Spotify Playlist Analytics

Anonymous Author(s)

ABSTRACT

This paper discusses the process and findings of building a Spotify Recommendation Application. The initially intended topic of this paper was to build a grocery product-related recommendation application, but due to the lack of existing data in the area, the topic idea was modified. This application used the Spotify Playlist Database which contained a more sufficient amount of information. With this new area of focus in mind, the application and research progressed. The first step in creating the physical application was to clean and prep the data. The Spotify Playlist Database contained an abundance of metadata that was not needed, therefore removed in this step. Once cleaned and formatted into the desired table format, the next step that began was the Analysis step. In this step, a thorough understanding of the data obtained was developed. This allowed for a more seamless process of building the rest of the application. Apart from building the song suggestion and popularity prediction functionality, this formal report was created. In order to create said report, research was done to discover how recommendation systems, in general, were generated as well as the importance of music in society.

KEYWORDS

Spotify, Music Recommendation System, Pitch, Timbre, Tracks

1 MOTIVATION

In this report, we will address the topic of music recommendations. Using Spotify's database on users' playlists, we intend to build a recommendation system. When given a playlist, we will be able to predict what song the user may add next. Along with building this application, we will analyze what we learned from the information within the database using different Data Mining Techniques.

Music recommendation systems can be of various types. Some systems like the one proposed by Bai Et. al. [1] use the user's parameters like heart rate, physical condition, etc. to recommend background music the user would like. Other systems typically take into account what a user is listening to/ already has listened to, to predict the type of songs that a user could like. The music system we propose is slightly different, instead of predicting a song based on the user's liking, we analyze a playlist of songs and determine which songs are likely to be added to the playlist based on similarities in songs.

The main function of a music recommendation system is to recommend a song that the user might listen to. To do this, we take the user-created playlists into account. We try to find similarities in a playlist, similar playlist to the one created, the popularity of songs, etc. to suitably give a song that would go with a particular playlist.

To design our application, we looked at the datasets. We tried to determine what group of users would find helpful information and how to present this information to the distinct groups. We are approaching the design of our application from three major lenses which we have named *Class A* and *Class B* users.

- A *Class A user*: Class A user is loosely defined as a regular Spotify user. This person has a profile and listens to music regularly. Our focus for this class of users is creating custom song recommendations based on things they have in their playlist and things users 'similar' to them have saved.
- *A Class B*: Class B user is defined as an infrequent user who uses Spotify infrequently. For this class of users, our goal will be to present information based on the track they currently have in their playlists.

We have currently identified one primary dataset that we plan to utilize. AICrowd [2] is a dataset created by Spotify for a music recommendation challenge that they created. It contains 1,000,000 playlist data and 2,000,000 track data that were created by Spotify users between January 2010 and October 2017. The data has attributes as follows:

- name: Name of the playlist (Nominal).
- collaborative: Boolean whether the playlist is collaborative with other Spotify Users (Nominal).
- pid: The unique identifier for this playlist in the database (Nominal)
- modified_at: Time of when the playlist was lasted updated (Interval)
- num_albums: Number of albums the songs in the playlist are from (Nominal)
- num_tracks: Number of songs in the playlist (Nominal)
- num_followers: The number of users that follow this playlist (Nominal)
- num_edits: The number of times this playlist was modified (Nominal)
- duration_ms: Length of the playlist (Interval)
- num_artists: The number of artists contributing to the playlist (Interval)
- tracks: List of tracks, which include metadata about track, album and artist
  - pos : Position of track in the playlist (Ordinal)
  - artist_name : Name of the artist (Nominal)
  - track_uri : Link to track page on Spotify and a unique identifier for track (Nominal)
  - artist_uri : Link to artist page on Spotify and a unique identifier for artist (Nominal)
  - track_name : Name of the track (Nominal)
  - album_uri : Link to album page on Spotify and a unique identifier for album (Nominal)
  - duration_ms : Length of song (Interval)
  - album_name : Name of the album (Nominal)

The main goal of our project is to design and implement a recommendation system using the Spotify Playlist Dataset along with a prediction on how popular a playlist may be. We will be using users' playlist tracks list to tailor song recommendations that may best fit their playlists. This recommendation and prediction system will focus on:

- *Suggesting new songs for existing playlists*: It will take in a file listing songs and compare them to other playlists in order to recommend a new song.
- *Providing an estimate on the possible popularity of a playlist*: It will take in a user's playlist and estimate how popular it may be based on the popularity of similar playlists.
- *Giving users the option to generate a playlist of popular songs*: It will give a user a playlist of songs that have the most followers (which is generated from the metadata from the playlist dataset).
- *Giving users the option to create a playlist of frequently added songs* It will give users songs that appear in the most amount of playlists.

When building an application, there are many moving parts. To help coordinate this, we developed an outline to guide our application flow.

(1) Dataset Discovery
(2) Goals and Subgoals
(3) Data Cleaning and Preparation
(4) Data Analysis
(5) Song Suggestions
(6) Popularity Prediction
(7) Playlists Generator

The first step in our outline was Dataset Discovery. When initially beginning this project, we intended to create a grocery store recommendation system. As we moved forward with that intent, we discovered that the amount of preexisting data was limited and would not be sufficient for our application. Therefore we changed the focus of our recommendation system from grocery products to music. The Spotify Dataset has a larger amount of data entries that will allow our application system to produce a sufficient analysis and versatile recommendation.

The second step of our outline is generating our goals and subgoals, which will be outlined in this section and referred to in Section 8, Status and Future Work. The overall goal of our application and project is to generate a recommendation system while learning how to adequately utilize data. The main functionality we would want to complete is taking in a list of songs (tracks) and generating a recommended song that a user may like. Our second functionality goal is to generate a prediction of how popular a user's playlist may be. These two main functionality goals allow us to meet our learning goals.

Our first learning goal is to understand and undergo the steps of cleaning and preparing a dataset. While we did this on a much smaller scale within GA1 and GA2, those datasets were already semi-clean and prepped. The Spotify dataset, while clean, is not prepped. It contains playlist metadata that we have no use for. Section 3 elaborates on the steps we undergo to prepare the data for our application's needs. Our second main learning goal is to utilize the data mining technique of prediction. Using a select few of the provided metadata for each playlist, we intend to estimate how popular any given playlist may be. This functionality is only possible due to analyzing the data enough to generate predictions.

Due to the nature of the academic workload in a semester, our time is limited. With this reality, we prioritized our desired outcomes into the main goals discussed above and subgoals we'd like to achieve given the time. Our first subgoal is creating a functionality where our application generates a new playlist based on the popularity of the song. The user can enter the threshold percentage of how popular the song is, and we would return a list of those songs. For example, if a user enters *generate playlist popular 10.0 30*, we would output a playlist that has at most 30 songs where each song has an accumulated followers amount in the top 10% of all songs in the database. Our second subgoal is similar to our first, but instead of followers, we would calculate it based on how many playlists the song is in. Our last subgoal is focused on the Data Mining Technique of Data Visualization. We would like to give the user options on how the estimated popularity of their playlist is presented.

The third step in our outline is Data Cleaning and Preparation. In this step, we are gathering information from our dataset into Python tables and filter out the information we do not need. Following this, we analyze our data in step four. This analyzing step allows us to know the extent our application can actually perform, as well as help us to understand the performance of each song truly. Steps 5 through 7 are where we build the functionality of our application which will be discussed in more detail in Section 3.

This report is split into 8 remaining sections. First, we will explore work done by others in Section 2. Followed by discussing how we structured our application and proceeded to implement it in Section 3. Section 4 explores the information we gathered from our data and analysis. Before addressing the issues we ran into and how we approached it in Section 7, we discuss the possible legal and ethical considerations that would have to be taken into account with our application in Sections 5 and 6. As we approach the end of our report, we address our current standing in the project as well as what we would like to see going forward in Section 8. Finally, we conclude our report with Section 9, where we summarize the intent of our application and data, then restate our results.

2 RELATED WORK

Recommendation systems have been a topic of research for a long time, and music recommendation has been tried using a variety of ways. Bai Et. al. [1] propose a system that uses user's physical parameters to predict music. Lopes Et. al. [4] propose a method which first extracts a user's social media text to determine a user's mood and then predicts a song according to determined mood. A convolutional neural network is used to perform both the lexical analysis to determine sentiment, and the prediction thereafter. Girsang Et. al. [3] proposed a traditional approach that uses collaborative filtering to predict the ratings of different songs a user would like on a scale of 0 to 100. Thus music recommendation systems can be broadly classified into two types:-

- Filtering based recommendation systems
- User data based recommendation systems

3 DESIGN AND IMPLEMENTATION

The first step in the design of this application is building the data tables. The playlist table will have these components:

- id (integer)
- Name (text)
- Description (text)
- numFollowers (integer)
- modifiedAt (date)
- collaborative (boolean)
- numTracks (integer)
- tracks (list)

The id will act as a unique key that will help us identify the playlist and navigate to it in the table fairly quickly. The name and description will help in terms of displaying the playlist to users. The number of followers directly aids in the prediction of the estimated popularity of given playlists as well as in generating a playlist based on the popularity of a song. The modifiedAt and collaborative fields aid in the analysis stage. It helps us to understand the patterns in playlist creation habits over time. And the tracks field is a list of songs that belong to that playlist.

Another table we will have will be the songs table:

- id (text)
- trackName (text)
- albumName (text)
- followers (integer)
- appearances (integer)

The id will act as a unique key that will help us identify the song and navigate to it in the table fairly quickly. The trackName and ablumName will help with formatting the generating playlist output for users, while also helping to match inputted songs with existing songs in the playlist. Followers and appearances are fields that we will need to calculate and add to the songs table.

In order to calculate the number of followers and appearances, in a time-complex efficient way, we will need to build the playlist table and song table simultaneously. We can use the track_uri from the database as the id for each song. This will help us navigate the song table using the information we find in the playlist table. As we build each playlist entry, we can go through the list of songs. If the song does not exist in the song table, then add it to the table and set appearances to one and amount of followers to the playlist's amount of followers. If the song already exists, then increment appearances by one and the number of followers by the playlist's amount of followers.

Once these tables are built and populated, we can move on to the analysis. When analyzing the data, we want to keep an eye out for patterns or anything that stands out. Looking at the number of modified playlists over time may allow us to see if playlists were a popular trend at certain times as opposed to others. On top of that analysis, we can see how many were collaborative over that stretch of time as well. Do people like to keep the creation of their playlist to themselves or do they prefer to work with others?

After understanding the trends of the data we are using, we can begin to build the functionality of the application and use the formatted tables. The first feature is the suggestion of a song based on the other songs in the playlist. One method of doing this is going through each song and adding any songs that have been on the same playlist to a dictionary. The key would be the song and the value would be the count of how many times the songs have been on the same playlists. Then we return the key with the highest value as our recommendation.

For our second feature, producing an estimation of how popular a playlist may be, we would need to utilize our followers field in our songs database. One step will be to get the sum of the followers of each song in the playlist and take the average. We would then need to compare this average with the average of each playlist in our playlist database. Doing this will allow us to see where the inputted playlist ranks amongst the others.

The two generating playlists features are very similar to one another. We would simply have to iterate through the songs table and add the songs to the playlist as we go through. We would start with an empty list with a size capacity as the user input max size. As we go through the list, we want to make sure the list stays sorted. If the song we are currently looking at has a higher number of followers or appearances amount (based on the desired generated list) than the song with the smallest amount, then we make the switch. After each switch, we would sort the list using bubble sort.

4 ANALYSIS

5 LEGAL CONSIDERATIONS

The main legal consideration with a recommendation system is that the user must consent to personal data being recorded. Various laws are to be considered while designing a recommendation system as the United States FTC Act empowers the Federal Trade Commission to act against unfair trade practices. The EU has a "protection of natural persons with regard to the processing of personal data and on the free movement of such data" act passed in 2016 that could be used in the context of recommendation systems. China has however passed the Internet Information Service Algorithm Recommendation Management Regulation in December 2021 that establishes the following requirements for a recommendation system

- Establishing a management system and service regulations
- Review and constant evaluation of the system
- Mechanism to moderate illegal and undesirable content
- Providing users options to turn off such a system if desired
- Guaranteeing the safe use of algorithms for elderly users

6 ETHICAL CONSIDERATIONS

Due to the personal nature of recommendation systems, there are some key ethical considerations to take into account while designing a recommendation system. One of the first concerns in such a system is the attack on personal autonomy. Recommendation systems tend to "recommend" a particular product to the user, pushing the user to the recommended product instead of the user making an independent decision on his own. Such systems may drive a user towards a more commercial product, instead of recommending one that is overall beneficial to the user, thereby unethically maximizing profit. The second concern with such a system is unethical user profiling. Since these systems generally have a good amount of the user's personal data, they may draw unethical conclusions which may lead to profiling. There are privacy concerns with such systems as well since they store and often remotely process a user's private data. Since these systems use complex codes and predictions to recommend a song, they are difficult to explain to the normal user, which may lead to unfair user assumptions. Because such systems make a prediction by reading a large amount of data, it may happen that some flawed data may lead to unethical/ socially harmful predictions. According to the research done by Milano Et. al. [5], the following ethical considerations apply to such a system, and care should be taken to minimize them:

- Inappropriate content:- Some content recommended to the user may be inappropriate according to the user's ethical/social values
- Privacy:- Special care must be taken to keep the user's data confidential.
- Autonomy:- The user should still be able to make autonomous decisions despite the presence of a recommendation system.
- Opacity:- The inner working of the system can be hard to explain to the everyday user
- Fairness:- These should be no commercial predictions and predictions should fairly consider all options.
- Social Impact:- Such a system can have a transformative impact on society as a whole and this impact should be considered while designing the system.

7 LESSONS LEARNED

8 STATUS & FUTURE WORK

We currently worked on the design, and implementation of the database and respective tables, and writing the code for loading data into the database. As we were running the code to load the data, we found that there was a bug where the insert query was not working for tables that had more than one column excluding the id column. We tried all different ways to use the tuple feature that Python MySql's execute function provided, but we weren't able to fix the issue. We plan on finding a solution for this bug by the next phase. We also planning of implementing the recommendation system and work toward the analysis portion.

9 CONCLUSIONS

REFERENCES

[1] Kaiyuan Bai and Kyoji Kawagoe. 2018. Background Music Recommendation System Based on User's Heart Rate and Elapsed Time. In *Proceedings of the 2018 10th International Conference on Computer and Automation Engineering* (Brisbane, Australia) *(ICCAE 2018)*. Association for Computing Machinery, New York, NY, USA, 49–52. https://doi.org/10.1145/3192975.3193013

[2] Dr. Daqing Chen. 2019-09-21. *UCI Machine Learning Repository: Online Retail II*. Retrieved September 22, 2022 from https://archive.ics.uci.edu/ml/datasets/Online+Retail+II

[3] Abba Suganda Girsang, Antoni Wibowo, and Edwin. 2020. Song Recommendation System Using Collaborative Filtering Methods. In *Proceedings of the 2019 The 3rd International Conference on Digital Technology in Education* (Yamanashi, Japan) *(ICDTE 2019)*. Association for Computing Machinery, New York, NY, USA, 160–162. https://doi.org/10.1145/3369199.3369233

[4] Pedro S. Lopes, Eduardo L. Lasmar, Renata L. Rosa, and Demostenes Z. Rodríguez. 2018. The Use of the Convolutional Neural Network as an Emotion Classifier in a Music Recommendation System. In *Proceedings of the XIV Brazilian Symposium on Information Systems* (Caxias do Sul, Brazil) *(SBSI'18)*. Association for Computing Machinery, New York, NY, USA, Article 41, 8 pages. https://doi.org/10.1145/3229345.3229389

[5] Silvia Milano, Mariarosaria Taddeo, and Luciano Floridi. 2020. Recommender systems and their ethical challenges. *AI SOCIETY* 35 (12 2020). https://doi.org/10.1007/s00146-020-00950-y