

Music Trends in the U.S.

An Analysis of Genre, Sentiment, and Audio Features

Caitlyn Maung, Shreya Vallala, Chris Nguyen

Professor Peter Kramlinger

STA141B Final Project

December 11, 2024

I. Introduction

Throughout human history, music has been a huge part in our day to day lives for many reasons. It inflicts emotions while we listen, it can define who we are as people, and just like people, music evolves as the time changes. Music continuously changes over time as new sounds, genres, artists, and technology evolves. However, quantifying this evolution is hard to visualize and interpret by the general population. Hence, through this project, we have a goal of analyzing trends in music over time through various methods of data acquisition to see how music evolved over the past decade using components such as genre, descriptions, and audio features. By visualizing these metrics through our project, we look forward to providing a comprehensive visual of how musical trends are not stagnant over time, but rather building on popular music tastes through the years.

II. Individual Contributions

Caitlyn Maung

Led scraping process for Billboard Hot 100 charts to access top 10 songs for each month between 2010 and 2023. Also worked on performing a sentiment analysis for each song observed by scraping data provided by Genius. Worked on analyzing and plotting the results of this section.

Chris Nguyen

Led efforts to use iTunes Search API to access genres of individual songs in the top 10 positions for Billboard Hot 100 charts between 2010 and 2023. Worked on analyzing and plotting the results of this section.

Shreya Vallala

Led scraping process for SongData.io to access audio features for individual songs in the top 10 positions for Billboard Hot 100 charts between 2010 and 2023. Worked on analyzing and plotting the results of this section.

III. Identifying Popular Songs between 2010 and 2023

The first objective we needed to achieve was getting the music data itself so we can later obtain the different aspects of those particular songs. We chose the Billboard website to extract our music data because it is a very reliable and popular source for music and entertainment. Billboard has weekly Hot 100 charts containing the top 100 songs in the US dating back to the 1960s. Along with that, Billboard hosts the Billboard Music Awards every year which are awarded to the top artists in a variety of categories making it a well established source of music data. Ultimately, we decided to scrape the songs for the years 2010 - 2023 because we hypothesize that the past decade is a sufficient time frame to still uncover interesting details on how songs with different genres or moods/sentiments changed in popularity. By observation, we also found that popular songs tended to stay high on charts for multiple weeks on end and so, to avoid extensive repetition of data, we decided to only use charts for the last week of every month. Although having access to 100 tracks per chart, we chose to acquire only the top 10 songs because it seemed to be the most efficient method in terms of runtime while still providing us with a sufficiently large dataset for our analysis.

To begin the scraping process for all songs, we used the *Calendar* module to write a function called `get_last_day_of_month` and find the last day of each month (in order to later get the top 10 songs at the end of each month). Then, we wrote a function called `get_top_10_songs` that takes a year as input and retrieves the top 10 songs at the end of each month for that year. A challenge we faced here was that the #1 song of each month was listed in a different class than the other 9 songs and that song was misplaced in the top 10 list, so we had to make sure to add the first song in their correct position. Afterwards, we used this function to get the top 10 songs for each month for the years 2010-2023 using the `get_top_songs_for_multiple_years` function. We then stored all of these songs into a dictionary called `top_songs_per_year` and printed all of the songs as a list under their corresponding end-of-month dates for convenient reference.

Top 10 Billboard Hot 100 Songs for 2010:

For 2010-01-31:
 TiK ToK by Ke\$ha
 Today Was A Fairytale by Taylor Swift
 Bad Romance by Lady Gaga
 BedRock by Young Money Featuring Lloyd
 Baby by Justin Bieber Featuring Ludacris
 Replay by Iyaz
 Sexy Chick by David Guetta Featuring Akon
 Empire State Of Mind by Jay-Z + Alicia Keys
 Hard by Rihanna Featuring Jeezy
 Hey, Soul Sister by Train

Example: The top 10 songs from last week of January 2010.

IV. Identifying Genres for Popular Songs

For our first analysis, we want to observe how genres of popular music changed over time to provide insight into the variety of genres listened to by the population. Once we got all the top songs from every month of every year, it is now time to figure out what their genres of music are. To do so, we utilized the iTunes Search API from Apple to make searches of the songs we got previously from Billboard and return a bunch of information about that song, which includes the genre of the song that we want for our purposes. We opted to use the iTunes API primarily as it is a well-known database that we can confidently extract the most accurate information of the genre of a song we are looking for.

The first step before using the iTunes Search API is to pre-process the Billboard data from a dictionary into a list object with all the songs from 2010 to 2023. Using a nested for loop, we got a list of 1680 songs stored in the `all_songs` list variable. This is an accurate result considering we are working with the top 10 songs of each month, making it 120 songs a year, across 14 years which includes 2010, and we get the same number of 1680.

Next, we created a function called `get_genre` which takes a song title from our list of songs, puts it into a search url which we crafted via f-string and sends a get request to the database using that search url. If the request is successful, indicated by a code status equaling 200, then it formats the request results in json format to finally extract the 'primaryGenreName' or genre of music of the first search result through indexing. If a genre can't be found or the get

request was unsuccessful due to that particular song not being available, it will print an error message and append a 'None' value to its genre result.

Once the function equipped with the iTunes API was constructed, we created another empty list to store the results of `all_songs` being run through the `get_genre` function. We then ran a for loop with some downtime, 0.5 seconds, in between iterations to cache our requests of 1680 songs and prevent getting timed out. In the end, out came `song_genres`, a list of all the genres correlating to the 1680 songs from the Billboard results in their respective order.

With our list of genres, we now needed to group the genres of songs back into the months they were from and then later group the months into their respective years to fulfill our objectives. So we must bundle the genre results into sublists within the big list to indicate a month's genre of music and count how many of each genre was present during that time. Bundling our data from `song_genres` was not hard, using the function `divide_list_into_chunks`, we made sublists within the big list of genres of `song_genres` to portion out 10 songs per sublist as a representation of a month's top 10 songs. Next was counting the amount of each genre within each sublist/month, we did so using the *Counter* package and storing the information in a dictionary object. What we got now is a list of dictionaries, each dictionary element correlating to a month's genres of music and their amounts of genres within said month.

Finally it is the plotting step where we used this list of dictionaries organized data to visually see how the genres' popularity behaves over the years. To achieve this, we created a data frame using the pandas package to format the data for plotting. Some data conversion with the indexes were made, data grouping the large list of dictionaries as mentioned was done for 10 songs per month and then 12 months per year, and lastly the utilization of the *matplotlib.pyplot* package for multiple plots representing the last decade (2010-2023), we have our overall graphical representation of each year, indicating the genre counts of the top 10 songs of each month within that year, seen as line graphs with legends to indicate which line is which genre.



Figure 1: All the plots of song genres for popular songs across the years 2010-2023.

From 2010 to 2014, we can observe that the Pop genre of music has dominated the top 10 billboards for those years, giving us a picture of how music back then revolved around high energy music and catchy beats. Interestingly, from 2015 to 2020, Hip-Hop/rap became a close contender to Pop music in popularity. Hip-hop/rap songs increased in the top 10 while Pop music declined in 2015, and even dominated over Pop music in 2018 and 2020. So now, we see the Hip-hop/rap genre taking the place as a popular music genre within this time period, giving us a picture of fast-paced rhythms and electric tunes being widely liked within its listeners. Concluding with 2021 to 2023, we see a resurgence of Pop music once again, especially in 2021 when it almost matched its popularity in the early 2010's. 2023 seems to have a mix of even amount of genres for its top 10 songs with Afrobeats being the top for the first half of the year and Country and Pop seemingly taking the popularity spot towards the end with Christmas for the holidays. In short, Pop music took back the spot as the top genre for the majority during the next 3 years with the same reasons for why Pop music was popular in the 2010's.

Based on our findings from the iTunes API, we can conclude that Popular music, or Pop music as it is widely known, is considered by the populus as the best genre of music from 2010 to 2023 overall. As a result, we can interpret this as the highly upbeat and fast paced sounds of Pop music being the reason why it caters the most favors from listeners of the US demographic. Admittingly, a challenge arose during this process where finding the genre of certain songs failed due to the naming or unique circumstances where the genres can't be extracted. But in the end, we chalked this up as a natural imperfection as it didn't really affect the results of our findings. Overall the iTunes API confirms Pop music to be the top genre of the 2010's and early 2020's.

V. Using Genius Descriptions for Sentiment Analysis

For the next procedure in our project, we decided to not only find the genres of the top songs, but also look into their lyric pages to figure out what kind of sentiment the song is really giving to its listeners. The Genius website was the choice to scrape the lyrics pages of each song, since Genius is the world's most popular collection of song lyrics and musical knowledge. It also provides really accurate descriptions and annotations of songs.

We first wrote a function called `get_genius_lyric_page` that takes in a song title and the artist's name, and gets the song's lyric page from the Genius website. We then applied this function to all songs in the `all_songs` list from the API stage, and printed out each song and their corresponding Genius links. However, while getting the Genius urls of the songs, I realized that the urls for some of the songs were formatted incorrectly, and led to a page not found. For example, the song Empire State of Mind by Jay Z and Alicia Keys was formatted as 'Empire State Of Mind by Jay-Z + Alicia Keys', on the Billboard chart. The Genius url of this song was “<https://genius.com/Jay-z-empire-state-of-mind-lyrics>”, which did not include Alicia Keys as a featured artist. However in some cases, the “+” sign between the artists does matter, such as in the song 10,000 Hours by Dan + Shay and featuring Justin Bieber, which has the Genius link “<https://genius.com/Dan-shay-and-justin-bieber-10000-hours-lyrics>” which includes all involved artists.

After retrieving all the Genius urls, we decided to use the urls to perform a sentiment analysis on all of the songs, specifically classifying them as either positive, neutral, or negative, given the song's description. To do this, we decided to scrape all of the songs' descriptions under their "About" section of their Genius page using a function called `scrape_description`.

"TiK ToK" was Kesha's debut single, written by Ke\$ha, Benny Blanco, and Dr. Luke, the latter two doubling as the song's producers. Kesha had previously found success as the hook singer on Flo Rida's "Right Round," but this 2009 party anthem made her a star in her own right.

The song follows a group of friends through their night out: from getting ready and arriving at the party to swatting away unwelcome guys and dancing till the cops show up. "TiK ToK" might seem like just another aural glitter bomb, but underneath the beat-heavy electro-pop is a message about being confident and showing no shame in having fun. It turned out to be a message that hit with a lot of listeners.

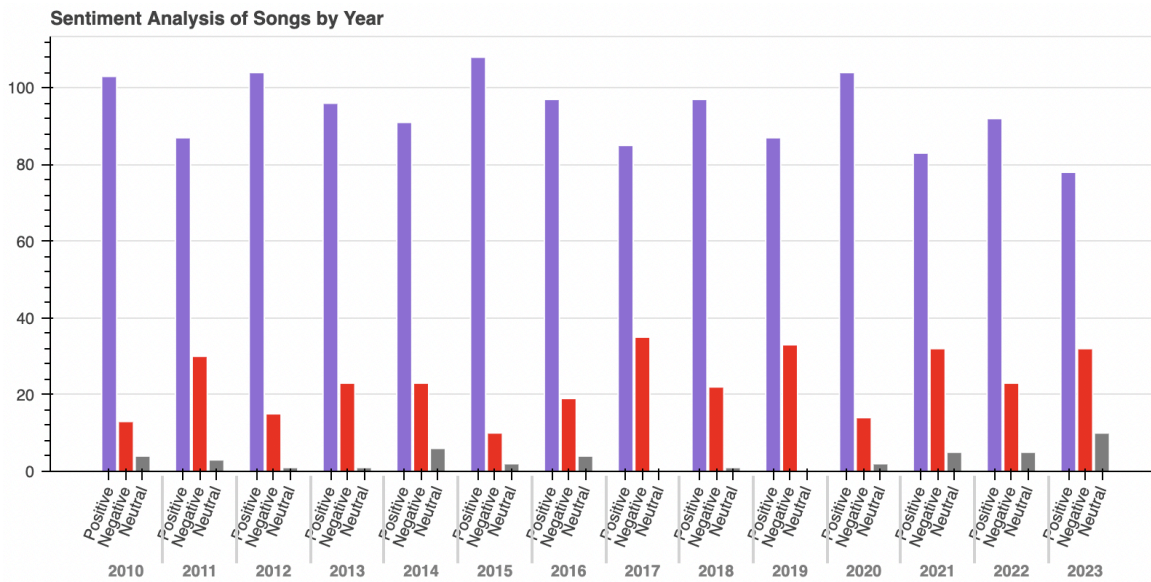
The song spent [38 weeks on the Billboard Hot 100] (<http://www.billboard.com/artist/305612/keha/chart?page=1&f=379>).

Example: Song Description of TiK ToK by Ke\$ha from the Genius website.

This scraping of the Genius website correlates to our goal of the project where we are trying to observe how songs of different genres, moods, and elements trend over time. To analyze the overall sentiment of the song, we used the "TextBlob" module in a function called `analyze_sentiment`, which creates a "TextBlob object" containing the song's description and calculates the sentiment polarity of that object. The sentiment polarity is measured on a scale of -1 to 1, with -1 indicating the most negative sentiment and 1 indicating the most positive sentiment. If the value is greater than 0, it's classified into the sentiment "Positive," if it's less than 0 it's "Negative," and if it's exactly 0 then it's "Neutral." For example, the song TiK ToK by Ke\$ha, which is described as a "party anthem," has a sentiment polarity of 0.1983, which indicates a positive sentiment.

Then, we iterated over each url in the `genius_urls` list to extract their song descriptions using the `scrape_description` function and extract their sentiment and polarity using the `analyze_sentiment` function. We added each url and their corresponding sentiment and polarity to a dictionary called `sentiment_ratings`. Lastly, we created a dictionary called `songs_by_year` to group all the songs by the year they were placed in the Billboard top 10, in order to create a plot using the Bokeh module and observe a trend in sentiments of songs over the past 14 years.

Figure 2: Sentiment analysis of popular songs by year from 2010-2023.



As we can see, in the past decade, there has generally been popularity among positive sentiment songs, with their sentiment count ranging from 80-100 (out of 120 total songs in the year). Negative sentiment songs happened to be more popular in 2011, 2017, 2019, 2021, and 2023 with an average sentiment count of 32, while they were the least popular in 2015, with a sentiment count of 10. As expected, there is less popularity with neutral sentiment songs, but they happened to be the most popular in 2023, with a sentiment count of 10.

VI. Quantifying Audio Features for Popular Songs

To understand how structural components of music evolve over time, songs can be decomposed into distinct audio features that help us quantify a song's characteristics based on audio signals. Classification of these audio features are extensively studied in music analyses and large music platforms like Spotify conveniently provide developers with these metrics to extend our understanding of music. However, with recent restrictions on directly using the Spotify API to access such information, we pivoted to exploring SongData.io, a database based on Spotify data with convenient access to different music metrics for over 80 million global songs. Although over 200 million songs exist globally and an ideal analysis would consider metrics for each of these songs, for the scope of this project we are considering only the most popular songs

of every month which are reliably included in SongData.io's database therefore making it a suitable source of data for our purposes.

To perform our analysis, we accessed our top 10 tracks for each month between 2010 and 2023 which have already been collected by scraping the Billboard Top 100 charts previously. Then, we proceeded to find the corresponding SongData.io page for each of these tracks by defining the function `get_songdata_url`. Since individual song page URLs were often composed of unique identifiers for each track (such as *"3USxtqRwSYz57Ewm6wWRMp"*) we decided to locate these pages by passing in a search query with the name of the song along with the artist(s) of that song instead. The search results often included our target page, but also a wide range of songs with not only identical names and artists, but also unofficial remixes of certain songs that were classified by the same name and artist information. Although remixes were often not returned as the top search results since these were automatically filtered by either the omission of the keyword "remix" in our search query itself or a specified remix explicitly mentioned in our search query, we found that when exact duplicate tracks were returned, accessing the first search result resolved the issue.

Now, for each track's page, we are able to extract the numerical metric for various audio features through our functions `get_feature_value` and `get_features_for_song`. `get_feature_value` collects the numerical value for each feature we observe in a track while `get_features_for_song` appends each feature value collected for a track to a consolidated dictionary by the name `features` for later reference. For our analysis, we considered the following features: acousticness, danceability, energy, instrumentalness, and BPM. Acousticness indicates whether the track is composed by classical versus electric instruments. Danceability indicates how danceable a track is. Energy indicates how intense a song is by considering factors such as tempo. Instrumentalness indicates whether a track is focused on instrumental versus vocal sounds. BPM, or beats per minute, is an indicator of how fast a track is. Unlike the previous metrics, BPM is not relative and so, it returns a distinct value as opposed to a percentage and so, for consistency, this justified conversion of this metric into a percentage based on the maximum BPM value of the songs we observed. Now, having the values for each audio

feature per track, we computed the average of these values across all the top 10 tracks of a month to produce a single averaged value per audio feature per month observed.

Based on these methods to access our desired data, we then transitioned into preparing all these values for visualization. First, we decided to create and update a comprehensive dictionary by the name `audio_data` whose keys were the month (in datetime format) and the values being a nested dictionary with our audio features and their corresponding averaged values. Then, we converted this dictionary into a dataframe, `df` (rows being the months, columns corresponding to each audio feature, and the cells being the computed average value) to prepare for our interactive visualization.

We chose to display an interactive line graph using *Pyplot* which tracks the audio features over time. This is optimal for the purposes of our project as it allows us to observe how specific features have changed over the past decade in terms of observable peaks and dips while also observing how the trends of different features correlate. Our graph allows users to select their preferred combination of audio features to display to allow for a customizable experience. A stationary version is displayed below.

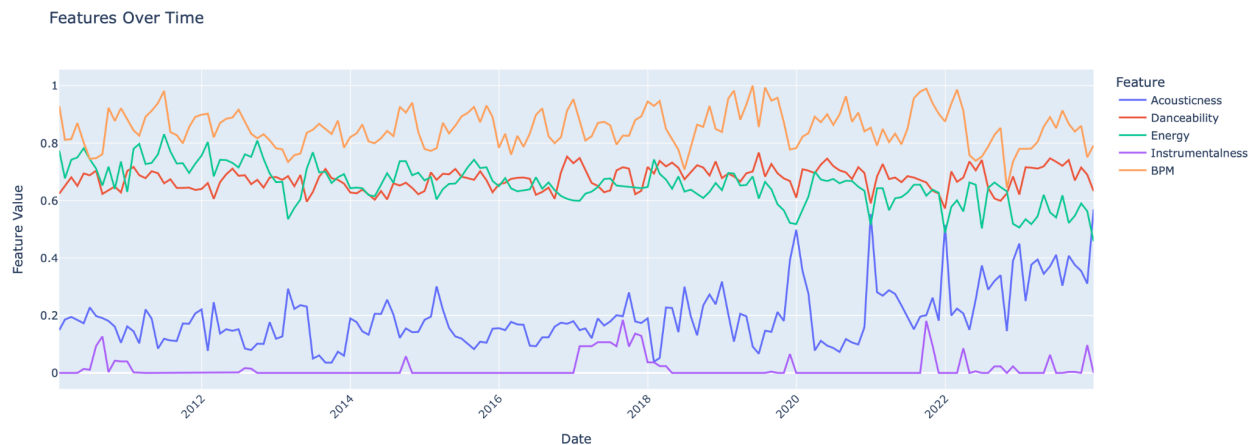


Figure 3: Visualizing the audio features of popular songs between 2010 and 2023.

First, let us consider the changes in overall trends and how fluctuations in feature values correlated with other features we observe. Recently, we can see an upward trend towards more acoustic sounds in music which indicates that popular songs in more recent years tend to be

composed using more classical instruments as opposed to more electronic sounds that seem characteristic to 2010s. We noticed how songs had higher energy in the first half of the 2010s compared to the second half of the 2010s. We also observe how danceability, energy, and BPM seem to show a positive correlation between each other where peaks in danceability values also correspond with peaks in energy values, and the same goes with the BPM values. This does logically translate well as more danceable songs tend to be more upbeat and energetic. Additionally, both danceability and energy seem to negatively correlate with acousticness where lower danceability and energy values correspond to higher acousticness. This also seems like a logical observation as acousticness tend to be higher when tracks use more classical sounds and classical instruments tend to be used for more melodic tracks which are often not as danceable nor energetic.

We can also consider characteristics of music that have stayed relatively stable over time. For example, instrumentalness has relatively low values consistently across the last decade which indicates that most popular songs have an emphasis on vocals as opposed to only instrumental sounds which can be supported by the fact that popular music and artists usually place an emphasis on songwriting and lyrical components. Even danceability showed consistent values which indicates that most songs that rose to popularity tended to be danceable in nature. Overall, future statistical analyses can further explore the significance of these correlations and consistencies, but our visualization aligns with our logical understanding and expectations of the observed features, thereby providing a foundation for future studies.

Overall, our methods for this analysis and visualization were successful in observing the different attributes of a song and applying these methods to understand how trends in music have evolved in the last decade. Our main challenge with this section was rate-limiting which randomly omitted on average 1-2 songs in our analysis despite efforts to periodically pause execution within our program. However, since our analysis is based on over 1600 songs, we decided that the omission of 1-2 songs would result in minor errors which reinforced our decision to calculate averages for each audio feature per month as opposed to a summation which would be more influenced by missing data.

VII. Conclusion

Through our analyses over genres, sentiments, and audio features of songs over the past decade we were able to observe several trends supporting our research question and hypothesis on how music has changed and evolved over time. Through our genre analysis, we extracted how certain genres like Pop and Hip-Hop/Rap are consistently popular among top songs across the last decade while some years showed a rise in other genres like Afrobeats. Within our sentiment analysis, we also found a stable trend of positive sentiment songs among the past decade, which is mostly due to the popularity of Pop songs that are usually upbeat. Finally, by our audio feature analysis, we observed correlations between certain features like danceability and energy as well as recent trends like increasing acousticness and decreasing energy over the past decade.

Based on our analyses and visualizations, we see value in potential extensions of this project to perform statistical analyses on these music trends and correlations to provide a more concrete and comprehensive understanding of music in the past decade. Although we limited our project to music from 2010 to 2023, we believe that valuable insights could also be gained by extending the time frame to include earlier music and observe generational differences in music and how the rise of music technology revolutionized the industry as a whole.