Spotify Playlist Challenge

Project Proposal

Brendan Ryan
University of Colorado Boulder
Boulder Colorado USA
brry3186@colorado.edu

Christopher Lescinskas
University of Colorado Boulder
Boulder Colorado USA
chle8754@colorado.edu

Matthew Martin
University of Colorado Boulder
Boulder Colorado USA
mama4085@colorado.edu

ABSTRACT

This document outlines the project plan for datamining on the Spotify Million Playlist Dataset (MPD) to determine at what confidence a playlist can be generated/extended from a given prompt. This plan will detail the problem statement/motivation, provide a literature survey or related work, propose work that the team will perform, provide the reference to the dataset, describe the planned evaluation method of the datamining, list any tools that will be used by the team and their purpose, and provide a milestone of activities to be performed.

1 Problem Statement/Motivation

There is a great deal of value in discovering patterns of like users for platforms like Spotify. The ability to provide suggestions to users and auto-generate playlists, with a high rate of user acceptance, provides stickiness and value to the user. The more value that the user experiences from this kind of app, the more likely they are to continue to pay for service and recommend to other users. According to Digital Media Association's Annual Music Report (March 2018)

> "54% of music consumers report that playlists are replacing albums for them."[1]

With this level of engagement, providing playlist with high user acceptance is vital to maintaining engagement with users and staying relevant in the digital music delivery space.

In this project, we imagine ourselves in the role of the streaming music provider, seeking to improve engagement by providing a feature to suggest new songs or artists for a user to listen to. Our primary goal is to provide suggested extensions to a playlist prompt. We are also curious about trends that may be discernable within the data, such as whether a song tends to appear at the beginning or end of a playlist, the likelihood for two songs to appear on the same playlist, or the degree to which a playlist has varied albums or artists will also be explored.

2 Literature Survey

Ludwig et al. (2018) describe a method for music recommendation using the MPD that combines multiple analyses into a hybrid model. The team used Item-Based Collaborative Filtering (ITEM-CF), Session-based Nearest Neighbors (S-KNN), and other analyses to generate initial models for recommendation. Models were then combined using techniques such as filling weighting and switching. Results and effectiveness calculations are given for all base and hybrid models.

Kelen et al. (2018) also submitted to the 2018 RecSys challenge using the MPD. Their approach focused on k-nearest-neighbor analysis to recommend additional tracks. The team then used different methods to improve their model such as amplification, weighting by inverse item frequency and weighting by item position.

As both papers are submissions to the 2018 RecSys challenge, they are directly related to our work and

demonstrate the work that has been done thus far on the dataset.

3 Proposed Work

Our team will evaluate the dataset provided as a part of the AIcrowd *Spotify Million Playlist Dataset Challenge*[2] and perform datamining to generate suggested playlists (based off of a small prompt) that have a high correlation to historical user generated playlists. We will also explore relative placement of songs in playlists to determine if there is any meaningful order of song placement within a playlist.

We plan to evaluate the dataset for basic statistics (total unique song count, frequency of songs, average playlist length, etc.). We will use this as a starting point for understanding the data that we have.

We will use 2/3 of the data set in our modeling and split off 1/3 of our data set to use as validation of our model as a means of classifier/predictor evaluation per *Data Mining: Concepts and Techniques*.[3]

We will explore different pattern evaluation methods to determine a good fit for our data e.g. Market Basket Analysis, Apriori Algorithm, K-nearest neighbor classification, etc. Our current assumption is that songs that frequently appear together in user generated playlists will receive a higher acceptance from users as playlist extensions when given the prompt of one or more of the songs. We will start our evaluation with the assumption that song title is the only parameter to determine correlation. Should this method prove unfruitful, or should we have sufficient time, we will explore if other parameters provided by the dataset to determine if there is a better predictor.

4 Data Set

Our dataset can be found here: https://www.aicrowd.com/challenges/spotify-million-playlist-dataset-challenge, and has the following statistics:

Metric	Value
Number of Playlists	1,000,000
Number of Tracks	66,346,428
Number of Unique Tracks	2,262,292
Number of Unique Albums	734,684
Number Of Unique Artists	295,860
Number Of Unique Titles	92,944
Number Of Playlists with Descriptions	18,760
Number Of Unique Normalized Titles	17,381
Avg Playlist Length	66.346

The following is a sample of attributes contained in the dataset:

Attribute Name	Attribute Type	Value
pid	Ordinal	Integer
name	Nominal	String
description	Nominal	String
num_artists	Numeric – Ratio-scaled	Integer
num_albums	Numeric – Ratio-scaled	Integer
num_tracks	Numeric – Ratio-scaled	Integer
duration_ms	Numeric – Ratio-scaled	Integer
collaborative	Binary	Boolean
track_name	Nominal	String
album_name	Nominal	String
artist_name	Nominal	String
pos	Ordinal	Integer (zero-based)

The dataset, built by researchers at Spotify, is sampled from more than 4 billion public playlists on Spotify, created between January 2010 and November 2017. Unfortunately, some "fictitious tracks" have been introduced to playlists by the authors of the dataset, as well as some "cherry picking" to filter out "offensive" content, so no real-world conclusions can be drawn from the dataset. Regardless, the dataset is still amenable to data mining techniques and will provide a sound platform to explore the methods that we have been exposed to in this course.

5 Evaluation Methods

1/3 of the original data set will be segregated to use for evaluations. We will assume that a suggested playlist extension should have a high correlation to a user generated playlist. We will evaluate matches of our suggested algorithm to the segregated data through methods like support-confidence framework. We would expect a relatively high correlation between our suggested playlist and a user generated playlist that has the same prompt song(s). One of the measures we will use to determine this will be lift.

We will also explore other means for visualizing and evaluating the results of our suggested playlist extension such as its fit in K-nearest neighbors. This may be a secondary evaluation method to understand if there are natural clustering of songs. If this behavior is found to exist, it could make suggested playlist extensions a function of finding the K-nearest neighbors from the prompt song(s).

To determine if there is a common positional placement of a song in a playlist, we will start by exploring basic statistics on the dataset. Presumably if there is no bias towards the beginning or the end of a playlist for any given song, then they should fall within some normal distribution around the mean length of a playlist. If, however, we find that certain songs fall significantly outside of normal variation of mean, then it would be an indication that there is a strong preference for that song to either be at the front or end of a playlist.

6 Tools

GitHub – Will be the primary storage and version management system for this project. It will house the dataset, code used to evaluate the dataset, and versioning of reports and visualizations.

Jupyter/Python3 – Will be the working environment for building and testing code related to dataset evaluation.

Tableau – Will be used to build data visualizations as needed to support findings.

MPD – Check.py to confirm that the dataset is correct/uncorrupted. Print.py and Show.py to print and show a subset of the data respectively. Stats.py and Deeper_stats.py to iterate through MPD and show summary information.

SQLite/MySQL – To query data as needed.

7 Milestones

Below is the proposed milestone schedule for this project. This has been updated from our original project proposal to include activities that have been completed up to the date of submission.

Item	Target	Complete
Data set basic statistics		
(practice accessing data and get basic statistics of data set)	31-Oct-22	31-Oct-22
Apriori Algorithm (1st cut)	11-Nov-22	9-Nov-22
Data Analysis Complete and Tuned	18-Nov-22	18-Nov-22
Project Final Report Draft	18-Nov-22	18-Nov-22
Progress Report	28-Nov-22	28-Nov-22
Project Final Report	8-Dec-22	
Project Code and Descriptions	8-Dec-22	
Project Presentation	8-Dec-22	
Peer Evaluation	8-Dec-22	

Figure 1: Proposed Milestone Schedule

7.1 Milestones To-do

We have taken an initial look at the basic statistics of the data and have a working Apriori algorithm. We will continue to refine the Apriori algorithm as well as look into other data mining techniques such as K-nearest neighbors for our final look at the data.

The list of incomplete milestones can be found in the above Figure 1 as *Items* with no associated *Complete* date. These include things like finishing the *Final Report*, saving and commenting on code, development of the *Project Presentation*, and *Peer Evaluations*.

8 Dataset Basic Statistics

The following is an initial look at the dataset provided for the Spotify Playlist Challenge.

#of Playlists	1,000,000
# of tracks	66,346,428
# of unique tracks	2,262,292
# of unique albums	73,4684
# of unique artists	29,5860
# of unique titles	92,944
# of playlists with descriptions	18,760
# of unique normalized titles	17,381
Average playlist length	66.346

For our assessment we will be focusing primarily on relation of tracks. With 1,000,000 playlists and 66,346,428 tracks (of which 2,262,292 are unique), we believe we will have a sufficient dataset to evaluate what relation, if any, tracks have with each other and if track alone is sufficient for recommendations of playlist extensions.

The top tracks are *HUMBLE.* by Kendrick Lamar (46,574 instances; 4.66% of all playlists), *One Dance* by Drake (43,447 instances; 4.34% of all playlists), and *Broccoli (feat. Lil Yachty)* by DRAM (41,309 instances; 4.13% of all playlists).

Only 665 songs appear have an instance rate of 10,000 or more. Even if we assume that a song can only appear on any playlist once, this means that at best, these songs only appear on 1% of all playlists. The rest of the 2,261,627 tracks will have a reoccurrence on increasingly smaller number of playlists.

A look at a histogram of playlist length indicates that 20-30 tracks is the highest concentration of tracks when we look at all of the playlists. The intent of our investigation is not to determine the optimum number of songs for an engaging playlist so we will simply use these values as an assumed benchmark for the length of playlist extension that we provide.

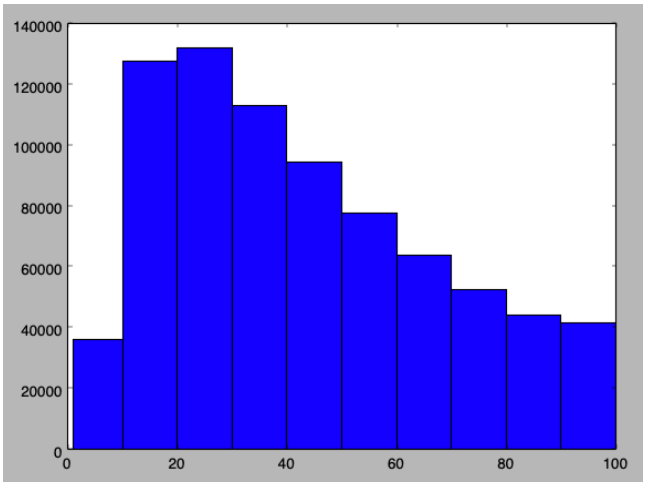

Figure 2 Histogram of Playlist Length

If we look a little more at the data we see that the first quartile is at 26 tracks, the median is 49 tracks, the average is 66, and the 3rd quartile is 92 tracks. This indicates that the data is right skewed (which we could also see in the above histogram). A box plot of the full data set in terms of playlist length is below.

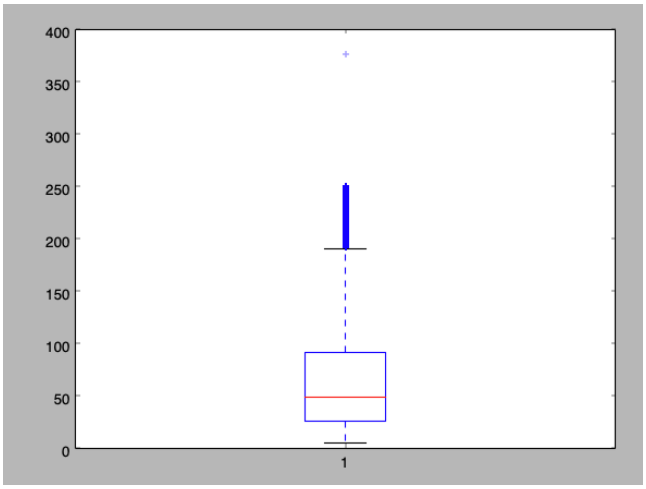

Figure 3 - Box Plot of Playlist Lengths

This information is encouraging for our investigation. As most playlists are of a reasonable size, determining relationships amongst songs should be at least probable even if the occurrence for any given song happens to be relatively low.

9 Results So Far

Although our analysis of the data set is not complete we have been able to get some initial models working. So far, we have a network diagram of a single file of playlists. The current algorithm and presentation of the model will need to be improved if we are to be able to extend this to a larger dataset.

Figure 4 - Full Network Diagram of First Set of Playlists

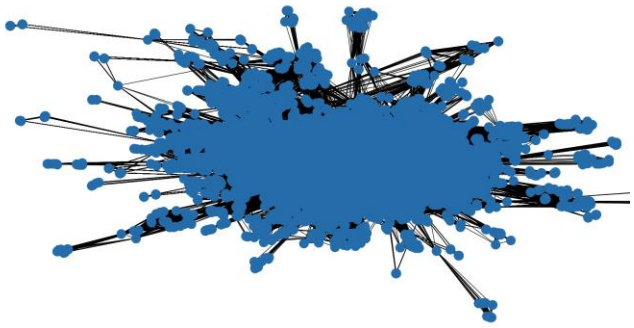

Figure 5 - Zoomed in Network Graph of the Main Cluster of Figure 4

We have also have a working Aprio algorithm using the Efficient-Apriori python package. The algorithm still needs a little cleanup to run more efficiently on such a large data set. We are also working on getting the output to match the desired output for evaluation by the challenge. The algorithm extracts the seed playlists from the provided challenge dataset. It then feeds each song from each seed playlist into an algorithm that scans the database for playlist matches with that song, runs an apriori analysis and returns the top candidate. Playlists generated from seeds are then saved in csv files.

In works is a K-Nearest Neighbors algorithm. We're using the "Nearest Neighbors" library from "scikit-learn", though we're still working to adapt the dataset (contained in .json files) to the needs of the library.

With the data that we have so far, we have drafted about four (4) pages of our Final Report. Sections needing to be completed are finalizing the *Abstract* (needs the Apriori and K-Nearest Neighbors algorithms finished), completing the introduction, adding details about Apriori and K-Nearest Neighbors (once complete), and our conclustion.

REFERENCES

[1] DiMA. Digital Medial Association: Annual Music Report; **A MIDIA Research Report**. (March 2018). Retrieved October 16, 2022 from https://dima.org/wp-content/uploads/2018/04/DiMA-Streaming-Forward-Report.pdf

[2] AIcrowd. Spotify Million Playlist Dataset Challenge: A dataset and open-ended challenge for music recommendation research. Retrieved October 16, 2022 from https://www.aicrowd.com/challenges/spotify-million-playlist-dataset-challenge

[3] Jiawei Han, Micheline Kamber, Jian Pei "Data Mining: Concepts and Techniques", 3rd Edition, Morgan Kaufmann, 2011. '

[4] Ludewig, M., Kamehkhosh, I., Landia, N., & Jannach, D. (2018). Effective nearest-neighbor music recommendations. In *Proceedings of the ACM Recommender Systems Challenge 2018* (pp. 1-6).]

[5] Kelen, D. M., Berecz, D., Béres, F., & Benczúr, A. A. (2018). Efficient K-NN for playlist continuation. In *Proceedings of the ACM Recommender Systems Challenge 2018* (pp. 1-4).