Spotify Streaming Audio and Behavioral Data Analysis

Motivation

Music is part of our daily lives. Valid and effective music suggestions can increase our content engagement level. With emerging on-demand digital music services, people can take control of what they listen to instead of passively perceiving the media from TV or radio.

By analyzing behavioral data from users, and features of the songs, the team aims to gain insights on what these listening sessions look like, and find common patterns in user-driven music curation, which could potentially lead to insights on what type of music should be recommended to users given various conditions.

Data Sources

Spotify Listening Sessions

The sessions dataset from the Spotify Sequential Skip Prediction Challenge, originally available as the Music Streaming Sessions Database. The data can be downloaded as a zip file from AI Crowd's official challenge page (https://www.aicrowd.com/challenges/spotify-sequential-skip-prediction-challenge/dataset_files). The dataset was generated by Spotify and contains ~130 million listening sessions and user interactions. The data is divided into a training and test set by default, with a total of 70 GB, in CSV format. Its key features include a track's order within each listening session, the date and hour-of-day when the track was played, and whether or not the user skipped the track before its completion. For this project, we used the data from the training set, which is about 50 GB compressed, and covers the period roughly between January and September 2018.

Spotify Track Features

Dimensional dataset containing attributes for the tracks in the Spotify Listening Sessions dataset. It is accessed via downloaded CSV files (https://www.aicrowd.com/challenges/spotify-sequential-skip-prediction-challenge/dataset_files), and contains observational and analytical measures for each track. This includes 1) basic descriptive attributes, such as Track ID, year released, and duration, 2) analytic measures, such as danceability, liveness, and valence. There are 3,706,388 unique tracks represented in the dataset.

Billboard API

This is based on an API that was originally offered by Billboard themselves to retrieve chart data from the Hot 100 song list. The official API has since been discontinued, but many platform-specific API's have been developed to replace it, most of which rely on simple web-scraping. **Billboard.py** is the most popular such API / library used with python. It takes a date as input and returns a ChartEntry object with attributes such as the Title, Artist, Peak Position, and Weeks on the Chart. The API was used to retrieve ~16,500 records. Information is available via the following url: https://github.com/guoguo12/billboard-charts. It can be installed via **pip install billboard.py.**

Spotify API

API provided by Spotify for querying their various databases. Information about the API is available on Spotify's developer page (https://developer.spotify.com/documentation/web-api/). The API can be accessed using the **spotipy** python library, information about which is available at the following url: https://spotipy.readthedocs.io/en/2.19.0/. Query results are returned as deeply nested dictionaries. The API was used to get track ids and audio features for the Billboard 100 songs, since that API identifies songs only by their title and artist. It can be installed via **pip install spotipy**.

Data Manipulation Methods

Sampling

Given the tools and processing power available, the team decided against trying to use the full dataset. Since the task was largely exploratory, we elected for the agility of a representative subset over the greater confidence of the full dataset. It is assumed that future work resulting from this exploration will be based upon the full dataset (perhaps an updated and expanded version), using higher powered resources and techniques. The work for this project was based on a 1% sample of the full Sessions dataset, which was generated using the pandas sample() function and then saved to CSV.

Relationships

The data that were used came from 4 different proximate sources—that is, while most of the data ultimately came from Spotify, we encountered it as a set of alienated CSV files and runtime API calls, the relationships between which had to be inferred and executed. In the end, the relationships were between those fields that identified a particular Spotify track, namely the “track_id” or sometimes “clean_track_id.” We were left to wonder under what circumstance the track id needed “cleaning.” Likely, “clean_track_id” was derived from the Spotify.com URL that navigates a listener to the song itself, the last part of which is the song’s unique id:

https://open.spotify.com/album/1EK3a0Yctg4d3nGQzE4Uty?highlight=spotify:track:**5SAUIWdZ04OxYfJFDchC7S**

All data from the Spotify API came with an appropriate formatted track id. Calls to the Billboard API, however, identified only a song’s title and artist. The Billboard song data was subsequently passed to the Spotify API as a title search, in order to associate it with a standard track id.

Because of the size of the Sessions dataset, join operations were often executed iteratively, upon subsets of the data. Depending on other active objects in memory, simply joining the full Sessions and Audio Attributes dataframes was enough to crash Collab, requiring a frustrating trip back to the drawing board. It ended up being much more efficient to iteratively subset the data, join these smaller dataframes, select relevant columns, perform transformations, and then store these results in a list or at least a much smaller dataframe.

Missing Data

The Sessions dataset, in particular, seemed to have been cobbled together from multiple fragmentary datasets. As a result, there were sporadic occurrences of missing data. In most cases, we selectively removed this data based on the specific analysis being executed. For example, if we were conducting an analysis that counted rows by date, we removed rows with missing dates. This proved a more useful approach than the wholesale deletion of every row with a missing value in one of its columns.

Data Conversion

Most of the fields we used for analysis or visualization were already the correct data type. However, since most of the analyses included a time component, it was necessary to make sure dates and times were explicitly stored as datetime objects. These objects were used to derive more general slices of time, such as Years, Months, Days, Hours, etc..

Aggregation

Since much of the project was focused on listening patterns across times of day, we most often grouped on the "hour_of_day" field within the Sessions dataset. Aggregation functions or transformations were then applied to the various audio features. Often these were simple means, but in some cases the goal was normalization or standardization, as discussed in the next section.

Feature Engineering

Based on questions that arose during the team's discussions, a number of features were appended to the existing datasets, including the

- **Standardized Audio Attributes**
  - Because audio attributes often varied widely across hours of the day, the attributes were converted to z-scores for some analyses and visualizations
- **Billboard Status**
  - We became interested in the relationship between a track's prevalence in the Sessions dataset and its commercial popularity. Accordingly, we sought to identify each song in the Billboard Hot 100 chart by its audio features and see if we could link them to the corresponding Spotify tracks.

Analysis and Visualization

Approach

Initially the team attempted to divide tasks functionally, as would be the approach in most professional contexts. Each of us would focus on a certain type of task. One of us might handle data cleaning and manipulation. Another might focus on visualization or analysis. Someone else might draft the final report. However, this approach proved clunky and unnatural for a number of reasons. First of all, since many of these tasks are sequential, this makes it difficult to work in parallel. It would be fairly ridiculous for Josh to work on data cleaning in week 1, Arthur to work on analysis in week 2, and Judy to tackle visualization in week 3. Second, since we are all taking the same classes and learning the same skills, we are all equally enthusiastic about applying them. Third, it seemed more advantageous to harness each member's sincere interest in different aspects of the project. As a result, and after discussing as a team, we each focused on a set of particular questions or topics that interested us.

The main disadvantage of this approach is a lack of standardization. Because we all use different programming idioms and visualization packages, it can be difficult to assemble a cohesive finished product. However, the advantages seem to outweigh the disadvantages. In essence, we each became experts in several "mini-domains," acquiring and leveraging thematic knowledge as we progressed.

Questions to Answer

The primary analyses ended up being roughly divided into 3 areas:

1. **Which audio features do listeners generally favor?**
2. **How do the audio features change across time?**
3. **How does listener behavior change across time?**

Which audio attributes do listeners generally favor?

Looking at mean song features across the dataset, some clear preferences emerge.

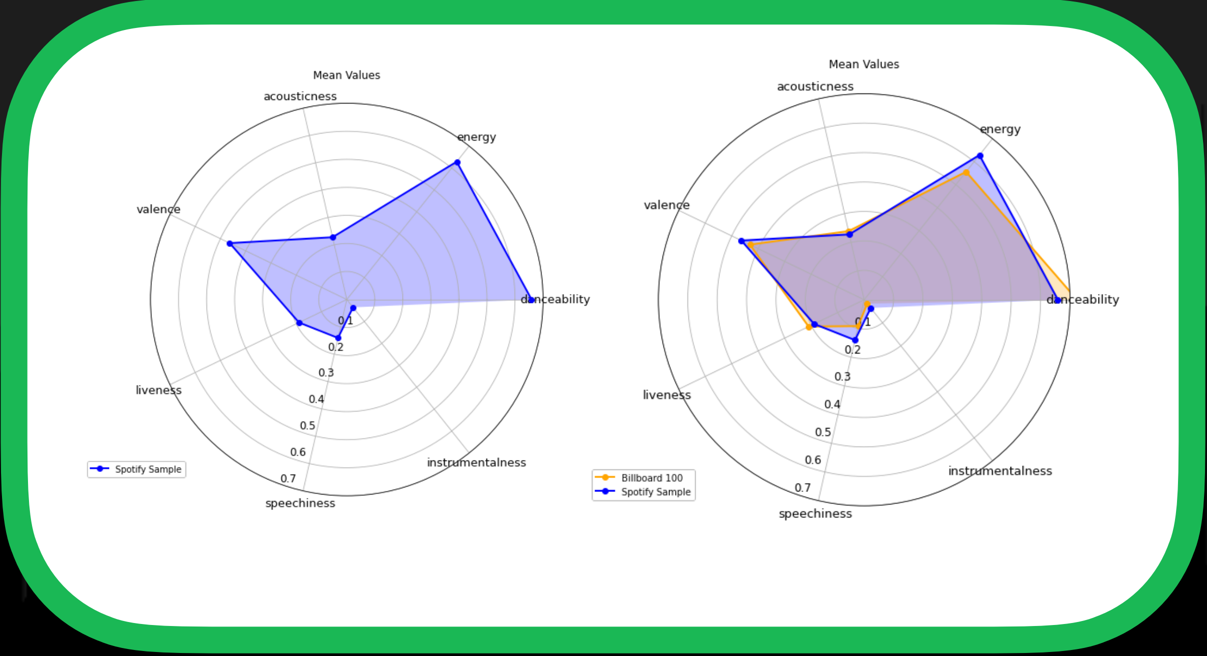
The data as a whole suggests a preference for “upbeat” tracks. Energy and Danceability score much higher than other features, with Valence (an analog for “happy-sounding”) not far behind. Since the data are unfiltered, and we do not yet know which tracks were skipped, we cannot say if this preference reflects choices of the listener or merely Spotify’s algorithm.

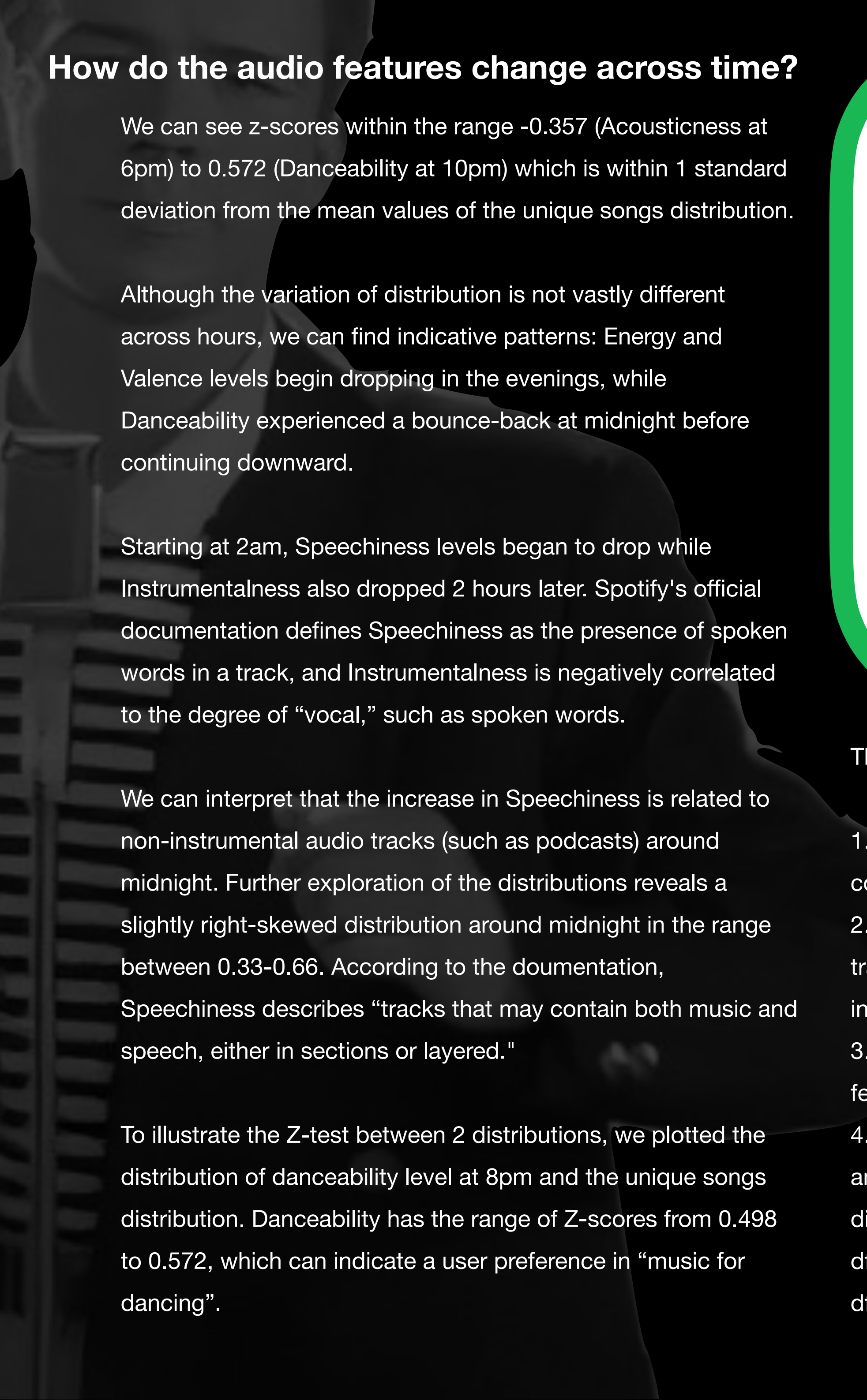
We were also curious how this effect would compare to songs from the Billboard Hot 100 chart. Perhaps unsurprisingly, the pattern is extremely similar, and seems to confirm a preference for happy, vivacious, and danceable music. It is unsurprising that these patterns are so similar since, as it turns out, top 100 songs account for the most of the tracks played. In fact, more than 80% of the entire dataset is comprised of 14 songs that were all, at some point, #1 on the Billboard Hot 100 list.

These charts required all four datasets and several transformations:

1. Call billboard.py API for every date in the Sessions dataset to return a list of the top 100 song titles for each date.
2. Call the Spotify API and search for each unique billboard song title retrieved in the previous step. Return the Song ID.
3. Call the Spotify API again, passing the previously retrieved Song ID and returning each song’s audio features.
4. Merge the Sessions and billboard dataframes by joining on the audio feature values themselves. This was not expected to work, but was surprisingly effective (values had to be rounded to 4 decimal places).
5. Append columns to the Sessions dataframe that identify play session tracks as songs from the Billboard Hot 100.
6. Take the mean for all tracks and for Billboard identified tracks.
7. Plot means

Billboard #1 Songs	Plays	% of Total
God's Plan	5,401,110	24.81%
Nice For What	4,796,790	22.03%
Sad!	4,078,232	18.73%
In My Feelings	2,832,750	13.01%
Girls Like You	1,115,014	5.12%
I Like It	1,041,792	4.79%
This Is America	102,613	0.47%
Perfect	96,552	0.44%
Shape Of You	20,020	0.09%
Look What You Made Me Do	6,976	0.03%
Starboy	4,074	0.02%
Despacito	3,211	0.01%
Locked Out Of Heaven	3	0.00%





How do the audio features change across time?

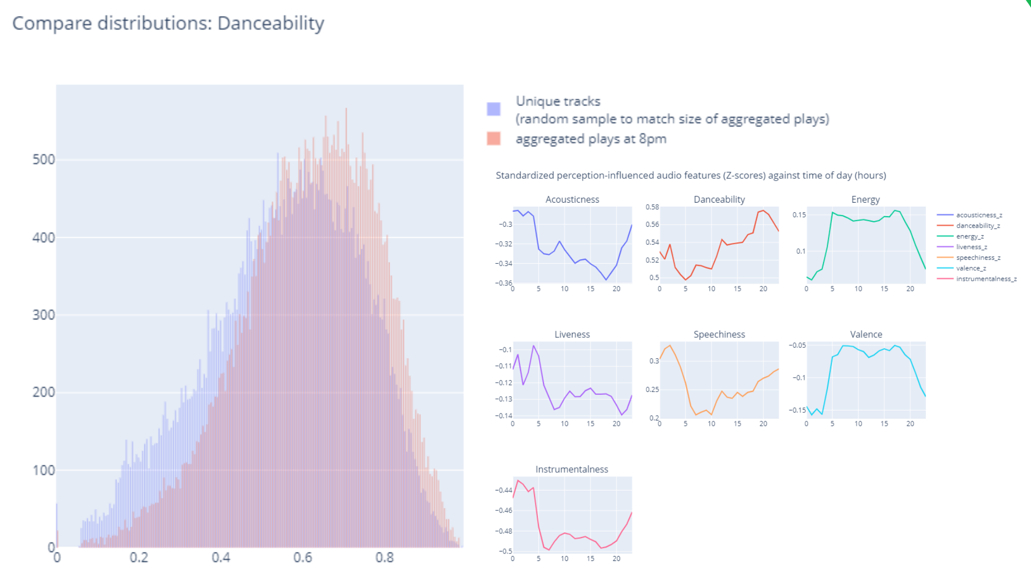
We can see z-scores within the range -0.357 (Acousticness at 6pm) to 0.572 (Danceability at 10pm) which is within 1 standard deviation from the mean values of the unique songs distribution.

Although the variation of distribution is not vastly different across hours, we can find indicative patterns: Energy and Valence levels begin dropping in the evenings, while Danceability experienced a bounce-back at midnight before continuing downward.

Starting at 2am, Speechiness levels began to drop while Instrumentalness also dropped 2 hours later. Spotify's official documentation defines Speechiness as the presence of spoken words in a track, and Instrumentalness is negatively correlated to the degree of “vocal,” such as spoken words.

We can interpret that the increase in Speechiness is related to non-instrumental audio tracks (such as podcasts) around midnight. Further exploration of the distributions reveals a slightly right-skewed distribution around midnight in the range between 0.33-0.66. According to the doumentation, Speechiness describes “tracks that may contain both music and speech, either in sections or layered."

To illustrate the Z-test between 2 distributions, we plotted the distribution of danceability level at 8pm and the unique songs distribution. Danceability has the range of Z-scores from 0.498 to 0.572, which can indicate a user preference in “music for dancing”.



Full Visualizations/ Code Snippets

The overall process involved the following steps:

1. The dataframe of music listening sessions was further sampled due to computation resource constraints, then we filtered out songs that were skipped at the beginning / only played briefly.
2. Inspected distributions of audio features from the audio features dataframe, which includes all tracks from the original dataset-- about 3.7 million tracks in total-- in which the distributions are not influenced by how users play the songs (distribution of all unique songs).
3. For each hour of the day, merged the 2 dataframes and generate average values of the audio features.
4. For each audio feature, apply standardization techniques (Z statistics) between hourly distribution and the distribution of all unique songs, such that we understand how the hourly average values differ from mean values of the entire distributions of all unique songs, for example:
df_audio_features_hourly['instrumentalness_z'] = (df_audio_features_hourly['instrumentalness'] - df_track_features['instrumentalness'].mean())/df_track_features['instrumentalness'].std(ddof=0)

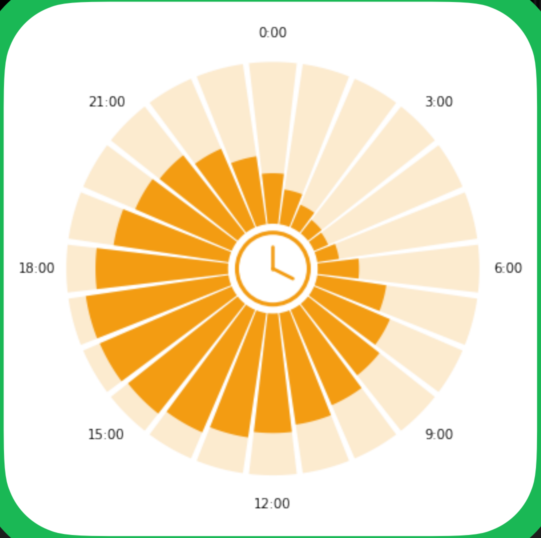
How does listener behavior change across time?

Here we try to analyze what time listeners use Spotify more regularly than others.

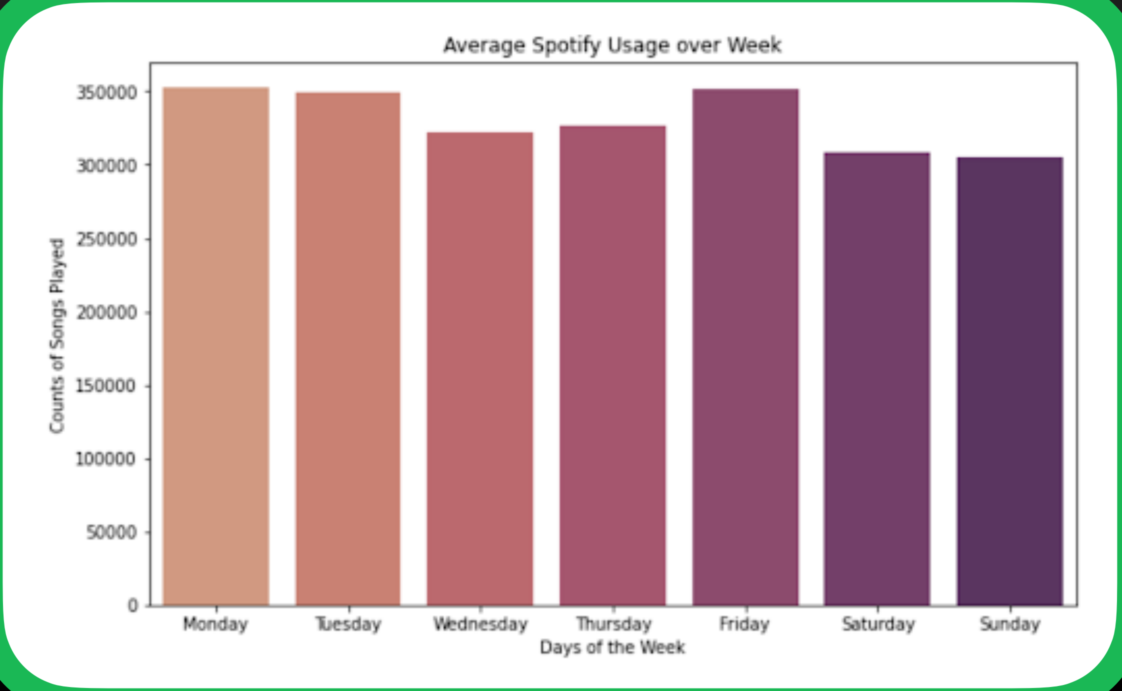
From these plots, we can see that the music listening time trend follows people's daily working and resting routines.

According to these graphs, we can see patterns in average usage:

1. Maximum around 3-5 PM by local time
2. Minimum around 2–4 AM by local time
3. Maximum during workdays
4. Minimum during weekends



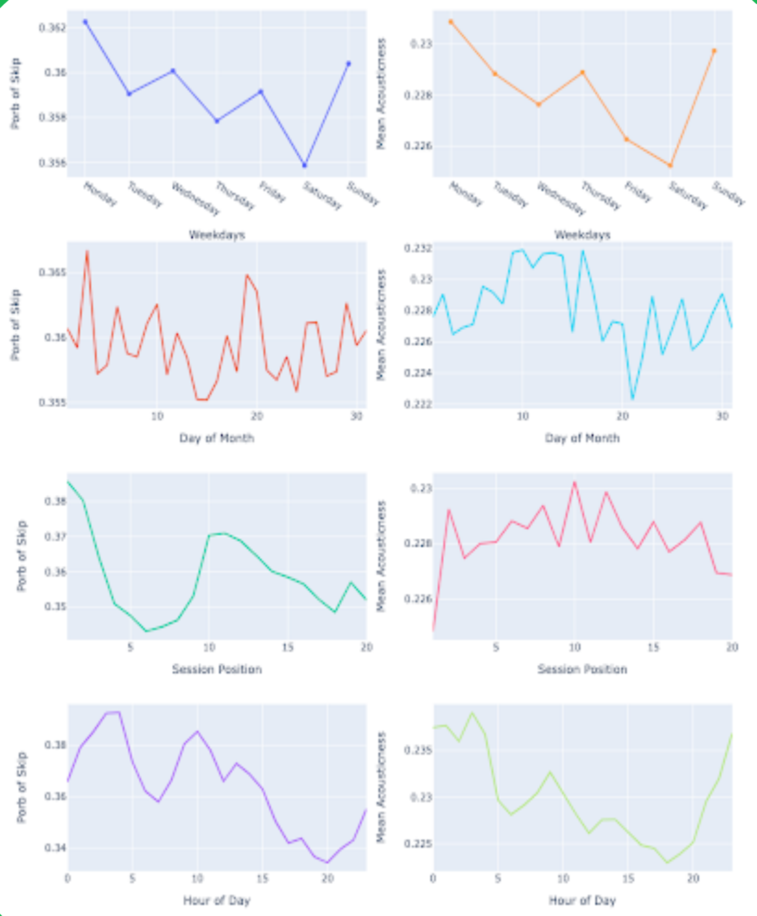
User listening time by hour of day



User listening time by weekdays

Skipping Pattern Analysis

Spotify doesn't have a dislike button, so the relationship between time-related song skipping patterns and context-related features such as Acousticness are the subtle cues we are interested to learn.

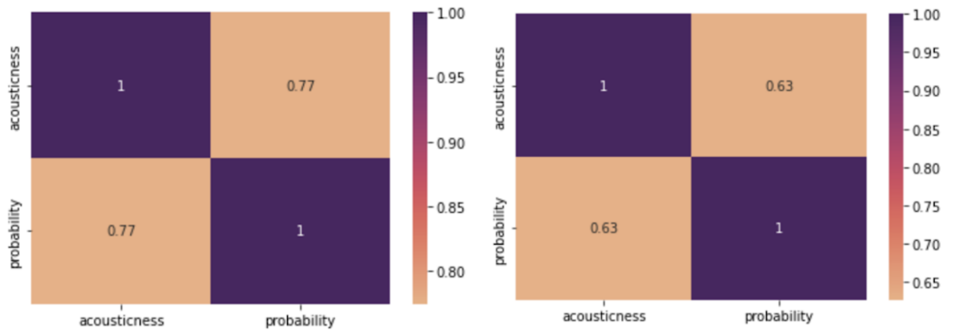


To derive this analysis, we use a "DatetimeIndex" to identify the event dates and the split-apply-combine method to calculate probabilities of song skipping on weekdays, days of the month, hours of day, and within a single session. We adopt the same process for the Mean Acousticness analysis.

The probability of skips remains high during weekdays, which follows the same pattern as the Mean Acousticness plot. Users are more likely to skip songs at the beginning and in the middle of a single session. (user sessions are between 10 and 20 tracks) . The probability of skips by day of month might be an apparent effect due to weekday and time of day effects.

We apply the Pearson corr method to find the correlation between the probability of skipping by the hour of the day and Acousticness and find a coefficient of 0.63. The coefficient for the probability of skipping by weekdays and Acousticness is 0.77. These coincide with our assumptions.

*Acousticness: A confidence measure from 0.0 to 1.0 of whether the track is acoustic.



Statement of Work

Arthur:

- Set up collaborative coding environment
- Conducted data analysis and visualizations based on the question “How do the audio features change across time?”

Judy:

- Optimized computation efficiency for dataframes
- Conducted data analysis and visualizations based on the question “How does listener behavior change across time?”

Joshua:

- Gathered dataset documentation and set up APIs
- Conducted data analysis and visualizations based on the question “Which audio features do listeners generally favor?”