Explorify: A Personalized Interactive Visualization Tool for Spotify Listening History

Inna Ivanova
University of British Columbia
innai@cs.ubc.ca

Jonatan Sissener Engstad
Norwegian University of Science and Technology
jonatane@stud.ntnu.no

Abstract

There are many music streaming platforms but Spotify has dominated the market and is currently the largest one with millions of users. Spotify collects huge amounts of data from their users on a daily basis which allows for a great research opportunities in the data science community. There are very limited attempts that have been published on the topic of visualizing Spotify streaming history which inspired this project. We introduce Explorify: a personalized interactive visualization tool for Spotify listening history. Explorify offers a plethora of interactive visualizations that help Spotify users explore their own music taste, patterns and enjoy the interaction with their own data.

1. Introduction

Spotify is the largest music streaming platform with millions of users. Spotify has revolutionized music discovery by collecting streaming data from their users and applying modern neural algorithms for their music recommendations. In addition, Spotify provides information and audio features for each track and artist on the platform which is easily retrievable through their API. Users are even able to request their own personal data including streaming history, playlists, top artists and others. While there are a lot of visualization attempts for Spotify data on the Internet they are relatively simple and do not have interactive components. In this project we propose a novel visualization platform for personalized music streaming history provided by Spotify called Explorify. The main goal of Explorify is to allow users to visualize in an interactive manner their personal streaming history and discover patterns for their music taste.

2. Related Work

To this date only a limited number of attempts has been published on the topic of visualizing the listening history of Spotify users. More work has been published on the topic of visualizing the listening histories of users of the Last.fm platform, a service with very similar offerings to that of Spotify. There are differences in what data the two platforms make available, but they are mostly minor, and the semantics remain largely the same. In discussing these works, we will usually make no distinction between what platform the work relies on, as it (in most cases) is immaterial to the work.

The published works within this area can be divided into two groups, based on their main goal. The first group's goal is to help the user discover new music. The papers targeting this goal seek to create tools that reveal tracks, artists and genres that were previously unknown but are believed to be enjoyable to the user. Usually, this is achieved by allowing the user to explore and traverse networks of similar artists in search of undiscovered ones [1, 2]. Several tools that use this approach are available on the Internet. To name a few: LivePlasma, MusicRoamer, Music-graph, Music-map.

The goal of the second group is to reveal trends and facts about the user's listening history, supporting the user in introspection about their own listening habits. These tools all visualize the user's listening history in one or more ways, and support various means of interaction such as adjusting the granularity of what is shown [3, 4, 5]. Another tool, LastHistory, attempts to make this presentation more meaningful to the user by augmenting the listening history with contextual information from other data sources, such as personal phots and calendar entries [6].

In a recently published work, Wirfs-Brock et al. explore how these goals can be combined, enhancing music exploration by giving the user deeper insight into their own listening history and listening patterns [7].

While Explorify falls into the second group, we hope that our solution will facilitate discovery by piquing the user's interest and deepening their insight into their own music preferences in a way that will inspire exploration.

3. Dataset

There are a total of three datasets tested for the visualizations provided by Explorify. Each dataset comprises the streaming history for a user over one year period. Each item of the datasets contains the `trackName`, `artistName`, `msPlayed` and `endTime`. An overview of the datasets is provided in Table 1. The raw datasets are further processed to extract detailed information for each track and the corresponding artists and album. Data processing libraries such as `numpy` and `pandas` are used for the computations as well as the Spotify API to retrieve more details for each track, artist, and album entity. Examples of additional information provided by the Spotify API include:

- Track audio features such as danceability, speechiness, etc.;
- Genres associated with the album and the artist;
- Popularity of the album and the artist;
- Number of followers the artist has;
- Similar artists for each artist;

	Total Tracks		Unique Tracks	Unique Artists
	Skipped	Not Skipped		
SH_J	1126	6887	4272	1598
SH_D	1919	3570	1428	375
SH_K	1862	1711	1451	818

Table 1. Datasets overview. Skipped tracks are ones that are played for less than 10 seconds.

Dataset availability is both static and dynamic. The static part is the streaming history provided by the user. The dynamic part is the prepossessing of the raw user data and the additional features fetched through the Spotify API.

4. Data and Task Abstractions

There are a total of 5 tasks we will be visualizing within the Explorify platform: **Artist-Genre Network**, **Track Clustering by Audio Features**, **Artists Stream-graph**, **Top Artist Over Time** and **Daily Listening Pattern**.

4.1. Artist-Genre Network

The first task aims to explore the the relationship between genres and artists. Questions this particular task aims to answer are:

- How do genres connect artists?
- What other genres are associated with connected artists and how different are they?

Both **genres** and **artists** are categorical attributes. These attributes form a dataset of type network since there is a genre relationship (link) between the artist items (nodes). Each artist can be associated with more than one genre. The number of artists for each dataset is relatively large (hundreds to thousands) so in order to have a meaningful visualization with distinct connections only the top 100 artists will be considered. In terms of abstract definition, this task explores the connectivity of nodes within a network and the differences between connected nodes. For a Spotify user, the goal of this task is to analyze the network. In addition, the user might want to look-up specific artist and their connections. The hypothesis is that artists connected by genres will not be associated with other completely different genres. To verify this hypothesis, the user needs to consume the network graph and discover patterns and abnormalities.

4.2. Track Clustering by Audio Features

The second task will be clustering user tracks by their audio features. Questions this particular task aims to answer are:

- Which tracks are similar according to their audio features?
- What is the predominant genre within a cluster and is there a distinct winner?

There are a total of nine audio features attributes for each track: danceability, valence, energy, tempo, loudness, speechiness, instrumentalness, liveness and acounsticness. All these features are ordered quantitative sequential data. Each track is further assigned a list of genres. The genre attribute is a categorical data. All these attributes form a dataset of type table since there are items (tracks) with associated attributes (audio features and genres), and numerical or categorical values assigned to each cell of the table (the audio feature value and genre). In terms of abstract definition, this task partitions high dimensional data points into clusters and maps them into 2D projection. In addition, the task assigns label to each cluster. For a Spotify user, the goal of the task is to explore the tracks they listen to. The hypothesis is that tracks with similar audio features will be mapped close together in the low dimensional space. In addition, tracks mapped together should have similar genre associated with them. To verify this, the user needs to look-up specific tracks within a cluster and check their genre labels.

4.3. Artist Streams Time-Series

In the third task, we wish to let the user explore the evolution of their listening patterns for the duration of the history. The goal is to allow the user to discover events such as when they first started listening to an artist, and patterns such as how the user's relationship with an artist changed over time. We also wish to make it easy for the user to see which artists they streamed the most, as well as the absolute number of

streams, at any point in time. The data used for this task will be the following:

- artist name: categorical
- artist id: categorical
- timestamp: quantitative, sequential, hierarchical

Each data set covers around one year of a user's listening history. It follows that the range of the timestamp data is within a single year. Becasue the timestamps are accurate down to the second, some aggregation will have to be performed so that items can be grouped together by temporal closeness. Grouping the items on a set number of days, like 5 or 7 will yield a good number of groups, being granular enough to reveal real detail without being so fine-grained that overarching patterns are hard to discern. There are no levels in the categorical data. Artist id should map 1-1 to artist name, though artist id will be used internally in the system to be safe.

For this task the data will be structured in the following way: For each set time-interval, every artist in the data set will be associated with the number of times they occur within that interval. One can then visualize, for each interval, the absolute or relative number of streams each artist received.

4.4. Top Artist of All Time

The goal if the fourth task is to provide the user with a clear idea of the most streamed artists in their history. This may at first glance seem to overlap with the third task, but while these two tasks do represent the same data (this task is practically a summary of the third task), the information the user will be able to easily extract from the respective visualizations will be quite different. The third task displays the most streamed artist for each time interval. For most users, the artist that is streamed the most will shift between each interval, and gaining a clear picture of (for example) the top 5 most streamed will be hard to do visually. We believe that the user will be interested in this data and therefore we have chosen to include the fourth task.

The data used for this task is the exact same as the one used in the third task. The data processing will be similar as well. The difference in processing is that the aggregation will only be performed over a single time interval, which will include the whole history.

4.5. Daily Listening Pattern

The final fifth task is visualizing the daily Spotify usage of a user. Questions this particular task aims to answer are:

- How music streaming routine changes over time?
- Which days observe the most listening hours?
- Is there spike usage for specific dates?

To retrieve the data needed for this task, the tracks streamed on a particular day are summed up together. This is a time series dataset so it's easy to aggregate tracks by day. This results in an attribute `tracks_count` that is ordered quantitative sequential data. This attribute forms a dataset of type table since there are items (days) with associated attributes (tracks_count), and numerical values assigned to each cell of the table (number of tracks per day). In terms of abstract definition, this task reduces a table dataset by aggregating attributes and displays the results. For a Spotify user, the goal of the task is to analyze their streaming listening history and search for patterns or spike usage for specific dates.

5. Solution

The work on the Explorify platform is performed using Python and JavaScript with D3 [8]. Python is mainly used for data processing while JavaScript with D3 is used for creating the visualizations.

5.1. Artist-Genre Network: Force-Directed Graph or Adjacency Matrix

Since the data to be visualized for task one (Section 4.1) is a network type a total of two visualizations are considered: adjacency matrix presentation and force-directed graph inspired by a paper on a similar visualization task [9]. The preference is over the force-directed graph, however, depending on the time constrains and challenges around the implementation, we might end up with the simpler adjacency matrix. Whichever visualization we land on, we will aim to be interactive.

In the force-directed graph each artist will be encoded by a node and each genre connection by an edge. Since artists could be assigned more than one genre some edges might have higher 'weight'. Therefore, the edge width will encode the number of genres in common – the more the wider the line. The colour channel can be used for highlighting some connections of interests. To inspect specific artists the user will be allowed to click on the artist (node) of interest and compare to another artist that is a connection. A pop-up display will show detailed information about the similarity and difference in genres between the two artists.

In the case of adjacency matrix each row and column item will correspond to the unique artist. The values of the cells in the matrix will display the number of genres in common between two artists. To encode the number of equal genres the colour and hue channel will be used. We will work with a single colour sequential colour map because the number of genres in common is always a non-negative number. To inspect a connection of interest, the user will be allowed to click on specific cell. As with the force-directed graph, a pop-up display will show the details.

5.2. Track Clustering by Audio Features: GrouseFlocks

For the track clustering task we will use a GrouseFlocks graph [10] visualization which was inspired by a post on the Internet [11]. First step is to partition the tracks into cluster according to their audio features. To do that we will use either k-means or quadtree clustering algorithms. Once the clusters are obtained a dimensionality reduction algorithm will be applied on the tracks audio feature vectors data to reduce the space. We will experiment with the most famous ones like tSNE, PCA or UMAP.

All these steps prepared the data we need for the actual construction of the GrouseFlocks graph. Each leaf node on the graph is a track. Tracks within a cluster are connected to a node which represents the genre distributions within that cluster. The colour channel will encode the unique genres which are categorical data. The connecting node will represent a pie chart of the genres distribution. The border outline of the node will be filled with the colour representing the predominant genre for the cluster. The user will be able to click on the nodes that connect the tracks within a cluster and inspect further which tracks are part of that cluster and what genres and audio features are there.

5.3. Artist Streams Time-Series: Streamgraph

For the task of visualizing the artist streams time-series, we imagine using the streamgraph idiom [10]. The overarching concepts of a streamgraph are relatively simple, but there are many details such as color scheme, ordering, labelling, scale and angle that must be taken into consideration [12].

5.4. Top Artists of All Time: List or Animated Bar Chart

For this task, we have imagined two different visualizations, depending on how much time we to work on the task, and how challenging the implementation of each solution proves to be. The simplest option is simply a list of the top 3 most streamed artists, but with the visualization arranged to look like a podium. A picture representing each artist (available via Spotify) will be used to represent each artist as on the podium along with the artist's name. The total number of streams for each artist will be displayed as well, above each artist.

The second option we are considering is an animated bar chart, sometimes referred to as "racing bar charts". This bar chart will display the top 5 or 10 most streamed artists cumulatively, moving from the start of the history to its end. The bar chart will contain the 5 artists with the most number of streams so far, and be sorted by the total number of streams in ascending order, up the y-axis. As artists overtake one another, they will trade places in the chart, moving up or down the ranking, or even falling off it completely.

A potential challenge with this second option is that some users may have very many streams of a small set of artists at the very beginning of their history, before later diversifying their taste in artists. In this case, it is possible that the animation will remain static for much of its playtime, as no new artists will have the number of streams necessary to break the top 5. A solution to this problem would be to introduce a popularity score that not only takes into account cumulative streams, but also the recency of streams, so that artists who have not been listened to for some time will have their score decay and thus fall off ranking. This will of course mean that the ranking is no longer based on a total number of streams, but an arbitrary formula made up by us, which users may not find to be meaningful. Nonetheless, it is a path we may chose to explore.

5.5. Daily Listening Pattern: Heatmap Matrix

For this task a heatmap matrix was chosen to display the listening patterns over time. The visualization is relatively simple and was inspired by the famous visualization for the number of commits per day by Github. There are 7 rows in total and each of them corresponds to the day of the week. Each column of the matrix corresponds to the week number but this will not be displayed. Instead, the year and the month will be displayed along the x axis as they are more descriptive. Each cell value corresponds to the number of tracks listened on the particular day. We use the colour and hue channels to encode the number of tracks listened on a particular day. The higher the hue, the higher the number of tracks played on that day. Since the values of each cell are quantitative sequential data with positive range, a single colour sequential color map will be used.

By aligning the streaming interaction in this way we can clearly observe whether there are any patterns over a month, week or specific days. Furthermore, we can clearly see if there are any spikes in the interactions. Additional marks can be used to highlight information, e.g. star mark to point specific days of interest. Since the data to be visualized is over the course of a year (a total of 52 columns), a slider can be added for the user to inspect the data in an interactive manner and navigate easier through the visualization.

6. Execution Timeline Milestones

The following timetable suggests a rough estimate with milestones that aims to achieve the proposed objectives in a total of 160 hours.

1. Familiarize ourselves with the dataset - 5 hours each (10 hours total)
2. Familiarize ourselves with D3 - 7 hours each (14 hours total)

3. **Milestone 1**: Create Daily Listening Pattern Heatmap - Inna 5 hours
4. **Milestone 1**: Create Artists Stream-graph - Jonatan 7 hours
5. **Milestone 2**: Add interactivity for Daily Listening Pattern Heatmap - Inna 5 hours
6. **Milestone 2**: Add interactivity for Artists Stream-graph - Jonatan 5 hours
7. **Milestone 3**: Create Artist-Genre Network Graph - Inna Jonatan 8 hours each (16h total)
8. **Milestone 4**: Add interactivity for Artist-Genre Network Graph - Inna Jonatan 5 hours each (10h total)
9. **Milestone 5**: Create Top Artist Over Time Barchart - Jonatan 7 hours
10. **Milestone 5**: Create Track Clustering by Audio Feature - Inna 6 hours
11. **Milestone 6**: Add interactivity for Top Artist Over Time Barchart - Jonatan 7 hours
12. **Milestone 6**: Add interactivity for Track Clustering by Audio Feature - Inna 6 hours
13. **Milestone 7**: Combine all visualizations together - Inna Jonatan 10 hours each (20 total)
14. Final report - Inna Jonatan (10 hours)
15. Presentation and demo - 5 hours

Total of 133 hours. Left 27 hours to spare if something goes wrong.

7. Results

A rough sketch of the Explorify platform shown in Figure 7.

8. Discussion

8.1. Limitations and Future Work

The Explorify platform could be extended to music streaming histories from other platforms such as YouTube or Last.fm. In addition, the Explorify platform could adopt datasets from two or more users and display their similarities and differences. Several people we have talked with have expressed interest in such a feature. In his work on Last.fm explorer, M. Pretzlav shows how this could be done for the data of two users [3]

Our results can be no more accurate or detailed than the data we receive from Spotify. For example, genres only being registered to artists and not tracks hurts how granular we can be in any analysis based on genre. In future work, it could be interesting to use external data source such as MusicBrainz or LastFM to find more granular genre data and see if more interesting results could be found.

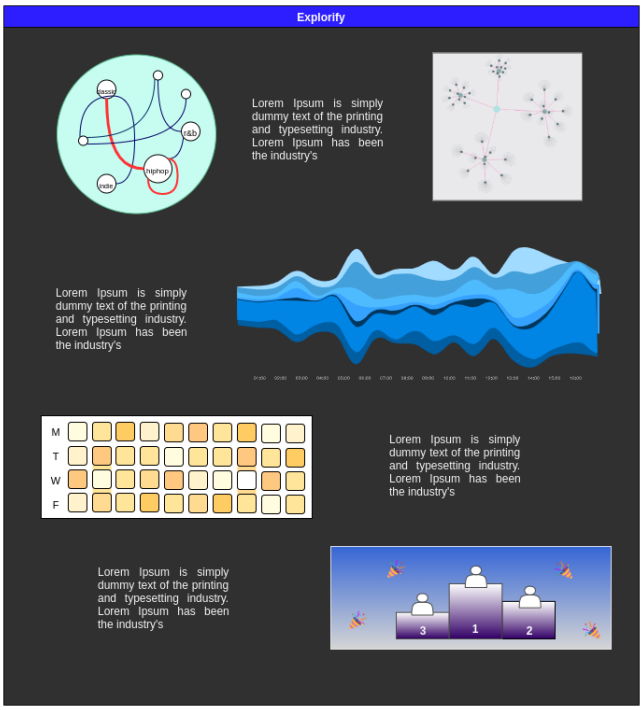

Figure 1. Explorify platform. There are five visualizations for the five tasks we define in Section 4.

References

[1] M. D. M. C. José Bateira, Fabian Gouyon, "Music discovery in spotify with rama," 2014. 1

[2] T. Dang, A. Anand, and L. Wilkinson, "Fmfinder: Search and filter your favorite songs," vol. 7431, pp. 348–358, 07 2012. 1

[3] M. A. Pretzlav, "Last.fm explorer: An interactive visualization of hierarchical time-series data.pdf," 2008. 1, 5

[4] R. Dias, M. J. Fonseca, and D. Gonçalves, "Interactive exploration of music listening histories," AVI '12, (New York, NY, USA), p. 415–422, Association for Computing Machinery, 2012. 1

[5] D. Baur and A. Butz, "Pulling strings from a tangle: Visualizing a personal music listening history," IUI '09, (New York, NY, USA), p. 439–444, Association for Computing Machinery, 2009. 1

[6] D. Baur, F. Seiffert, M. Sedlmair, and S. Boring, "The streams of our lives: Visualizing listening histories in context," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1119–1128, 2010. 1

[7] J. Wirfs-Brock, S. Mennicken, and J. Thom-Santelli, "Giving voice to silent data: Designing with personal music listening history," *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, 2020. 1

[8] M. Bostock, "D3.js - data-driven documents," 2012. 3

[9] N. J. Bryan and G. Wang, "Musical influence network analysis and rank of sample-based music.," in *ISMIR*, pp. 329–334, 2011. 3

[10] T. Munzner, *Visualizaiton Analysis and Design*. CRC Press, Taylor Francis Group, 2014. 4

[11] "Spotify music clustering." https://observablehq.com/@sandravizmad/force-directed-layout. Accessed: 2021-10-20. 4

[12] L. Byron and M. Wattenberg, "Stacked graphs – geometry amp; aesthetics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1245–1252, 2008. 4