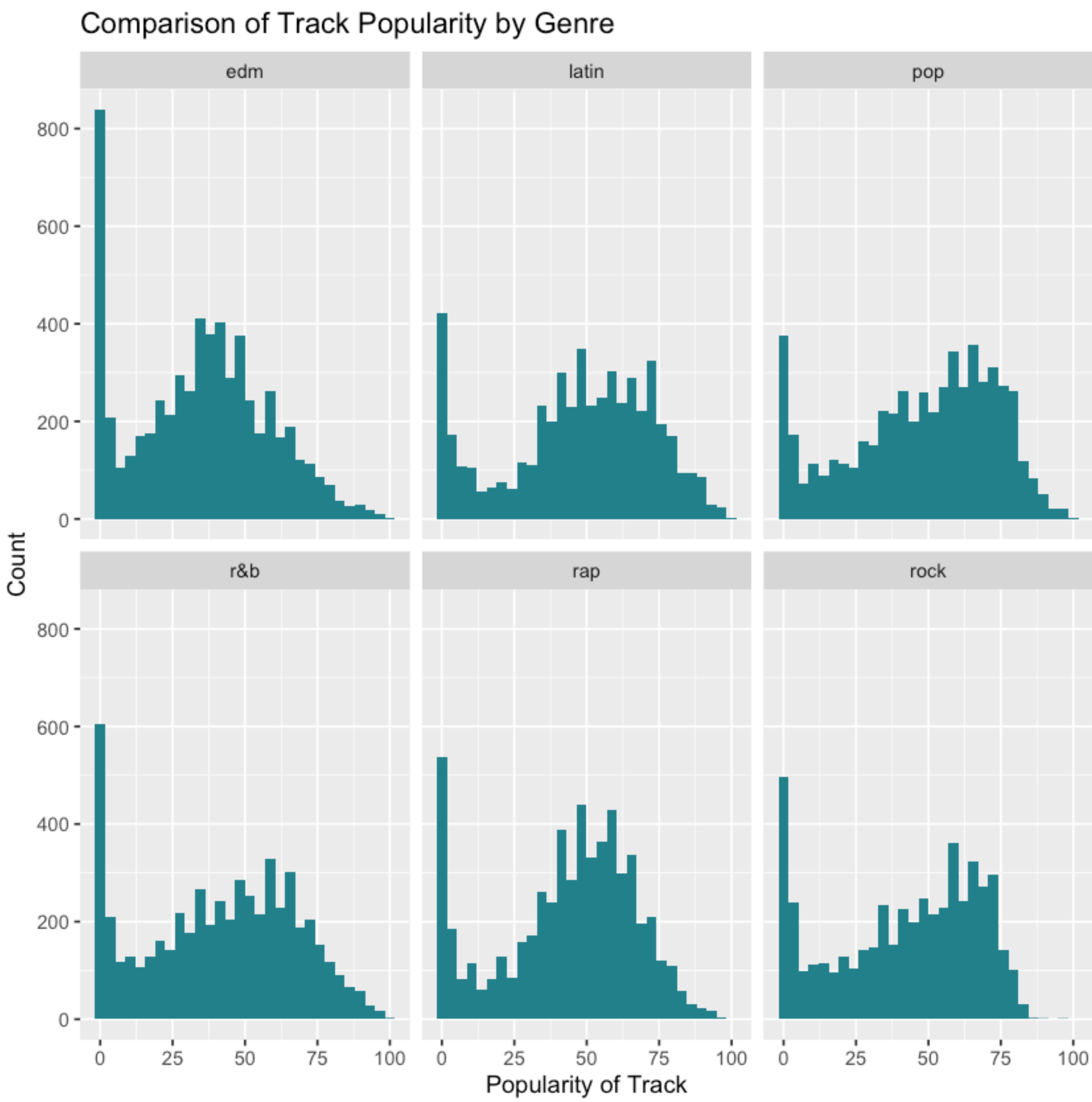
[3]:
```
# Grab data from github
spotify_songs <- readr::read_csv('https://raw.githubusercontent.com/
→rfordatascience/tidytuesday/master/data/2020/2020-01-21/spotify_songs.csv')
```

```
Parsed with column specification:
cols(
  .default = col_double(),
  track_id = col_character(),
  track_name = col_character(),
  track_artist = col_character(),
  track_album_id = col_character(),
  track_album_name = col_character(),
  track_album_release_date = col_character(),
  playlist_name = col_character(),
  playlist_id = col_character(),
  playlist_genre = col_character(),
  playlist_subgenre = col_character()
)
See spec(…) for full column specifications.
```

3.1 Point of analysis: Track Popularity by Genre

Below is a visual comparison of track popularity values, separated by genre. The Pop and Rock genres have the best luck with track popularity, where EDM has a high number of tracks with a relative popularity close to zero.

[4]:
```
ggplot(spotify_songs) +
 aes(x = track_popularity) +
 geom_histogram(bins = 30L, fill = "#26828e") +
 labs(x = "Popularity of Track", y = "Count", title = "Comparison of Track␣
 ↪Popularity by Genre") +
 theme_gray() +
 facet_wrap(vars(playlist_genre))
```



Even with the differences in the genres, each graph takes on a distribution shape which is roughly normal. Pursuing EDM may just mean that beginning artists have tougher luck getting popular. However, with the increasing accessibility of electronic music tools, and the steep prices of acoustic instruments (and lessons) typically associated with rock bands, this is to be expected. EDM also has a visibly high total count — this means that many EDM tracks exist in Spotify's database, which may also explain why it's harder for individual tracks to stick out and grow popular.

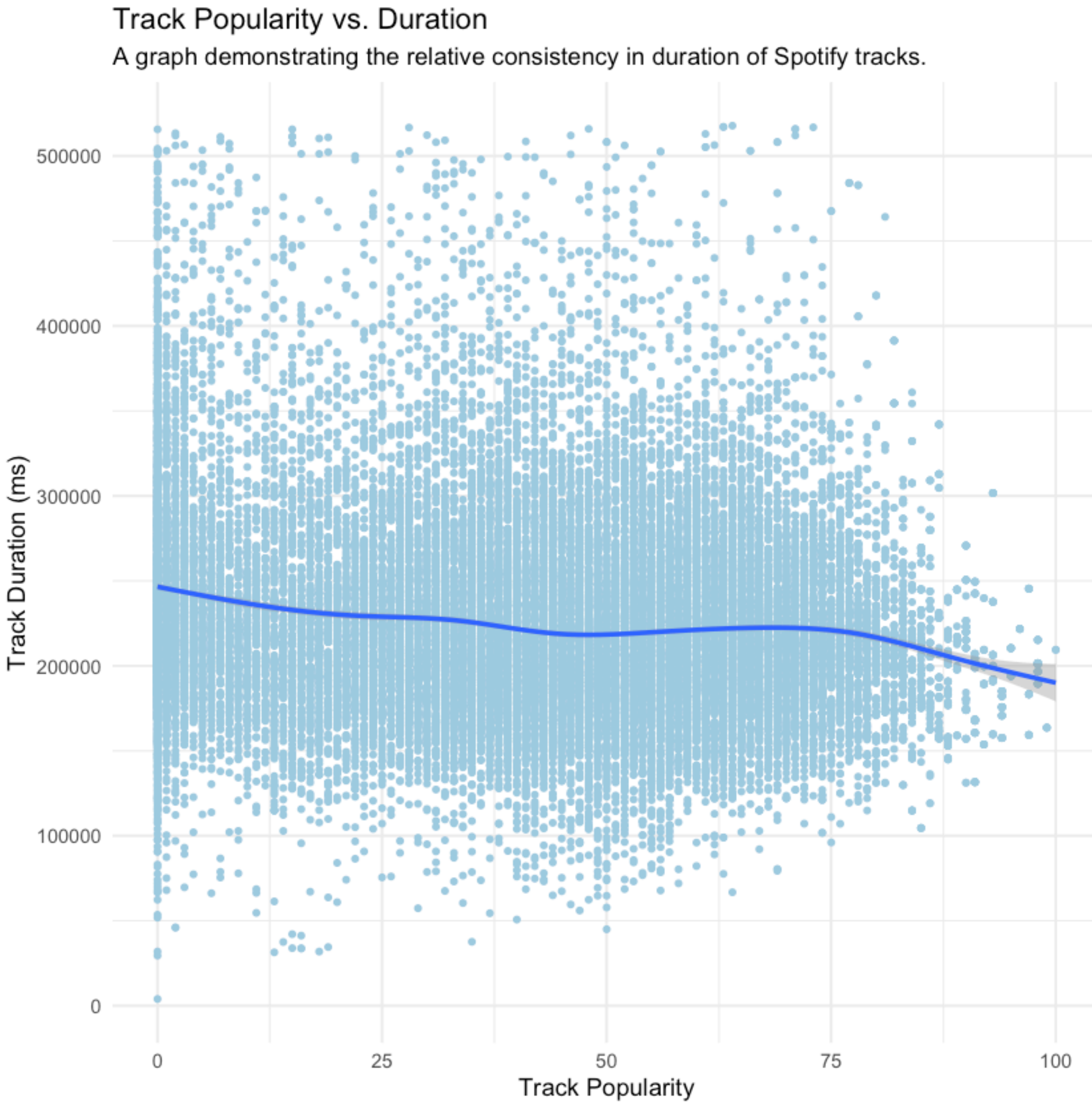
While these graphs each tell of track popularity within genre, none signify a genre as **unpopular**, and this is key. Each of the above genres has its own measure of popularity, but none are particularly low.

3.2 The significance of song length, pitch class, and culture

Below is a graph of the popularity of individual tracks vs. their duration. They stay largely within the range of 3.33 minutes and 4.17 minutes. Notably, however, a track popularity tends to decline as track duration grows — the two are visibly negatively correlated.

[5]:
```
ggplot(spotify_songs) +
 aes(x = track_popularity, y = duration_ms) +
 geom_point(size = 1L, colour = "#9ecae1") +
 geom_smooth(span = 0.75) +
 labs(x = "Track Popularity", y = "Track Duration (ms)", title = "Track␣
↪Popularity vs. Duration", subtitle = "A graph demonstrating the relative␣
↪consistency in duration of Spotify tracks.") +
 theme_minimal()
```

```
`geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



Centuries ago, audiences would pay high prices to see professional music, and so they expected their money's worth — some pieces (like 16th religious passions, for example) would go on for hours.

More recently, with the development of vinyl records, buyers would expect enough of a runtime to justify their purchase, and this birthed the industry standard still in use today. Songs are expected to be somewhere around 3 minutes, 33 second (incidentally, this was Johann Sebastian Bach's favorite length, as well). For someone to buy a solo release, the song would have to be appropriately long to justify the use of physical records/tapes/CDs.

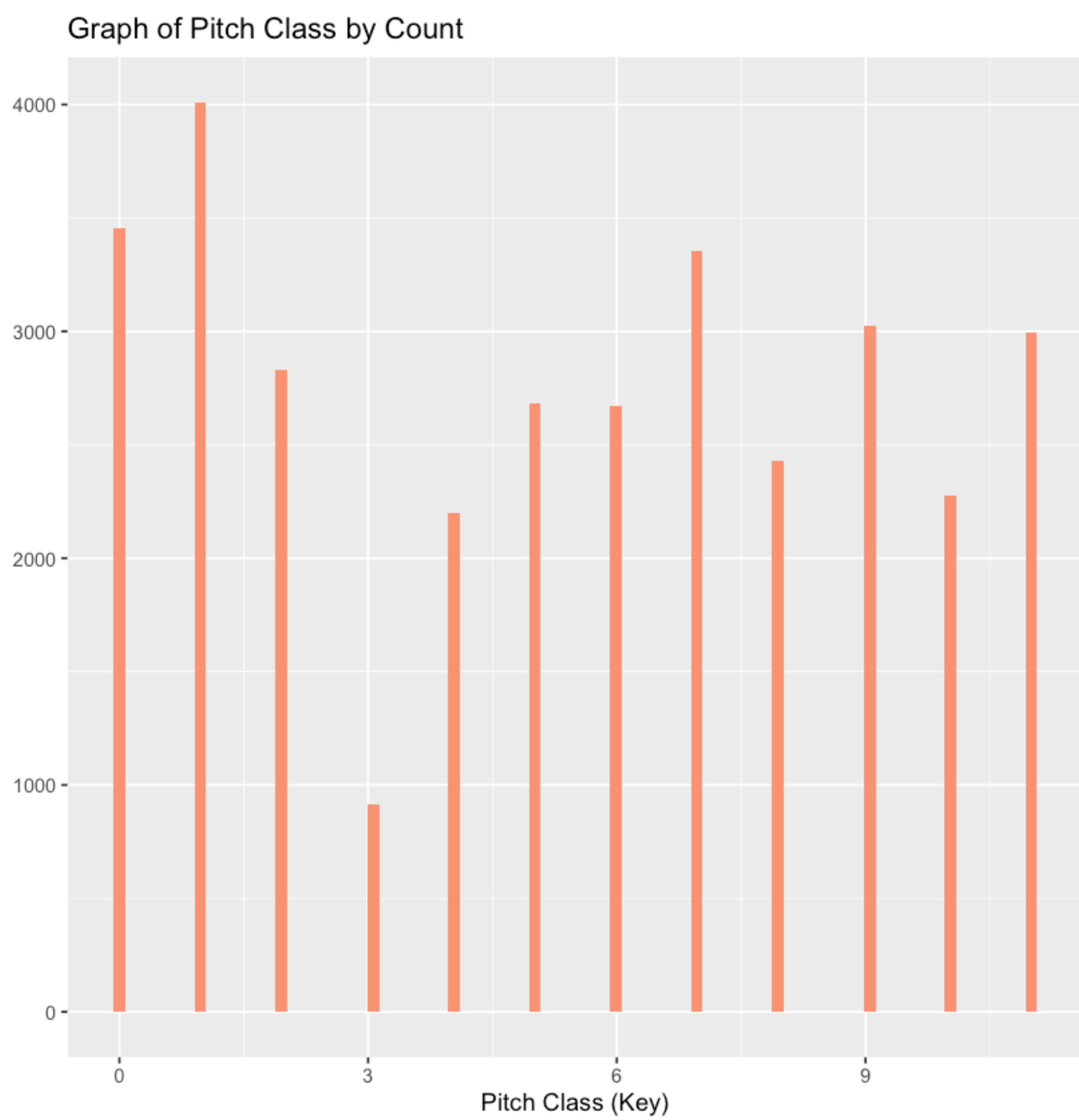
These days, however, music distributors typically employ the subscription model. Spotify, Apple Music, and sometimes YouTube require recurring payments but allow access to all songs within the database. For modern-day streaming audiences, song length matters much less, as music streaming gets closer to unlimited.

Because of the way streaming works, it actually works in a song's favor to be shorter.

Lil Nas X, the author of Old Town Road, knows this well, and keeps his songs to a much shorter average, effectively making his streaming bite-sized. It's easier for his tracks to garner a higher stream count, simply because it takes less time to do so. This pattern can be seen above: for the most part, the tracks stick to the industry standard, but shorter tracks tend to do a bit better.

A similar example of industry standard can be seen in the graph of pitch class — or, key. I've graphed the usage of pitch class below; out of all the songs indexed, the one with the obvious minimum is 3.

[6]:
```
ggplot(spotify_songs) +
 aes(x = key) +
 geom_histogram(bins = 80L, fill = "#fc9272") +
 labs(x = "Pitch Class (Key)", y = " ", title = "Graph of Pitch Class by␣
↪Count") +
 theme_gray()
```



This graph employs the use of **pitch class**, which assigns a numerical value to each key. The key of 3 is E flat, and clearly, it's being underutilized.

There is no technical reason for this. E flat is a fine key; in fact, it's one of the easiest to play on piano, because it consists of mostly black keys, which are easier to jump back-and-forth from in fast sections. But as evidenced, it is the least used key by a wide margin. This tells me that the songwriters of Spotify are involved with each other in a way that indicates a shared culture: one which excludes the key of E flat.

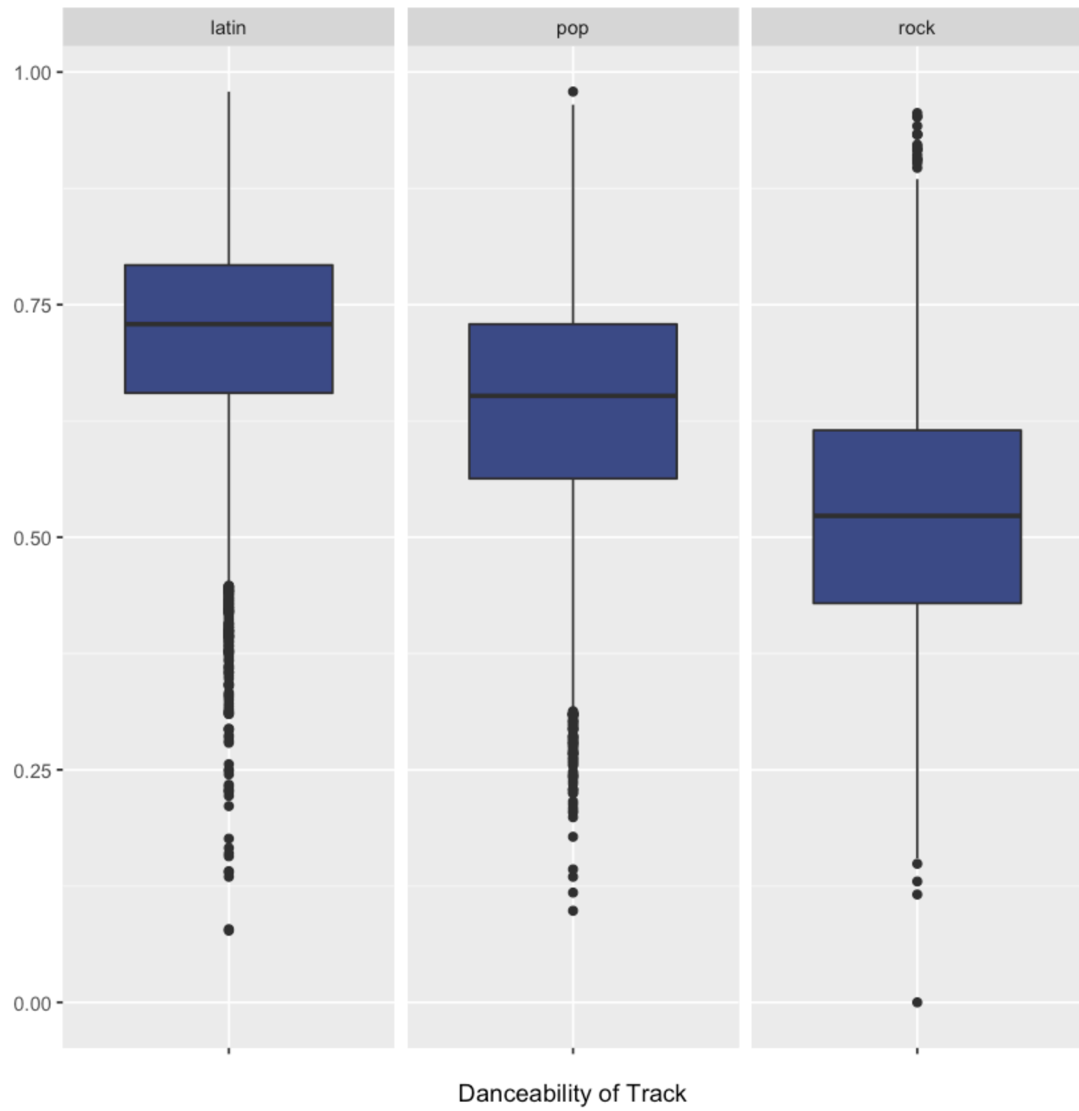
This confirms to me, as a musician, that **musicians talk to each other** and generally agree on things like song duration and pitch class. There are clear industry standards, and they tie together all music, regardless of genre classification.

3.3 A look at Latin

When looking at Latin music, as a genre, it becomes apparent that it holds variety. This variety is enough to tell me that, perhaps, this genre classification should be broken up.

In the boxplot below, I've compared Latin to the genres of Pop and Rock, in regards to individual tracks' relative danceability factor.

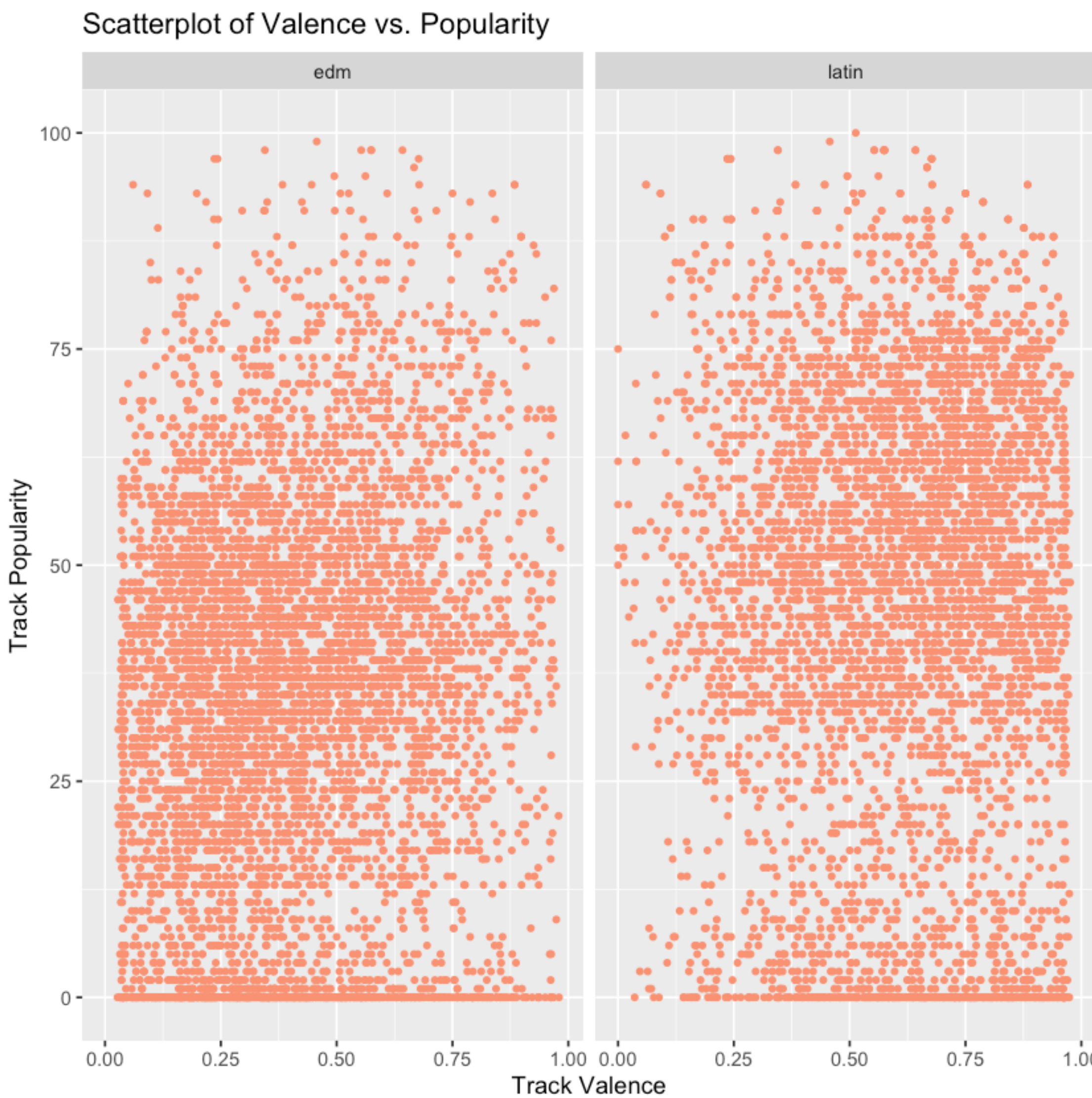
[7]:
```
spotify_songs %>%
 filter(playlist_genre %in% c("pop", "rock", "latin")) %>%
 ggplot() +
 aes(x = "", y = danceability) +
 geom_boxplot(fill = "#3e4a89") +
 labs(x = "Danceability of Track", y = " ") +
 theme_gray() +
 facet_wrap(vars(playlist_genre))
```



Something to notice is that **Latin has a lot of outlier tracks,** about twice as many. The tails of the Latin boxplot are the shortest, which indicates a high density with many outliers.

Below is another graph, which may illustrate why that is:

[8]:
```
spotify_songs %>%
 filter(playlist_genre %in% c("latin", "edm")) %>%
 ggplot() +
 aes(x = valence, y = track_popularity) +
 geom_point(size = 1L, colour = "#fc9272") +
 labs(x = "Track Valence", y = "Track Popularity", title = "Scatterplot of␣
↪Valence vs. Popularity") +
 theme_gray() +
 facet_wrap(vars(playlist_genre))
```



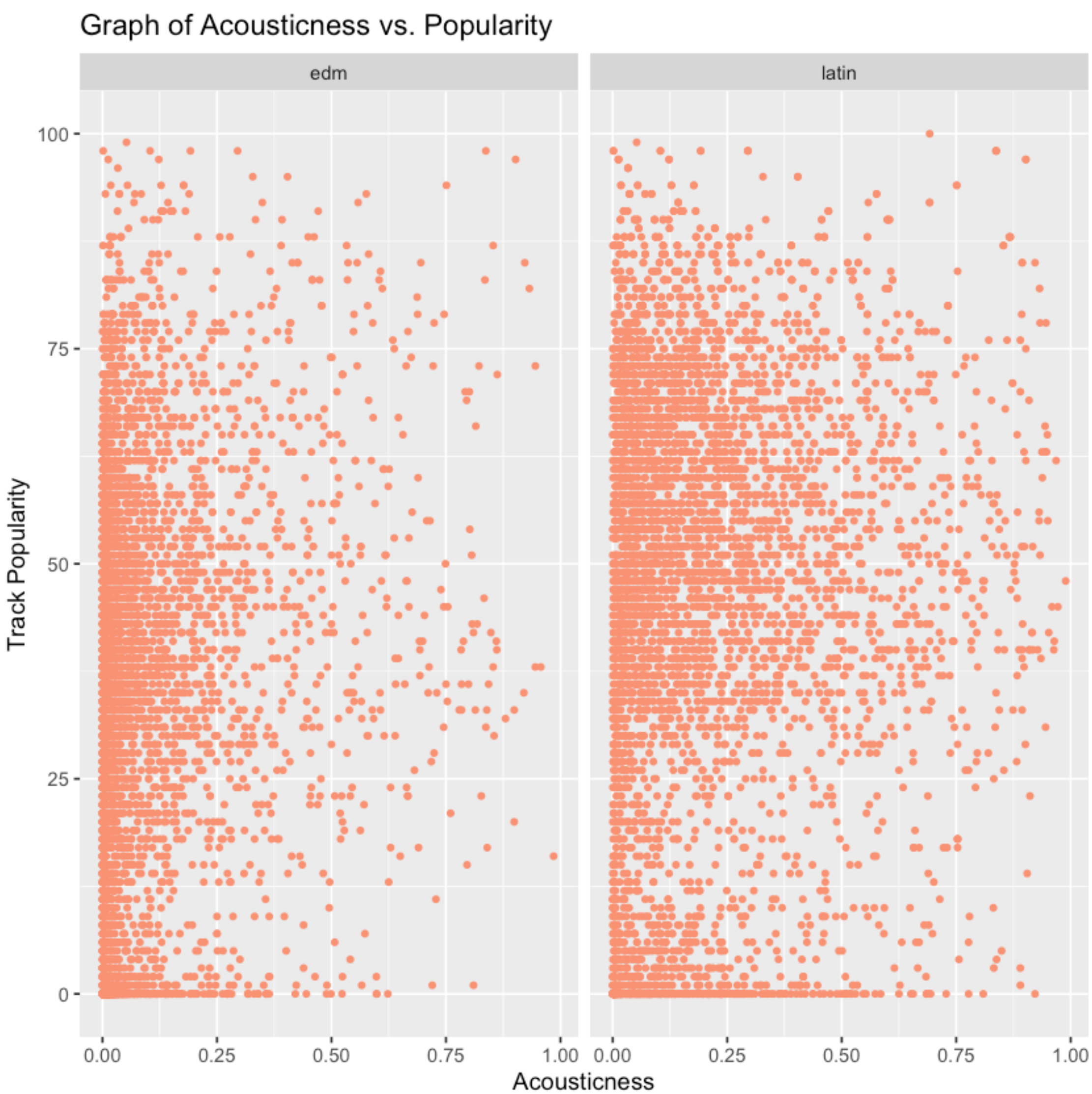
These two scatterplots contrast the popularity and valence of two genres: EDM and Latin. According to these, Latin tracks tend to cluster to the right, while EDM tracks cluster to the left. EDM has more songs with lower valence, and Latin has more with higher valence.

Using domain knowledge, and looking at the comparison to EDM, my conclusion is that **the Latin genre has more ballad-type songs.** Ballads are slower, sadder, and more acoustic, as a rule (think Adele).

We can test this by looking at two more comparisons between EDM and Latin: acousticness, and tempo. The following two graphs display acousticness vs. popularity, and tempo vs. popularity, respectively.

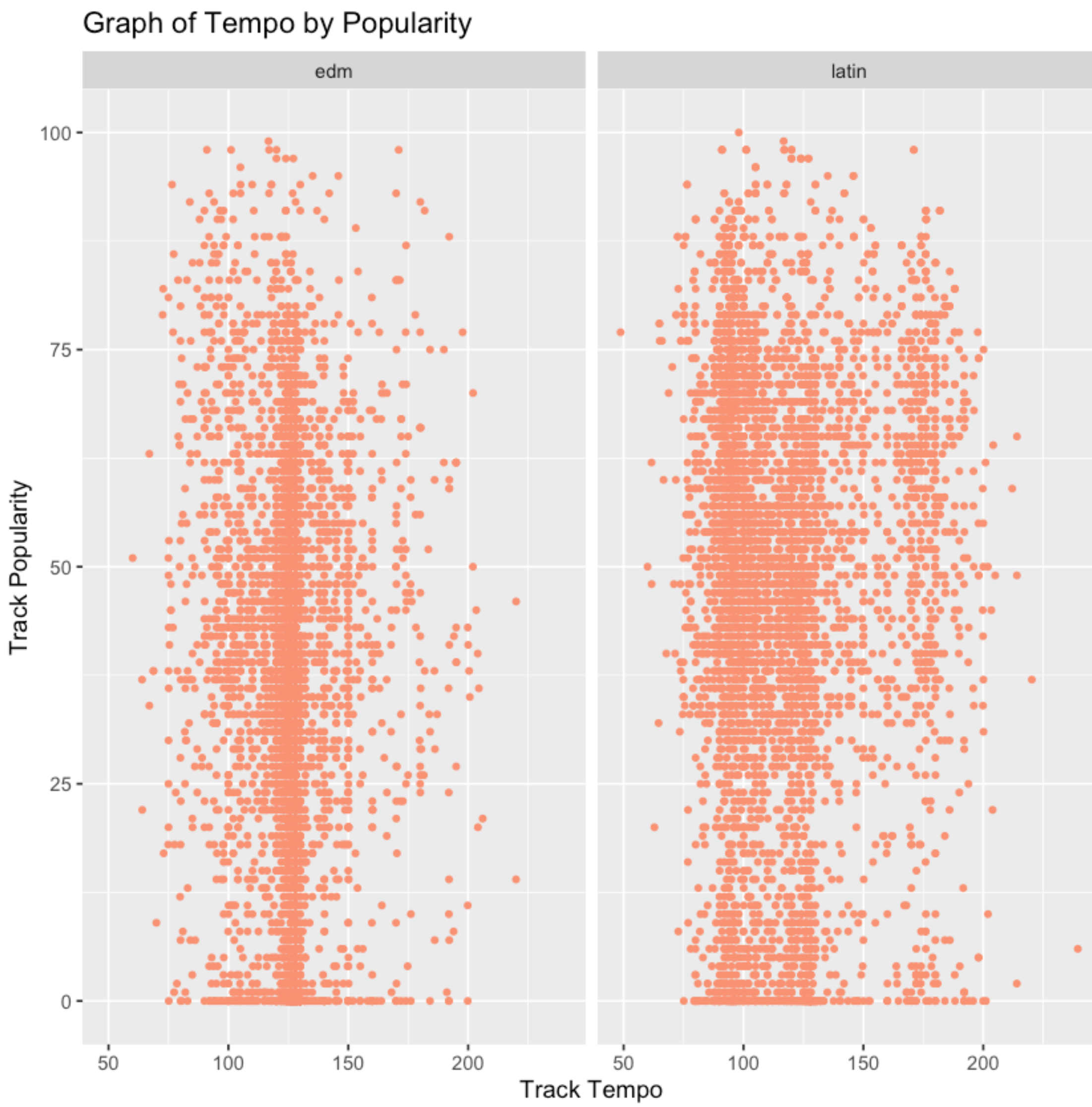
[9]:
```
spotify_songs %>%
 filter(playlist_genre %in% c("latin", "edm")) %>%
```

```
ggplot() +
aes(x = acousticness, y = track_popularity) +
geom_point(size = 1L, colour = "#fc9272") +
labs(x = "Acousticness", y = "Track Popularity", title = "Graph of␣
↪Acousticness vs. Popularity") +
theme_gray() +
facet_wrap(vars(playlist_genre))
```



[10]:
```
spotify_songs %>%
filter(playlist_genre %in% c("latin", "edm")) %>%
ggplot() +
aes(x = tempo, y = track_popularity) +
geom_point(size = 1L, colour = "#fc9272") +
```

```
labs(x = "Track Tempo", y = "Track Popularity", title = "Graph of Tempo by␣
↪Popularity") +
theme_gray() +
facet_wrap(vars(playlist_genre))
```



3.4 Using subjective values to define genres

In this section, I want to illustrate the occurrences of variables like energy, danceability, and valence, to hopefully prove how capable they are at drawing distinctions. Specifically, I want to show that these values have been identified properly, and fit a normal distribution.

In the cell below, I took several steps to create a subset with those previously mentioned subjective categories. I wanted to see how accurate they were, and a good way to do that is to measure the subset's adherence to a normal — Gaussian — distribution.

Similarly, I wanted a subset that contained objective data points. Key, tempo, and duration were the attributes I selected, and if my prediction is right, then the subjective values should more closely adhere to a normal distribution.

[11]:
```
very_sub <- select(filter(spotify_songs), energy, danceability, valence) #␣
 ↪subset with subjective values
clean_sub <- na.omit(very_sub) # this cleans by listwise delete missing data␣
 ↪values

very_obj <- select(filter(spotify_songs), key, tempo, duration_ms) # subset␣
 ↪with objective values
clean_obj <- na.omit(very_obj) # cleaning them up by listwise deletion
```

Now that I've prepared the data, I search for a model with best fit. The below code cell performs a model comparison on the subjective qualities, and summarizes the log likelihood of the data fitting a Gaussian model.

[12]:
```
sub_fit <- Mclust(clean_sub) # gaussian model based clustering
#reference from https://www.statmethods.net/advstats/cluster.html
summary(sub_fit) # perform a summary
```

```
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust VVV (ellipsoidal, varying volume, shape, and orientation) model with 9
components:

 log-likelihood     n df      BIC      ICL
       36455.15 32833 89 71984.78 41121.53

Clustering table:
   1    2    3    4    5    6    7    8    9
4866 2380 2404 3091 4470 7240 2365 1562 4455
```

And once again, this process can be performed on the objective qualities, which I don't expect to be normally distributed.

[13]:
```
# gaussian model, same reference: https://www.statmethods.net/advstats/cluster.
 ↪html
obj_fit <- Mclust(clean_obj)
summary(obj_fit) # print summary
```

```
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------
```

```
Mclust VVE (ellipsoidal, equal orientation) model with 9 components:

 log-likelihood     n df      BIC      ICL
      -638659.6 32833 65 -1277995 -1299286

Clustering table:
   1    2    3    4    5    6    7    8    9
6057 2555 4247 4074 6799 3386 2017  939 2759
```

The log likelihood for the subjective value vector is much, much higher. This tells me that the subjective data is normally distributed, and that it's likely to fit natural occurrences of qualities like energy or danceability, which are hard to narrow down.

This can be further proved by the variance and standard deviation values shown below. The objective descriptors are higher, indicating more space between values.

[14]:
```r
print("Statistics descriptions for the subjective values:")
stat.desc(clean_sub)
print("Statistics descriptions for the objective values:")
stat.desc(clean_obj)
```

```
[1] "Statistics descriptions for the subjective values:"
```

	energy	danceability	valence
nbr.val	32833.0000000000	32833.000000000	32833.000000000
nbr.null	0.0000000000	1.000000000	1.000000000
nbr.na	0.0000000000	0.000000000	0.000000000
min	0.0001750000	0.000000000	0.000000000
max	1.0000000000	0.983000000	0.991000000
range	0.9998250000	0.983000000	0.991000000
sum	22937.7665150000	21500.674300000	16763.248450000
median	0.7210000000	0.672000000	0.512000000
mean	0.6986192707	0.654849520	0.510560974
SE.mean	0.0009984064	0.000800697	0.001286686
CI.mean.0.95	0.0019569128	0.001569395	0.002521952
var	0.0327284404	0.021049750	0.054357045
std.dev	0.1809100339	0.145085320	0.233145974
coef.var	0.2589536841	0.221555206	0.456646682

```
[1] "Statistics descriptions for the objective values:"
```

	key	tempo	duration_ms
nbr.val	32833.00000000	32833.0000000	32833.000000
nbr.null	3454.00000000	1.0000000	0.000000
nbr.na	0.00000000	0.0000000	0.000000
min	0.00000000	0.0000000	4000.000000
max	11.00000000	239.4400000	517810.000000
range	11.00000000	239.4400000	513810.000000
sum	176460.00000000	3968890.1920000	7413685215.000000
median	6.00000000	121.9840000	216000.000000
mean	5.37447081	120.8811315	225799.811622
SE.mean	0.01993202	0.1484757	330.211960
CI.mean.0.95	0.03906748	0.2910178	647.227409
var	13.04406924	723.8049936	3580108295.744392
std.dev	3.61165741	26.9036242	59834.006182
coef.var	0.67200242	0.2225626	0.264987

4 Conclusion

My hope in having this project completed is that I can use what I know as a data scientist, combined with what I know as a musician, to advocate for cleaner classification methods within the industry. I understand that song genres are messy, and undefined, but if boundaries between song genres grow to be too undefined then my worry is that mainstream organizations will wield this as a tool to justify discrimination against targeted artists. The decision to put an artist in a particular category should not have a race factor. If genre data can do the bulk of the work in classifying music, then that's one less arbitrary decision to be made with ulterior motives.

By examining objective classifications, we can see that the industry is very standardized. This tells me that genre boundaries aren't as clear as one may think, and certain songs may require deliberation regardless of scientific classification methods. Since musicians talk to each other, and commonly agree on subjects like pitch class and song length, it stands to reason that genres have more factors in common with each other than they do different.

Popularity should speak for itself. Popularity should be more highly weighted in award categories than any subjective label, because distinctions due to genre have less significance than one might think. If a track has popularity, there's no reason it can't be considered pop, and nominated for awards in the Pop genre.

Subjective classifications — like danceability, valence, or energy — are what make up a song's genre, and different genres are defined by breakages from the normal pattern. Many Latin tracks bear more similarity to other genre styles than they do to other Latin tracks within the same genre. If Latin music is highly danceable, but it's also very sad, and it's similarly very slow, then perhaps individual songs are different enough from one another to warrant classification in a new genre. Fast, loud Latin songs should be eligible for Rock category awards, and slow, sad ones should be reclassified as R&B or pop-ballads.

No genre differs from any other by substantial means. What this means to me, as a musician, is that music companies should put faith in their artists when it comes to genre. Especially when examining a genre-bending piece, it should be up to the artist to classify it for the sake of awards and recommendation.

5 Bibliography

Object Model | Spotify for Developers. (n.d.). Retrieved May 8, 2020, from https://developer.spotify.com/documentation/web-api/reference/object-model/

Owoseje, T. (2020, January 27). Tyler, The Creator Tyler slams Grammys' "urban" category as a politically correct version of the n-word—CNN. Retrieved May 25, 2020, from https://www.cnn.com/2020/01/27/entertainment/tyler-the-creator-grammys-intl-scli/index.html

Pavlik, K. (2019, December 20.) Classifying genres in R using Spotify data. Retrieved May 18, 2020, from https://www.kaylinpavlik.com/classifying-songs-genres/

Rfordatascience/tidytuesday. (n.d.). GitHub. Retrieved May 18, 2020, from https://github.com/rfordatascience/tidytuesday

Spanos, B. (2019, December). How Lil Nas X and "Old Town Road" Defy Categorization. Smithsonian Magazine. Retrieved May 16, 2020, from https://www.smithsonianmag.com/arts-culture/lil-nas-x-old-town-road-american-ingenuity-180973492/