INFO-I590 Data Visualization
Fall 2023

From Playlist Comparisons to Shared Recommendations Through Visualization

Team Visual Tunes – Ayush Khaneja, Laxmikant Kabra, Pooja Hegde, Radhika Agarwal

ABSTRACT

Spotify is the world's most popular audio streaming subscription service [1] with over 574 million users (226 million subscribers) across more than 180 countries. It is a rich source for data, and this data can be leveraged to do various analyses. This project aims to explore data from two different playlists, compare how different or similar the two playlists are, identify some common trends in the playlists, and make recommendations that the listeners of each of the two playlists can enjoy, all through simple and intuitive visualizations. The project explores both static and dynamic approaches to visualizing the data, and finally presents an interactive dashboard that lets users compare two playlists, get insights about their music tastes, and get new recommendations based on common or similar interests.

1. INTRODUCTION

1.1 MOTIVATION

The major motivation behind this project is the Spotify "Blend" feature [2]. It is a unique recommendation system based on multiuser personalization. It allows multiple users to collaborate on a shared playlist and then generates recommendations based on shared interests. However, it does not have a visual aspect to it. It just reports the similarity percentage between two playlists and a song that both users listen to. It would be great to supplement this shared recommendation approach with some visuals. The potential for discovering new music through visualizing relationships between artists and genres is substantiated by studies in music recommendation systems. There are studies supporting the idea that visualizing user preferences can enhance music discovery [3].

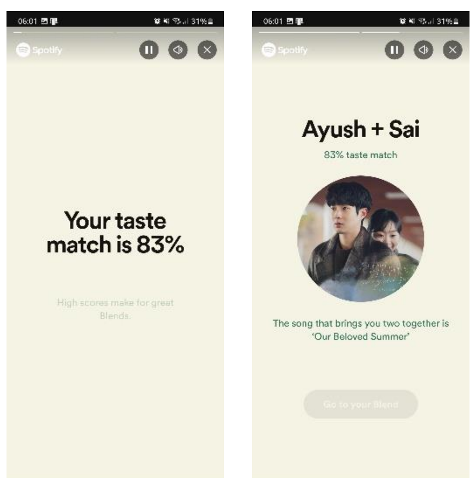

Figure 1: Current Spotify Blend Visualization (Images from own profile)

Another motivating factor for the project is to make a tool for comparative analysis of two playlists. This tool could be used to compare musical tastes between two friends (compare personal playlists). It could be used to compare top music from two years (eg: top tracks from 2019 vs. top tracks from 2023) by comparing top artists, genres, and song features like energy, acousticness etc. In a similar fashion, it could also be used to compare listening choices of two different countries (eg: top tracks in the USA vs. top tracks in India). All in all, the motivational is to build a flexible tool that can compare trends in music choices.

Aside from all this, there is the motivation to explore a new class of data in musical data, and analyze the data through various visualizations. The idea is to see how well do the commonly used visualizations like scatterplots, bar plots etc. work with this type of data, and find out if there are specific visualization techniques that are more suitable to this kind of data.

1.2 EXISTING WORK

1.2.1 Stats for Spotify

One of the most popular platforms to assess Spotify trends is “Stats for Spotify” [4]. It is a simple website that displays top tracks, top artists and top genres for different time periods. Although it is simple and easy to interpret, no deeper insight is gained from the visuals. The visuals are just names of tracks or artists, and a bar plot for genres.

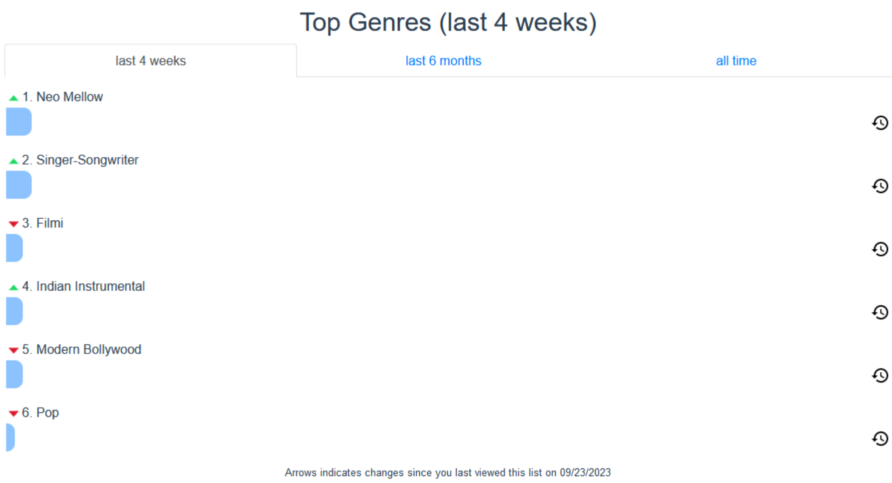

Figure 2: Bar plot showing top genres in ‘Stats for Spotify’

The above bar plot has an issue with a lack of a scale. There is no indication of what the length of the bars represents. This is slightly confusing. The visualizations work on only one user profile and do not allow a chance for comparison, nor do they recommend new music.

1.2.2 Spotify Pie

It is a simple pie chart [5] showing the top genres and has a list of top artists at the bottom. It also gives users to download the generated chart as an image file.

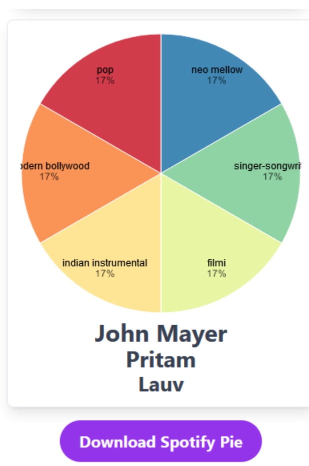

Figure 3: ‘The Spotify Pie’ interface

The list of artists is there without any context, and the numbers in the pie chart do not even add up to 100% This is a very confusing and misleading visualization, and again, does not provide any interesting information about the data. Again, like the previous work, the visualization only works on one Spotify profile with no comparison options and no recommendation features.

1.2.3 Obscurify

“Obscurify Music” [6] is an interactive dashboard with a lot of interesting features. The main aim is to quantify and compare how obscure a person’s tastes in music are to the general trends of a selected country. It interactively provides insight on top tracts, artists and genres, and has some other very interesting and insightful visuals as well. Some of them are shown below.

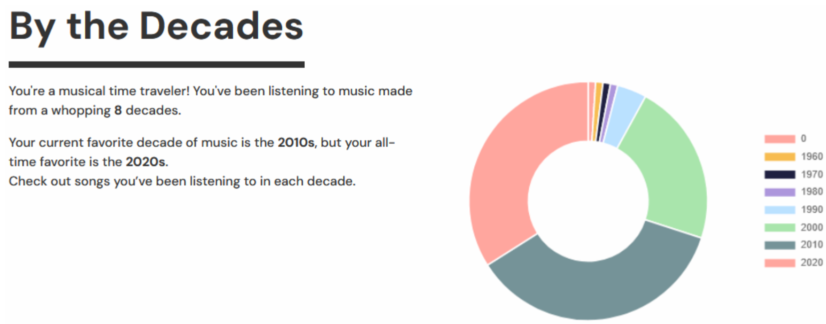

Figure 4: Donut plot showing number of songs for each decade

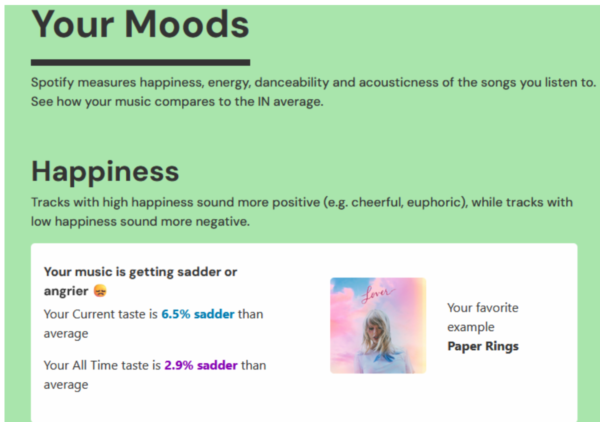

Figure 5: Comparing user moods

The website even has an option to share results and compare with other users, but the comparison is not depicted visually, and no new combined trends are extracted. The site also generates recommendations and gives an option to save them into a new playlist, but the recommendations are based on a single user's music choices, and the attributes of the recommended songs are not visually explained. Also, it is a very text-heavy tool with more written insights than visually explained ones. Even so, it presents a very complete picture of a person's taste in music, it is very interactive, and does a good job of reporting trends.

1.2.4 Musictaste.space

"Musictaste.space" [7] is another highly interactive platform that let's users view insights about their own listening habits, and allows for comparison with other Spotify users. It also presents general trends, along with trends for moods. This platform uses the obscurity scores from "Obscurify" to report them as an insight. It also has the option to compare two playlists.

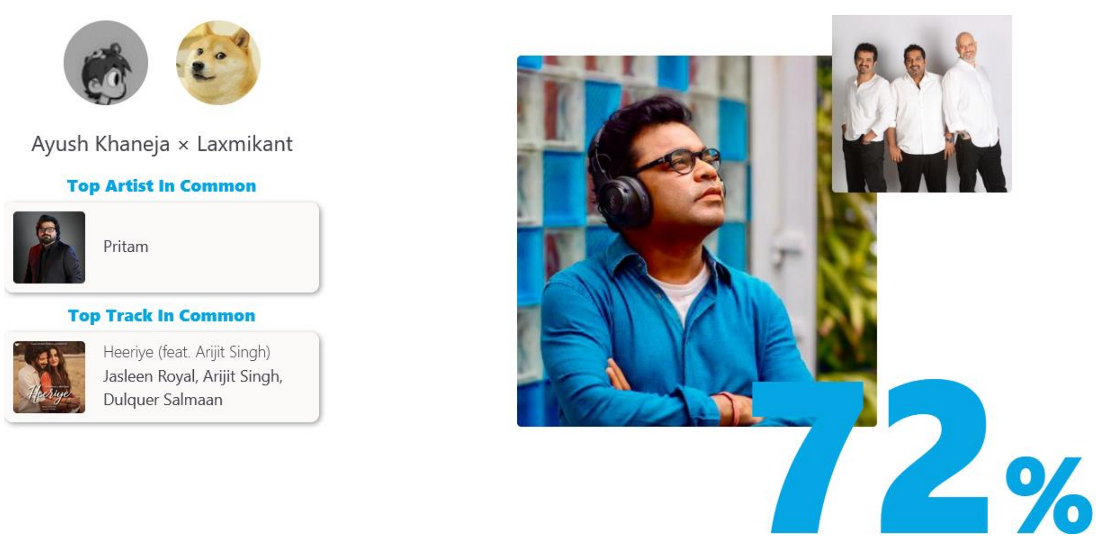

Figure 6: 'Musictaste' data comparison

It generates a playlist based on shared interest but does not visualize the new data. This platform is also very text-forward, and does not contain many visualizations. It comes the closest to what we want from a tool, but falls short with the types of visualizations we get. There is interactivity, there is comparison of data, and there are recommendations based on shared interests, but no visualizations to explain the numbers and logic behind the final results. Also, this tool does not allow the flexibility to choose the playlists that we compare. Because of this, it remains limited to a personal analysis tool only, and cannot perform analysis for any other scenarios.

1.3 CONTRIBUTION

The existing works discussed above do not contain all the features we are looking for, namely:

- Compare any two playlists and visualize the similarities / differences in their musical attributes
- Gather common traits and present them as visuals
- Generate recommendations based on the visualized common traits
- Present the newly formed playlist through visualizations
- Visualize the musical features of the new playlist

We want to give users the flexibility to decide the context of the comparative analysis, and not just rigidly compare two personal profiles to study personal tastes. If users want to study the change in musical preferences from 5 years ago to now, they can easily do that by getting the 'Top Songs' playlists for the particular years that Spotify generates every year. To compare musical tastes between two countries, a similar strategy can be deployed by using Spotify-generated playlists for the most popular songs in each country. Another use case could be to compare the musical features for different years or decades. The musical features can be studied to explain the general vibes of the songs prevalent in those years, and use the recommender to get new songs with the same vibes. It could also be used to get music that is a mix of the styles of two artists. These are just some use cases that our project can handle.

The existing works also do not explore the artist network that gets formed between all the artists from the two playlists through their collaborations with one another. We analyze that aspect of the playlists as well. It might not be a very useful feature for all use cases but it could lead to some very interesting results in many cases, and it would be a great way to discover unlikely collaborations and tracks that are not too popular.

Lastly, the existing methods either had visualizations that made very little sense, or reported on the trends found in the data in a majorly textual format. We want the focus of the entire analysis to be the visualizations. We want to build visualizations that are meaningful, easily interpretable, and also aesthetically appealing. Also, we don't want the graphs to be interactive just for the sake of interactivity. We want the interactions to add another layer of information to the plots.

2. DATA

Data gathering is handled through the use of the Spotify API. It was a fairly easy and direct process. A pipeline was set up to get playlist hyperlinks, get the playlist data from Spotify, perform the comparative analysis and generate the recommendations. The columns gathered in the data are explained below:

- **Track_Name**: Song name
- **Track_ID**: Spotify ID for the song
- **URL**: Web URL for the son
- **Artist(s)**: All artists associated with the song
- **Genres**: Genres associated with the artists
- **Album**: Album that the song is part of
- **Song Popularity**: song popularity score (0-100), based on no. of streams
- **Artist Popularity**: artist popularity score (0-100), based on no. of streams
- **Duration**: Song runtime
- **Tempo**: Approx. beats per minute
- **Loudness**: How loud a song is (dB)
- **Key**: 0-11, one of the 12 notes on a musical scale
- **Mode**: Major or minor
- **Danceability**: A probabilistic measure of how danceable the song is
- **Energy**: A probabilistic measure of how energetic the song is
- **Speechiness**: A probabilistic measure of the presence of spoken words in the track
- **Acousticness**: A probabilistic measure of how acoustic a song is
- **Instrumentalness**: A probabilistic measure of the vocals in the song
- **Liveliness**: A probabilistic measure of the presence of a live audience
- **Valence**: A probabilistic measure of how positive the song is
- **Time_Signature**: The time signature (beats in a measure) of the song

3. VISUALIZATION METHODS

Before arriving at the final visualizations, a lot of trial and error was done. However, there were always some design choices in mind for the visuals.

a.
When trying to analyze any single continuous variables (tempo, energy, acousticness etc.), the idea is to use either density plots or histograms overlayed one on top of the other to compare and contrast. Another contender is CCDFs or ECDFs.

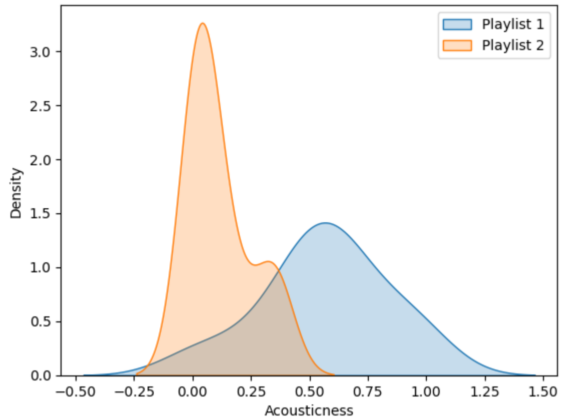

Figure 7: Test plot to compare a single continuous variable

b.
When trying to analyze more than one of these continuous variables together, the idea is to use a scatterplot. The axes can represent two of the features being studied, while size can be controlled by a third one. The color can represent either a fourth variable, or which playlist the data is coming from. This can be used to compare two playlists, or analyze the new playlist created from song recommendations.

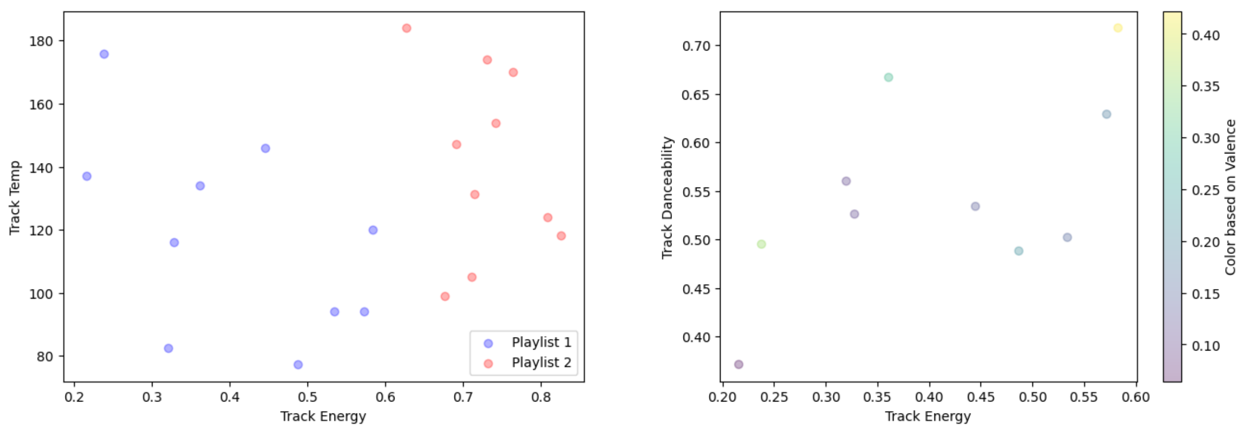

Figure 8: Test plots to compare multiple continuous variables together

c.
Another way to look at all important continuous variables together is using a spider plot or a radar plot. It would give an aggregated value of all the continuous variables and make it easy to compare the datasets based on the selected variables.

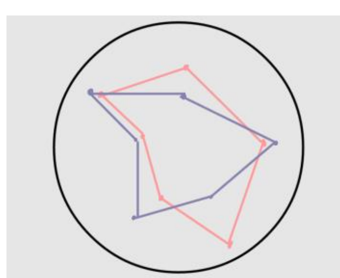

Figure 9: Rough sketch for proposed spider plot

d.
When analyzing common artists or genres, a bar plot can be used. Alternatively, a bubble plot can also be used. The bubble plot can incorporate multiple visual encodings into one. On the other hand, a bar plot is easier to interpret, and can be sorted to show order. Side by side bar plots can also show individual top artists or genres from playlists.

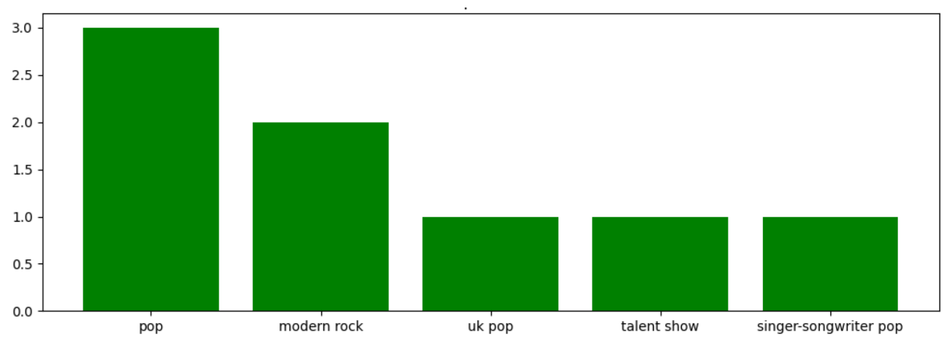

Figure 10: Test plot for top genres

e.
To understand collaborations between artists, a network map can be used. Edges can be weighted to encode a measure of the data. Absence of an edge means no collaboration. Nodes can also incorporate multiple levels of information through multiple encodings like size and color. Color could represent which playlist the artist comes from, and size could show artist popularity.

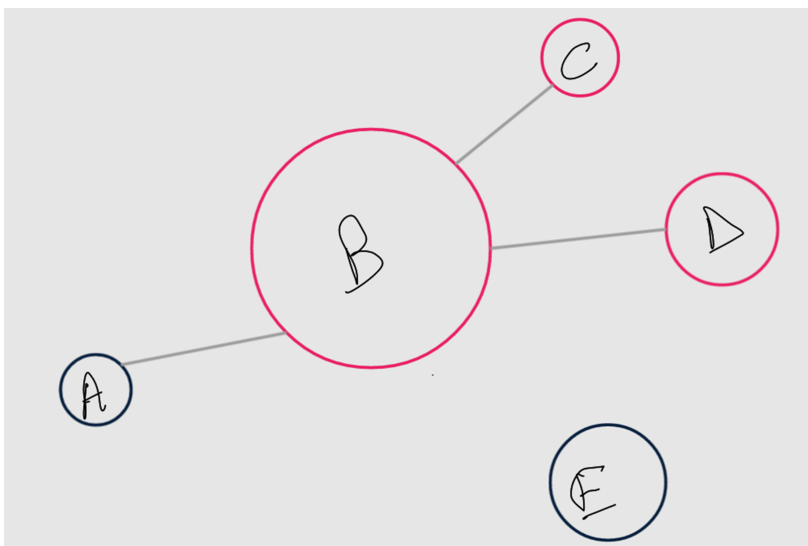

Figure 11: Sample sketch of the collaboration network

f.
To show recommendations, either by common artist or common genre, a treemap can be used. The hierarchy levels can be as follows:
Artist / genre → song
Sizes can be based on popularity; colors can be based on artist/genre.

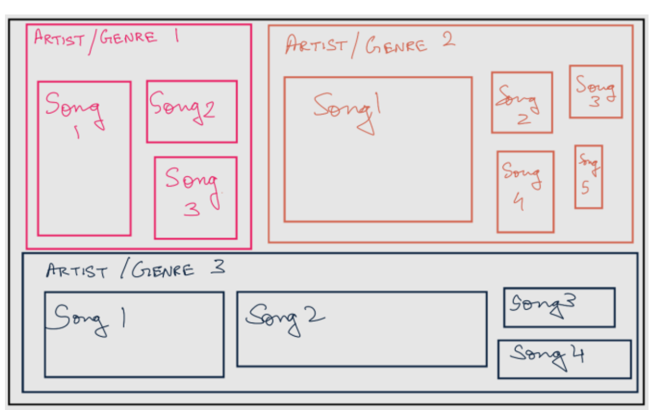

Figure 12: Sample sketch for treemap

The initial phase of plotting was done to make static graphs. The following results were obtained:

1. The plot compares the top 5 most frequent artists in two playlists, showcasing their relative occurrence percentages. The histograms display the distribution of artist frequencies, with Playlist 1 on the left and Playlist 2 on the right.

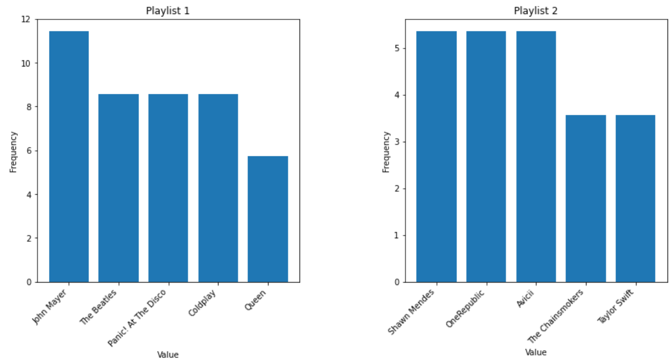

Figure 13: Top artists in playlists by song frequency

2. The plot displays the top 5 music genres and their frequencies in two playlists (Playlist 1 and Playlist 2). Each bar represents a genre, sorted in descending order by frequency. The x-axis shows the genres, and the y-axis indicates the frequency of each genre in the respective playlists. The two subplots allow for a visual comparison of genre distribution between the two playlists.

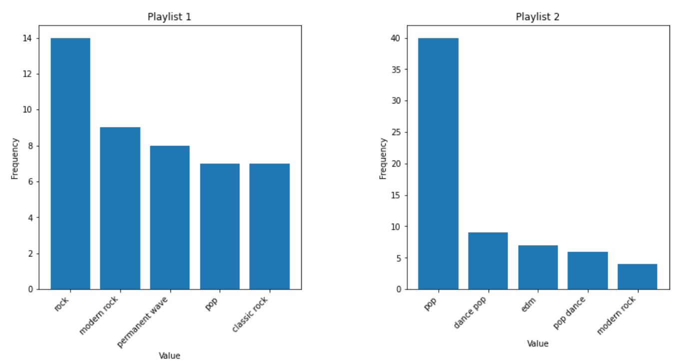

Figure 14: Top genres by frequency

3. The plot shows the popularity distribution of songs in two playlists (Playlist 1 and Playlist 2). Each subplot displays a histogram with 20 bins and a kernel density estimate, illustrating the frequency of songs at different popularity levels. The visual comparison allows for an easy assessment of how the popularity of songs is distributed in each playlist.

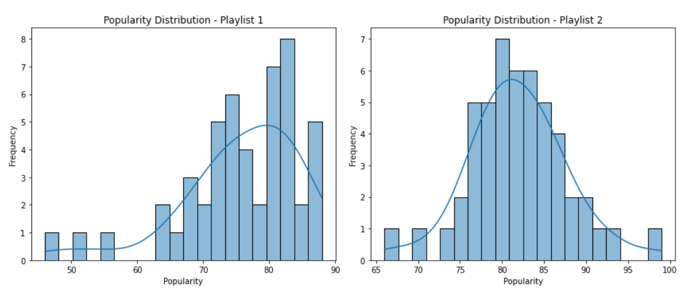

Figure 15: KDE with histogram for popularity distribution

4. The plot compares the relationship between song Popularity and Energy for two playlists (Playlist 1 in blue and Playlist 2 in green). Each point represents a song, with its position indicating both its Popularity and Energy values. The visualization aims to illustrate any patterns or differences in the distribution of these features across the two playlists.

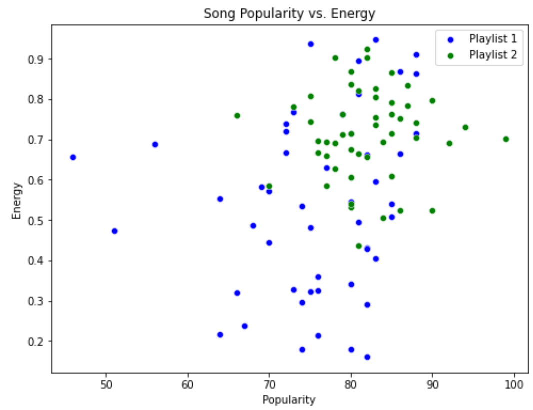

Figure 16: Scatterplot of multiple variables for the two playlists

5. The plot represents the distribution of song durations in Playlist 1 and Playlist 2. The x-axis denotes the duration in seconds, while the y-axis indicates the frequency of songs within each duration range. The visual comparison allows for a quick assessment of the differences in song duration distributions between the two playlists.

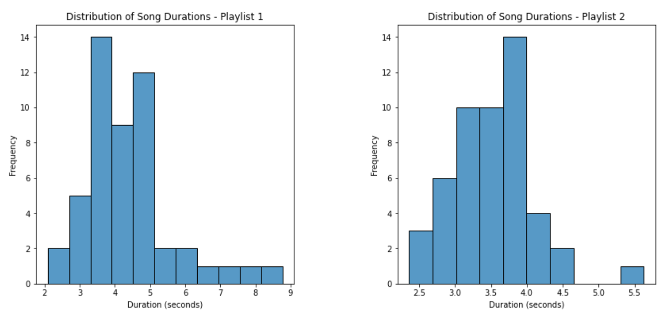

Figure 17: Histogram distribution to show song durations in playlists

6. The parallel coordinate plots compare the song features (Danceability, Energy, Loudness, Speechiness, Acousticness) across varying levels of popularity for Playlist 1 and Playlist 2. Each line represents a song, and its position on each axis indicates its feature value. The color intensity reflects the song's popularity, allowing for a visual assessment of feature patterns in relation to popularity within each playlist. (Experimental, alternative to spider plot)

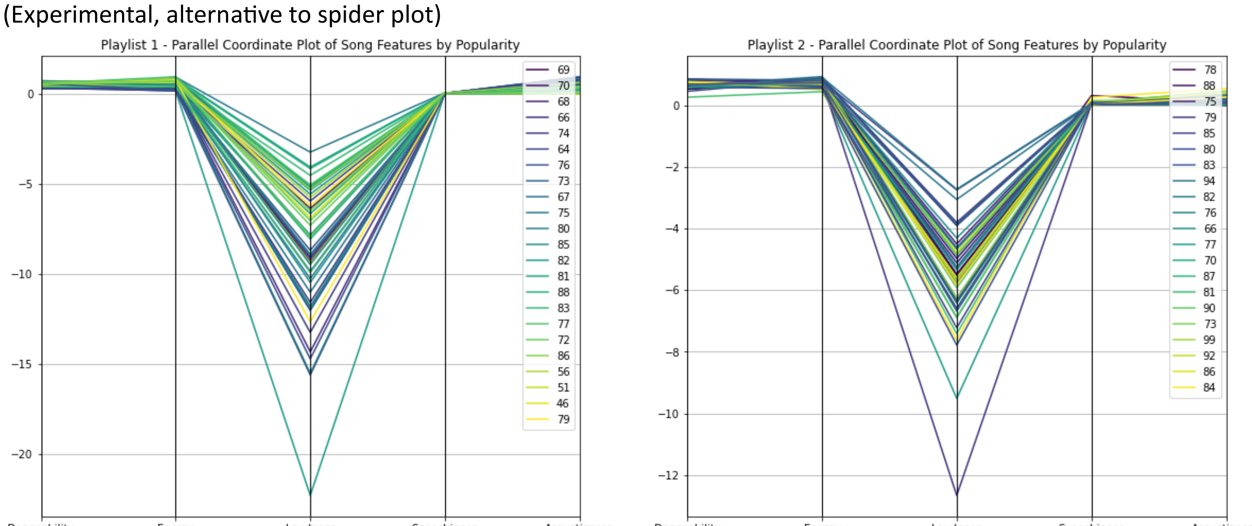

Figure 18: Parallel coordinate plots for multiple variables

7. The code creates a pair of treemaps using the **squarify** library to visually represent the genre distribution of artists in two playlists (Playlist 1 and Playlist 2). Each square in the treemaps corresponds to a genre, with the size of the squares proportional to the number of artists in that genre. The plots provide an overview of the diversity and distribution of genres within each playlist.

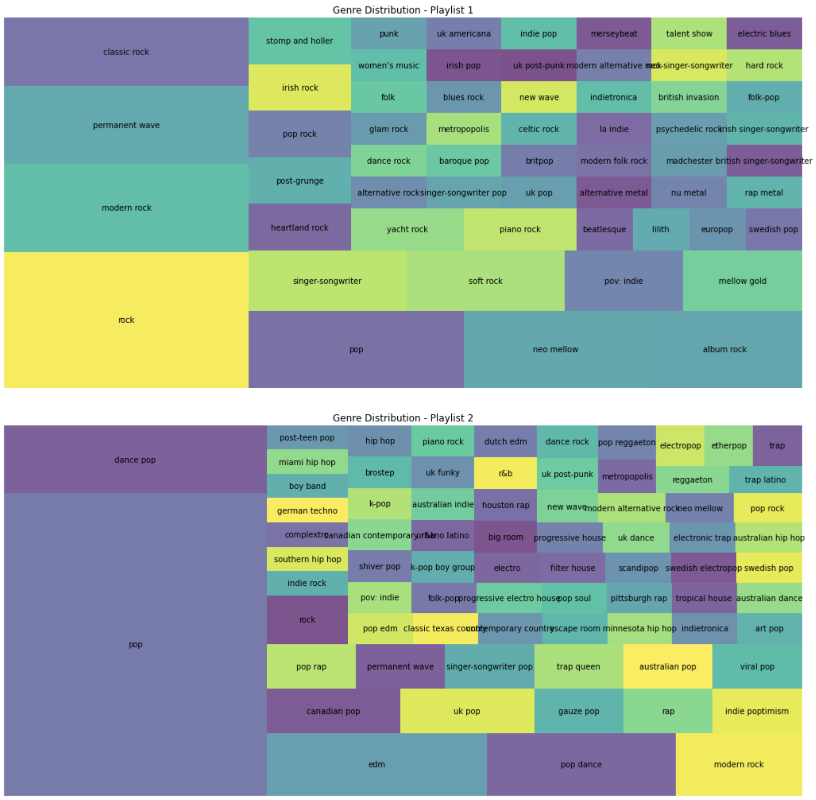

Figure 19: Treemaps for genre

8. The plot consists of two word clouds representing the most common music genres among artists in Playlist 1 and Playlist 2. Each word in the cloud corresponds to a genre, with the size indicating its frequency. The visual representation offers a quick overview of the genre distribution in each playlist, helping to identify the predominant music genres within the respective collections. (Experimental)

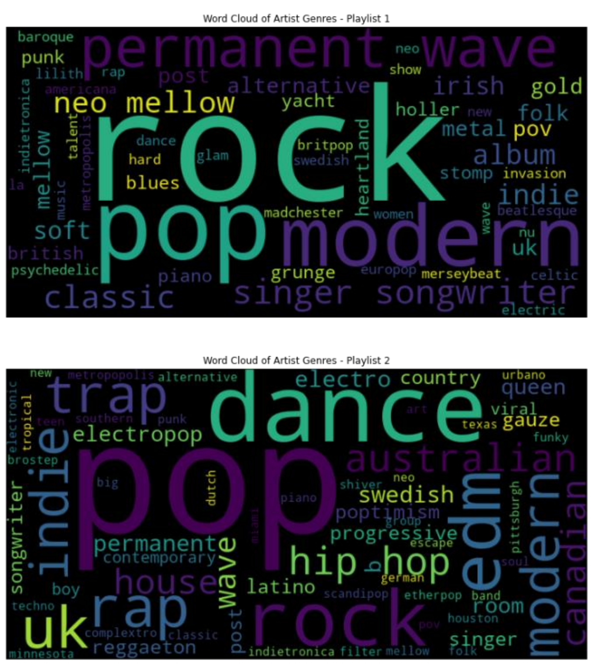

Figure 20: Word clouds for genres

9. This radar chart compares the mean values of selected song features (Speechiness, Energy, Instrumentalness, Acousticness, Liveliness, Valence, Danceability) between Playlist 1 and Playlist 2, offering a visual representation of their overall musical characteristics. The chart displays each playlist as a separate line, showcasing how they differ across the chosen features.

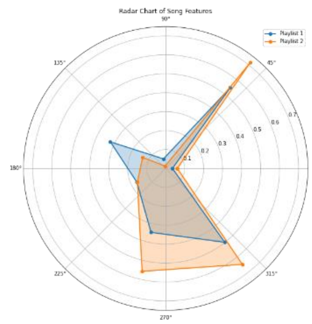

Figure 21: Radar chart for various musical attributes

10. The violin plot compares the distribution of song popularity (Popularity_x) among different artists in Playlist 1 and Playlist 2, revealing the spread and concentration of popularity values for each artist across the two playlists. The plot helps visualize variations in artist popularity between the two playlists. (Experimental)

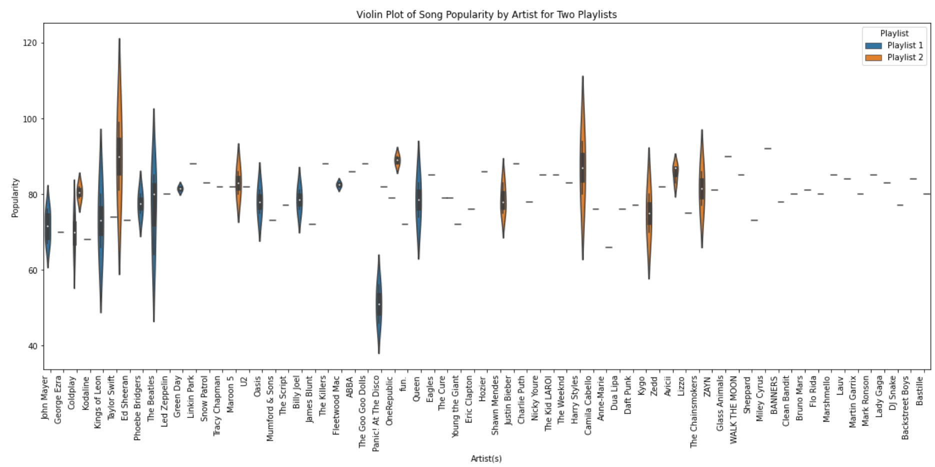

Figure 22: Violin plots to compare popularities

11. The box plot visualizes the distribution of song durations across different genres, with each box representing the interquartile range (IQR) and the median. It helps identify variations in song duration and potential outliers within each genre category. (Experimental)

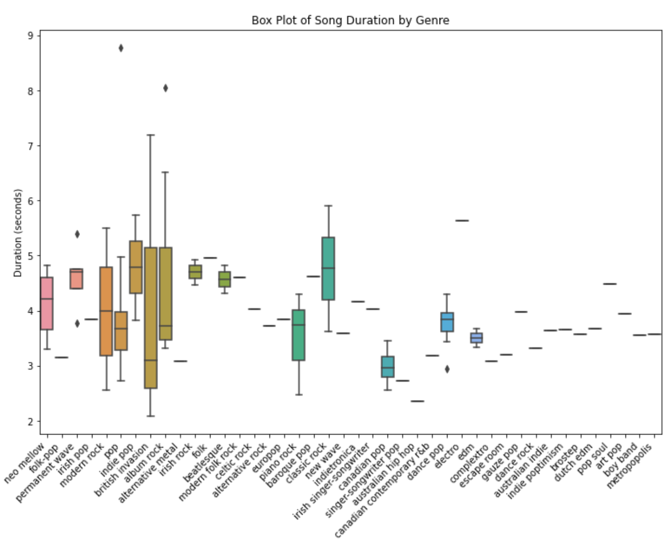

Figure 23: Box plots for song durations across genres

The next step was to experiment with interactive visuals. Some sample Tableau plots were made before proceeding to make the final dashboard in Plotly, Dash and Python. Some of the Tableau visualizations are shown below.

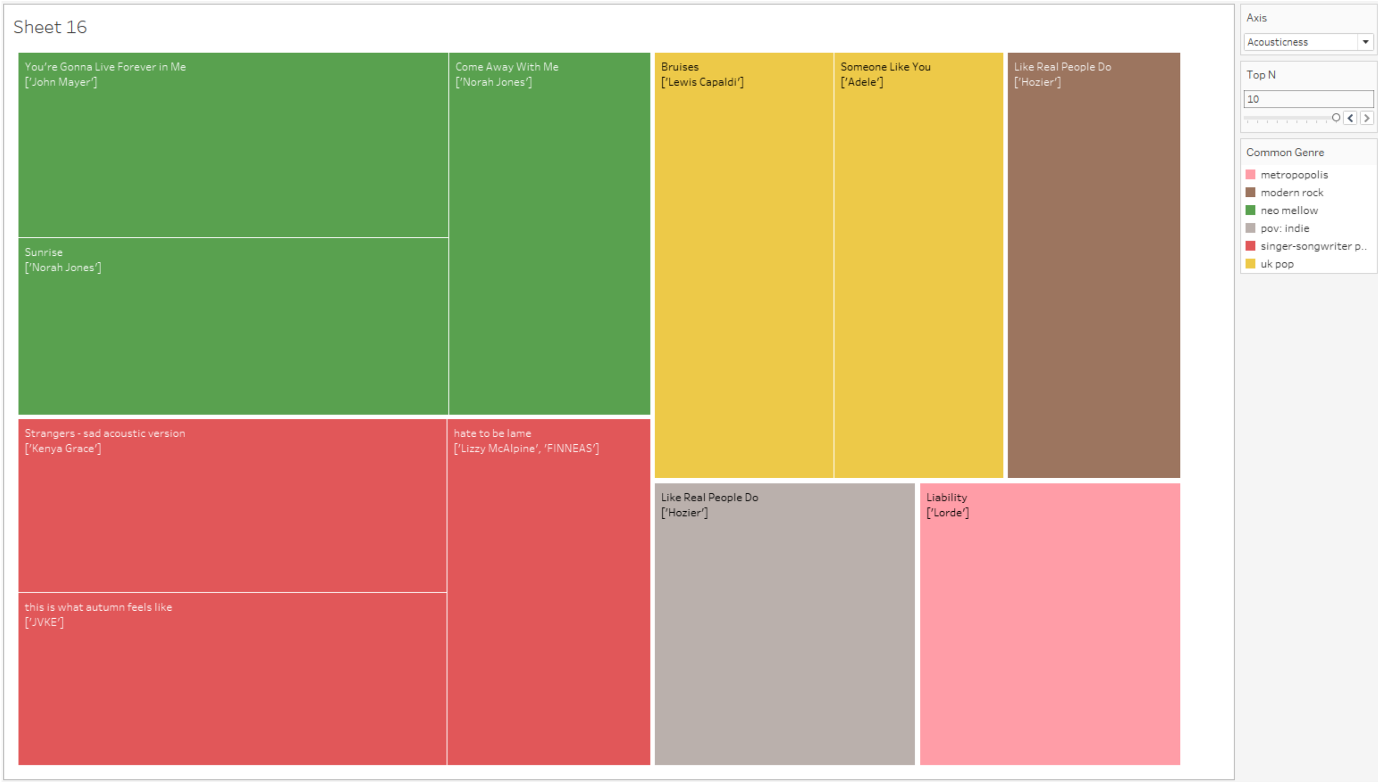

Figure 24: Treemap for top songs by selected measure for common genres

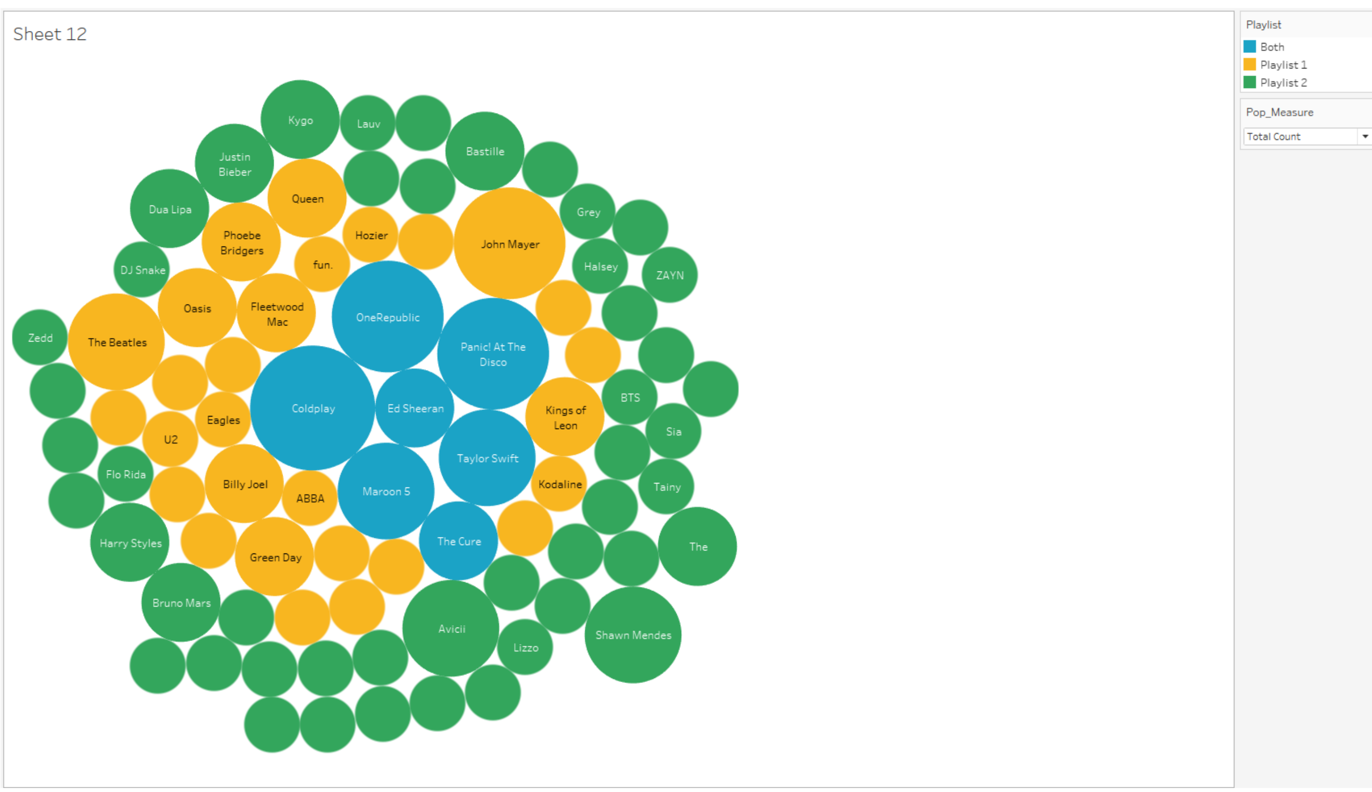

Figure 24: Bubble chart for all artists in the two playlists combined

The final visualizations and the reasons for choosing the visualizations is explained in the results.

4. RESULTS

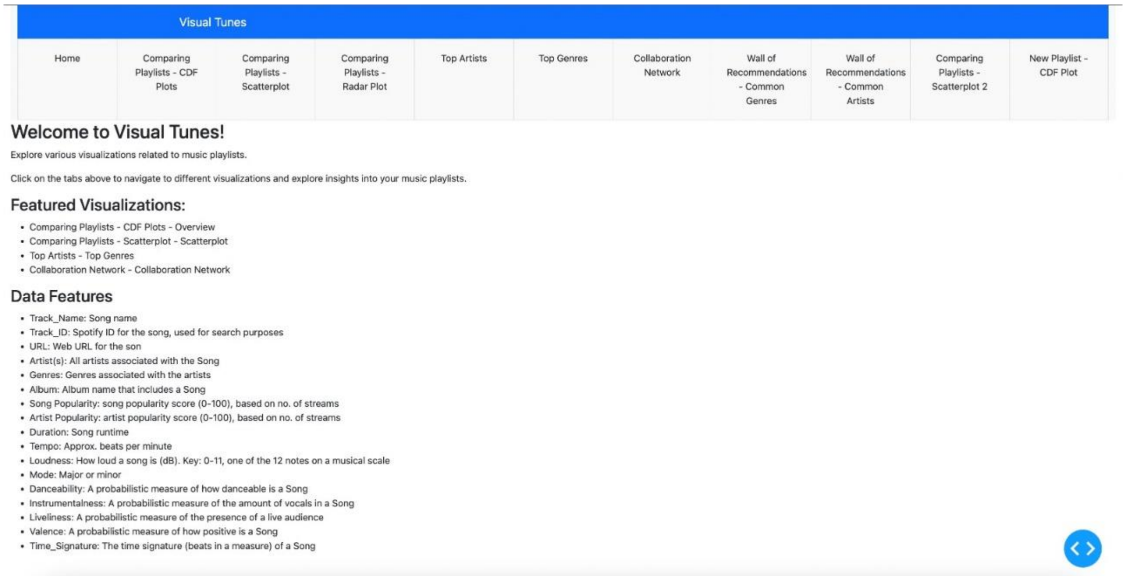

Figure 25: Home Page

1. A cumulative distribution plot for comparing the various numerical continuous variables in the data. The cumulative distribution plot makes a stepwise function that intuitively displays the cumulative distribution of data by visually connecting each data point through a continuous, monotonically increasing line. This connectivity and continuity between elements, along with our tendency toward perceptual closure to fill in gaps and view the curve as a complete, coherent shape, allows ready perceptibility per the Gestalt principles. However, the uniform stepping can introduce visual noise that disrupts figure/ground separation and exact distribution shapes can be obscured. Overall, the cumulative distribution plot strikes a useful balance between visual coherence and distribution resolution.

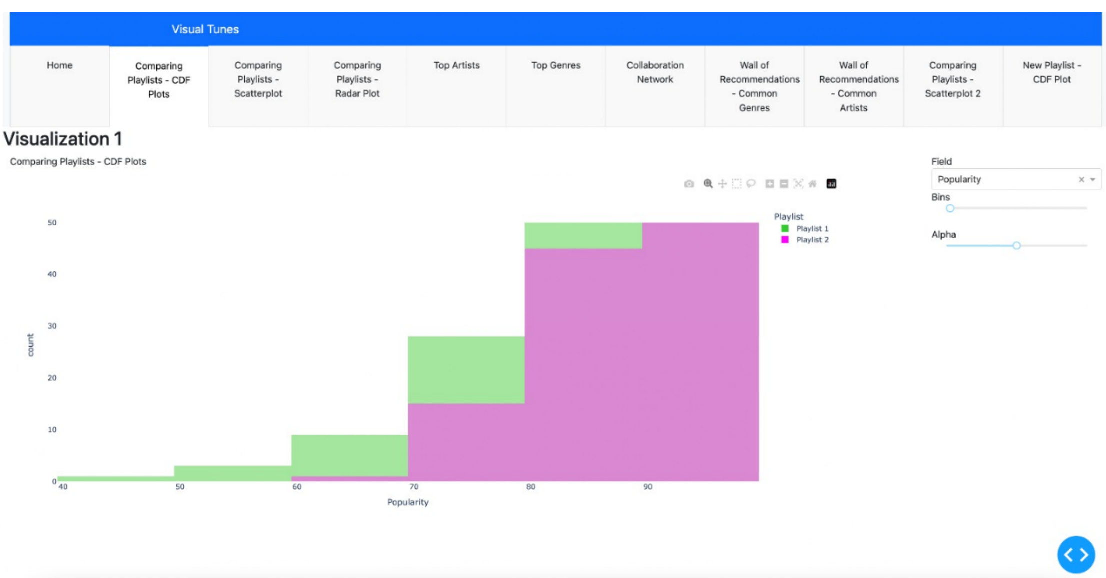

Figure 26: Plot 1

2. A scatterplot comparing the songs from the two playlists, which allows users to control what variables to see on the axes, and what variables control the size of the scatters. Scatterplots with multiple visual encodings leverage Gestalt principles to intuitively convey multidimensional data patterns. Varying element properties like position, size, color, and shape triggers similarity grouping, with proximity also binding related elements. This combines continuity of common fate across encodings with figural emergence from background trends. However, overloading the visual field hampers figure-ground separation, and excessive heterogeneity disrupts grouping continuity. Therein lies a delicate balance to promote perceptual closure without crossing boundaries into disjointed disruption. Methodical manipulation of visual variables against innate biases enables layering of data dimensions upon a foundational

base encoding, unlocking multidimensional insight. But restraint and deliberation are needed to walk the fine line between intuitive unification and cognitive overload.

Figure 27: Plot 2

3. A radar plot for comparing average values of musical attributes of the two playlists. Radar plots leverage similarity and continuity of visual elements to highlight multivariate patterns. Adjacent and enclosed axes promote proximity grouping by factor. Closed polygons take advantage of closure tendencies, with smoothed shapes improving continuity versus ragged or sparse plots. Segment color coding also enhances unified emergence per category. However, complex interleaving and crossing lines can disrupt continuity and obscure trends. Overall, radar charts artfully map multidimensional data to innate biases, but busy plots risk figure/ground confusion. Visual simplicity through deliberate axis choice and smoothing best allows intuitive perceptibility.

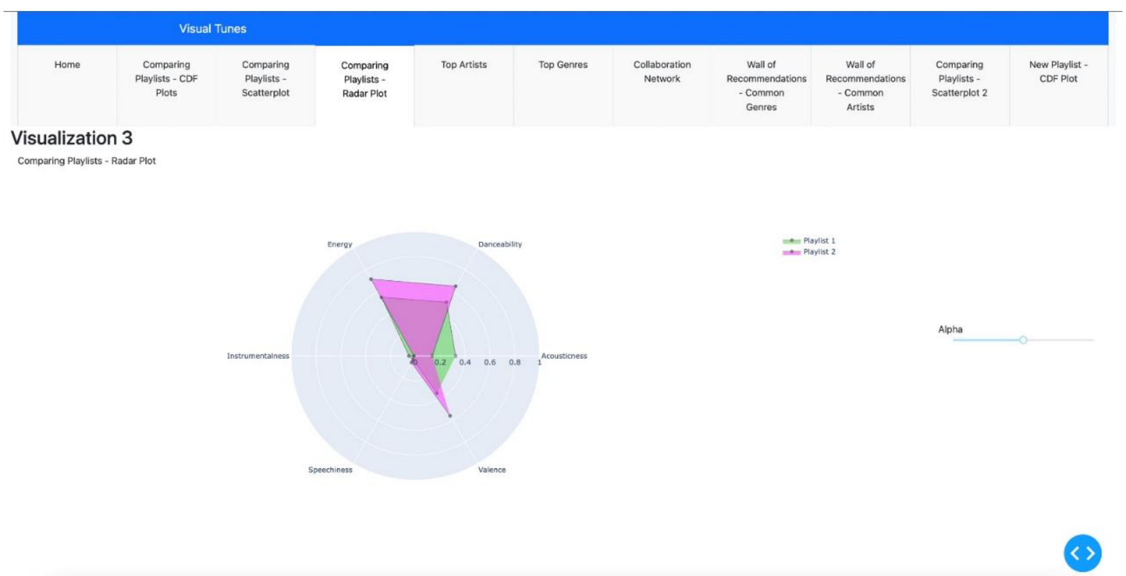

Figure 28: Plot 3

4. A bubble chart that shows the top artists in both playlists combined. There is a slider to change how many artists to view at once. The bubble colors represent which playlist the artist comes from, and the size is the total number of songs that the artist has in the two playlists combined. Top artists can bee seen either by their popularity, or their number of followers on Spotify. Bubble charts use intuitive visual cues to convey multidimensional data. Similar bubble sizes group related elements, with spatial proximity also linking entities. This layered encoding promotes visual emergence of data patterns. But too much variation or overly-dramatic scaling distorts trends. Simplicity through restricted, even sizing and spacing allows continuity of marks to speak for the underlying distributions, without competing clutter disrupting the patterns.

Figure 29: Plot 4

5. A bubble chart for top genres by count in both playlists combined. There is a slider to change how many genres to view at once. The color of the bubbles represents which playlist the genres come from, and the size is the total number of songs of that genre in the two playlists combined. The reason for choosing bubble plots here is the same as the previous plot.

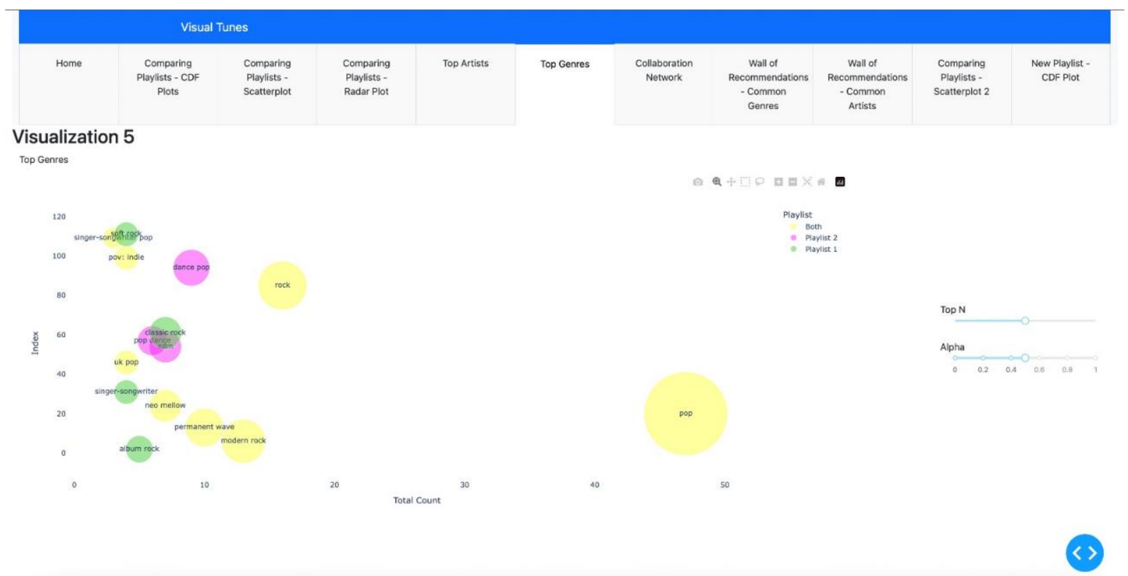

Figure 30: Plot 5

6. A network map to show artist collaborations. Network maps can clearly show relationships and connections between different items. Seeing how artists have collaborated through linking lines applies Gestalt principles like continuity, closure, and connectedness so we perceive integrated clusters and communities. The node are artists and the colors represent which playlist the artist comes from and the node size is controlled by degree centrality. Edges represent that a collaboration exists between the two artists. Nodes with no edges are given a default size of 1. Weighting the edges based on collaboration popularity was being though of, but the results were not as expected, so for now the edges remain unweighted. Nodes with no connections were left on the map with the smallest default size, just to show the presence of the artists in the combination of the two playlists.

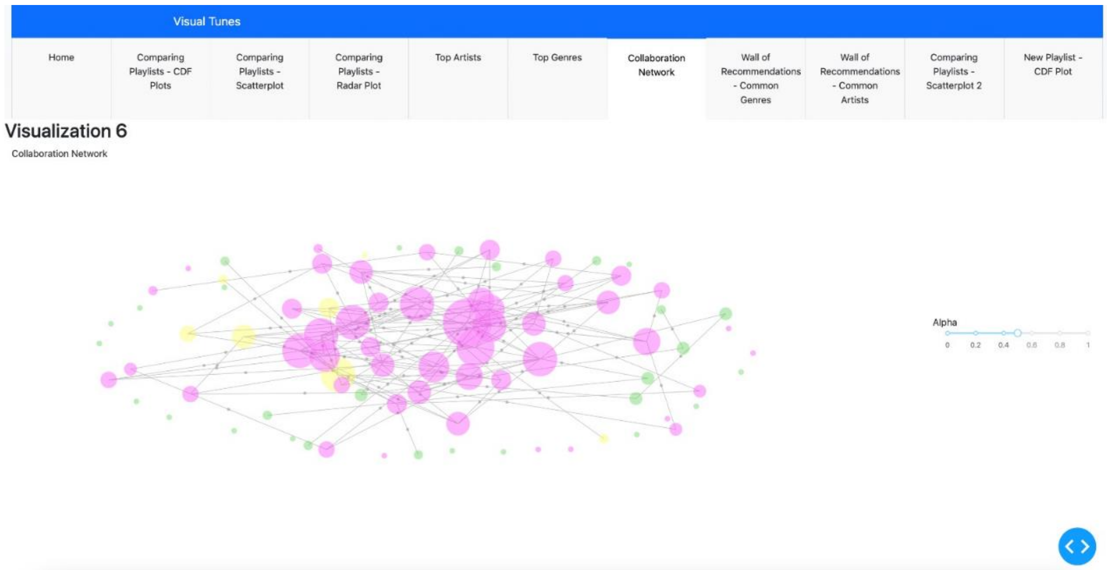

Figure 31: Plot 6

7. A treemap showing top new recommended songs from common genres. There is a slider to change how many songs to view at once. The measure on which the top songs can are chosen be selected by the user. The size and color of the treemap boxes is controlled by the value of the user-selected measure. Treemaps use nested rectangles to visualize hierarchical data. The Gestalt principles of similarity and closure make it easy to perceive groups by color and enclose related subgroups. However, discontinuities from uneven rectangles or imbalanced hierarchy can distort understanding. Well-shaped containment guides the eyes down levels and facilitates pattern comprehension.

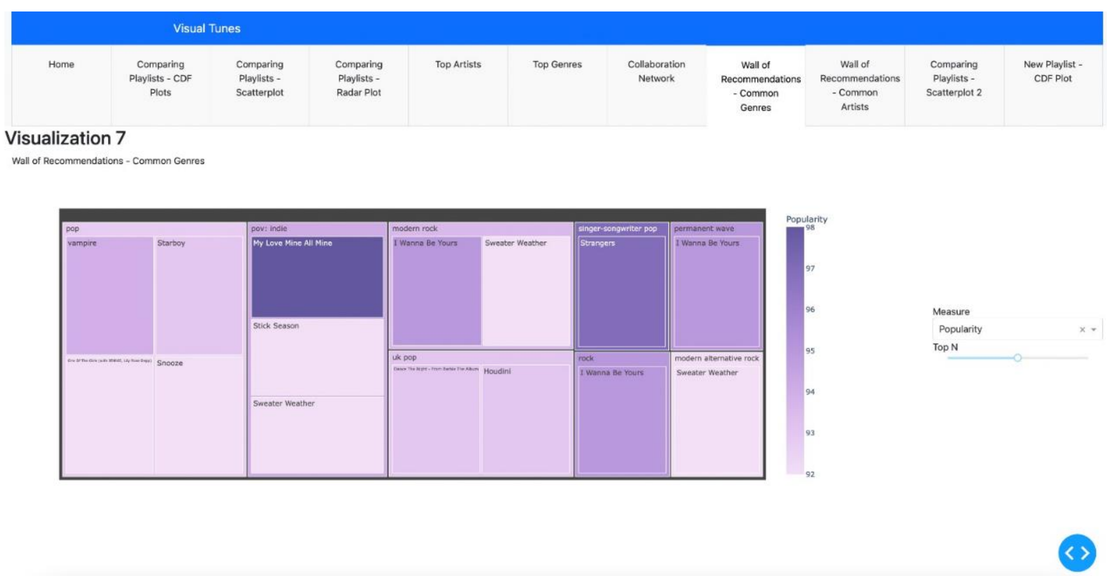

Figure 32: Plot 7

8. A treemap showing top new recommended songs from common artists. There is a slider to change how many songs to view at once. The measure on which the top songs can are chosen be selected by the user. The size and color of the treemap boxes is controlled by the value of the user-selected measure. The reason behind choosing the treemap here is the same as the above plot.

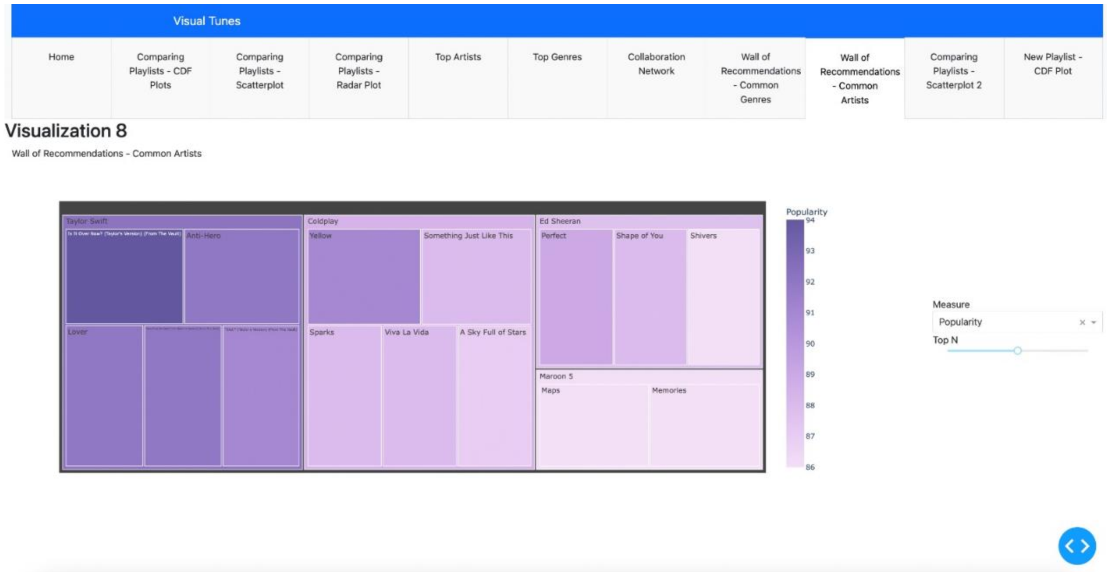

Figure 33: Plot 8

9. A scatterplot visualizing the songs from the new playlist, which allows users to control what variables to see on the axes, and what variables control the size and color of the scatters. Here also, the scatterplot was chosen to visualize multiple encodings efficiently in one plot, and to estimate some trends as well.

Figure 34: Plot 9

10. A cumulative distribution plot to study the multiple musical attributes of the new playlist.

Figure 35: Plot 25

There are sliders added to adjust opacity wherever necessary. Scatterplots have trend lines to discover patterns. Hovering on the plots displays additional information that adds more information to the visualizations. The visualizations in the report were made on playlist data from two different individuals, so this is a demonstration of how personal tastes can be compared, and new recommendations be generated keeping in mind shared interests.

5. CONCLUSION

In the project, we started out to make visualizations for Spotify data that not only allow two playlists to be compared effectively through visualizations, but also see and understand how shared recommendations are being made. We started out by exploring static plots and eventually transitioned to a dynamic dashboard. We even got to experiment with the Tableau platform as well.

The final plots can be studied together to tell a complete story, or can be used individually as required for specific purposes. The plots like the scatterplots can also be used for more than one purpose (eg: find a song based on a certain mood, compare trends etc.).

There is some redundancy in the plots and the data they represent, but that was a conscious choice to let the users decide how they want to view and analyze their data.

The collaboration network and the dynamic scatterplots seemed to be great ways to model musical data to get a lot of insight.

The colors chose for the visualizations were tested using the 'Let's get colorblind' extension and are access-friendly.

Overall, the project fulfills its motivation, and presents an easy to interpret, interactive view of the data. However, there are some limitations to the project as well.

The project works by running multiple scripts separately. Combining the scripts together and allowing the user(s) to simply input the playlist URLs can improve the overall quality of the project. Exploring more complex visualizations, and using more advanced tools to build these visualizations will also be an improvement. Allowing users to provide more than two playlists, and then letting them select which playlists to compare will also be a great addition to the project. Finally, integrating it more closely with Spotify and the Spotify API so that playlists can be fetched directly and collaborations with friends become seamless will make this tool worthy of release to the public.

All the above limitations can be fixed and working on these limitations can be considered as future work scope for the project.

sREFERENCES

[1] “Spotify — About Spotify,” Spotify, Oct. 24, 2023.
https://newsroom.spotify.com/company-info/
[2] Stacy.Goldrick@groupsjr.com, “How Spotify’s Newest Personalized Experience, Blend, Creates a Playlist for You and Your Bestie — Spotify,” Spotify, Aug. 29, 2023.
https://newsroom.spotify.com/2021-08-31/how-spotifys-newest-personalized-experience-blend-creates-a-playlist-for-you-and-your-bestie/
[3] Herlocker, Jonathan L., et al. "Evaluating collaborative filtering recommender systems." ACM Transactions on Information Systems (TOIS) 22.1 (2004): 5-53.
https://grouplens.org/site-content/uploads/evaluating-TOIS-20041.pdf
[4] “Stats for Spotify.”
https://www.statsforspotify.com/
[5] “My Spotify Pie - Top artists - Top genres,” My Spotify Pie - Top Artists - Top Genres.
https://huangdarren1106.github.io/spotify-pie
[6] “Obscurify Music.”
https://www.obscurifymusic.com/home
[7] “Compare your Spotify music taste! - Musictaste.space.”
https://musictaste.space/