Sadie Harper
DSC680 – T301 Applied Data Science
Professor Williams
10/23/2022

Business Problem

Recommendation systems can be utilized for various purposes. Such examples include personal online shopping, companies ordering parts for manufacturing, people looking for new books, podcasts, or songs, and more. For this project I will focus on recommending songs for users to add to their playlists. Being able to contribute new songs to these users can help expand their horizon, potentially introducing them to new songs, artists, or genres similar to what they already like. Products such as Spotify, Apple Music, and Amazon Music are examples of companies that can greatly benefit from quality music recommendation systems. A company that can tailor suggestions to their users improves customer satisfaction, retains listeners, and helps bring in new people generating more sales.

Background

Spotify created this million playlist dataset so that it could be used to build a recommendation system to improve users personal playlists. They found that their users love to both listen to curated playlists and create their own for a number of different reasons. Someone may want a playlist for a party, driving, singing in the shower, or to use as a way to organize their favorite songs. Finding a way to help users add to these playlists can allow users to continue listening to what they love while introducing them to new ideas and possibly new music.

Data Explanation

The Spotify playlist dataset contains 1 million playlists. Within these playlists there are more than 2 million tracks from almost 300,000 artists. The playlists were sampled from users in the United States who created them from January 2010 and November 2017. I also incorporated

audio data to match up with the tracks in the playlists using Spotify's API. See Appendix A for a data dictionary with more details on each column.

Data Preparation

Target

This project will have multiple targets, those being a list of songs that a user might want to add to their playlist. Ultimately, that would be the Spotify track URI so that the song could be easily made visible in the app for the user. The target pool of songs was created from all the tracks found in the sample playlist dataset.

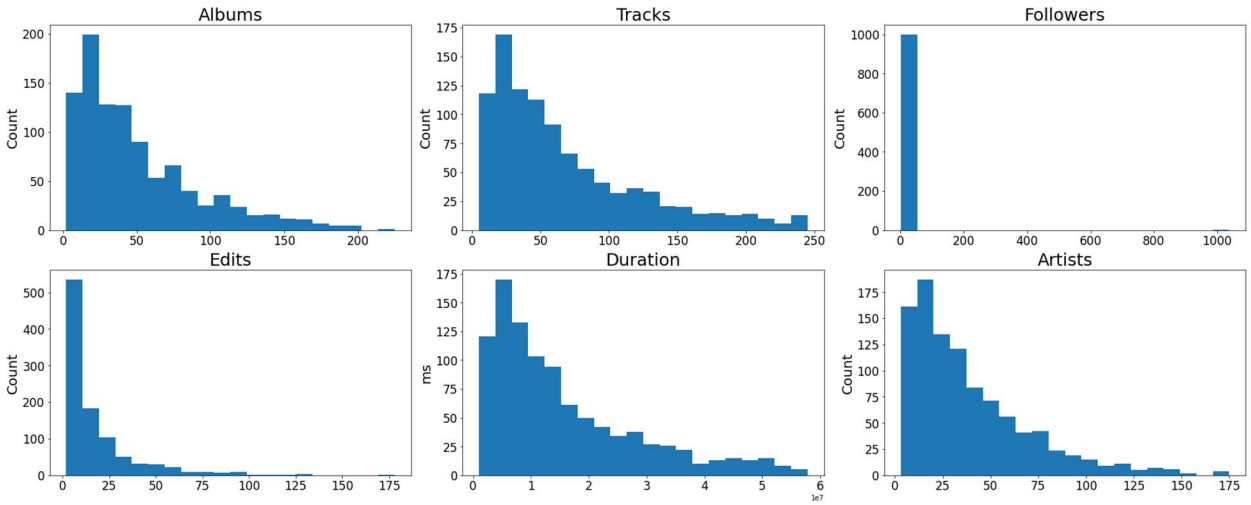
Features

The dataset contains a lot of features around both the playlist and the songs that make up each playlist. Because this data was created by Spotify for this project, it was gathered to be clean. There are no playlists with incomplete data and it has been filtered to remove most offensive material. With the sample data I have explored I have not yet found any discrepancies, missing values, or mistakes that need to be accounted for. Preparation of the features has been a time consuming task. To help with this, I am only working with 1,000 playlists. These playlists all have a varying number of songs in each one. There are 10 features that describe the playlists, and 20 features that describe the tracks. Most of the beneficial information are audio features that describe a song, another large chuck of features are descriptions like names and unique identifiers which will be excluded for the recommendation system. As you can see in Figure 1, all of the numerical features that describe the playlists are positively skewed to some extent. The number of followers is the only feature to give off the look of extreme outliers.

Figure 1

Sadie Harper
DSC680 – T301 Applied Data Science
Professor Williams
10/23/2022

Numerical Features for Playlists Histogram

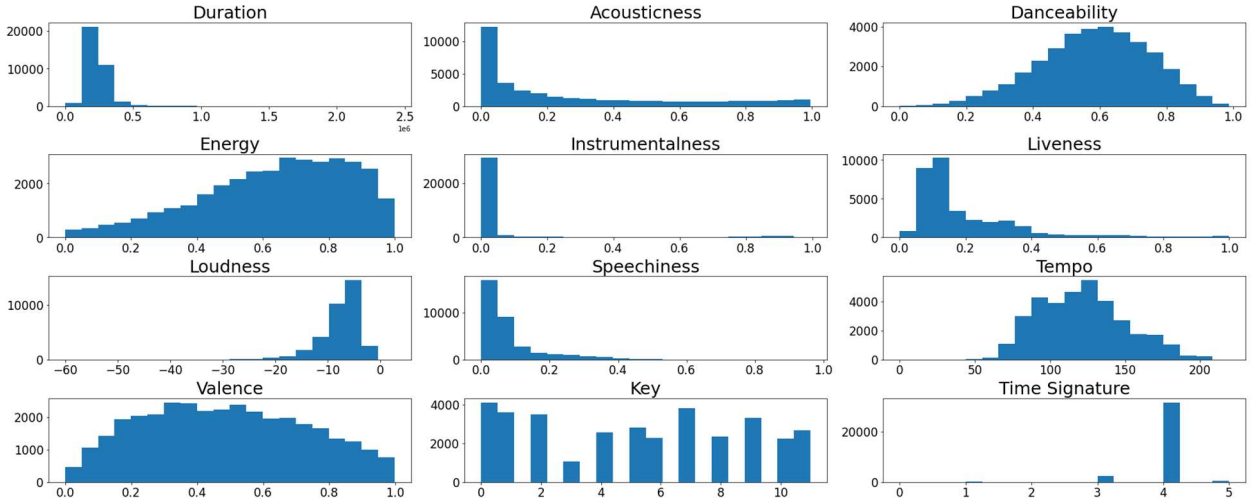


Note. The distributions in this feature set are positively skewed.

The numerical features that describe the individual tracks do not share the same distributions, although there is more variation which can be seen in Figure 2. Danceability, tempo, and valence all have a fairly normal distribution. Loudness, and time signature are negatively skewed, and everything else has more of a positive skew. The visual also shows the distribution of key which is a categorical variable.

Figure 2

Numerical Features for Tracks Histogram



Note. The audio features have a mix of distributions.

Methods

There are many ways to build a recommendation system. Each type has a different set of pros and cons, and for this first system I wanted to go with something easy to understand and explain. For that reason, I opted to use cosine similarity. See Appendix B for a detailed explanation of this method. For this method, I ended up using only audio features in the analysis. I one-hot encoded the categorical features, and used a min max scaler for the numerical features. This left me with a final dataset ready to be used for modeling. To generate song predictions I would summarize the features of every song in a playlist to get one value for each feature. These values would then be compared against the song list to generate the top 50 song recommendations. I also made sure to exclude all songs already in the playlist from the recommendations.

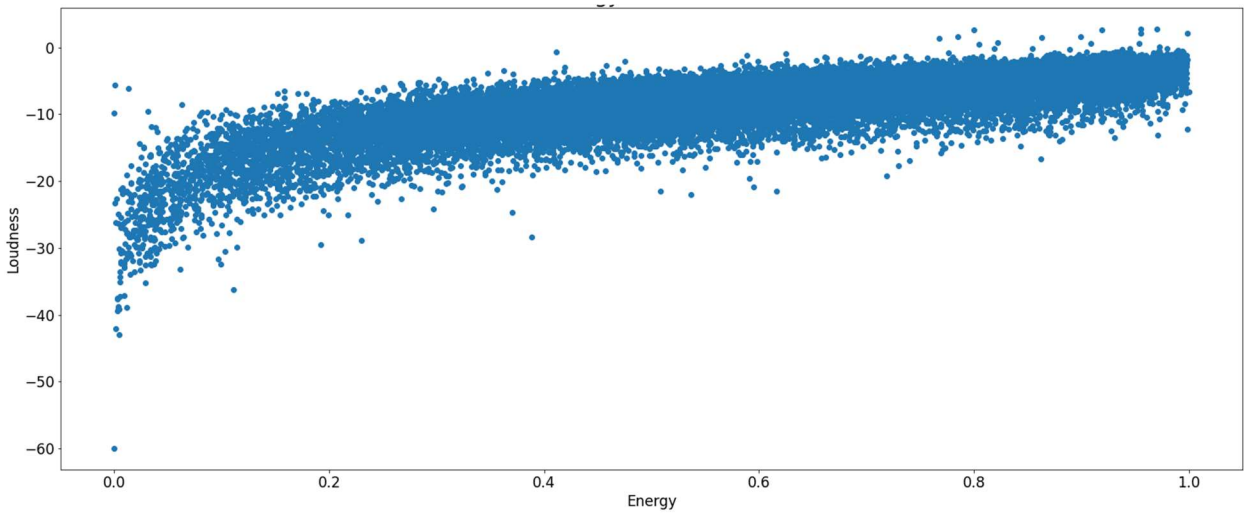
Sadie Harper
DSC680 – T301 Applied Data Science
Professor Williams
10/23/2022

Analysis

To understand the similarity recommendation system, I needed a good understanding of the Spotify data and which features had the strongest relationships. In Figure 3, you can see that energy and loudness have a somewhat strong positive relationship.

Figure 3

Energy vs Loudness Scatter Plot

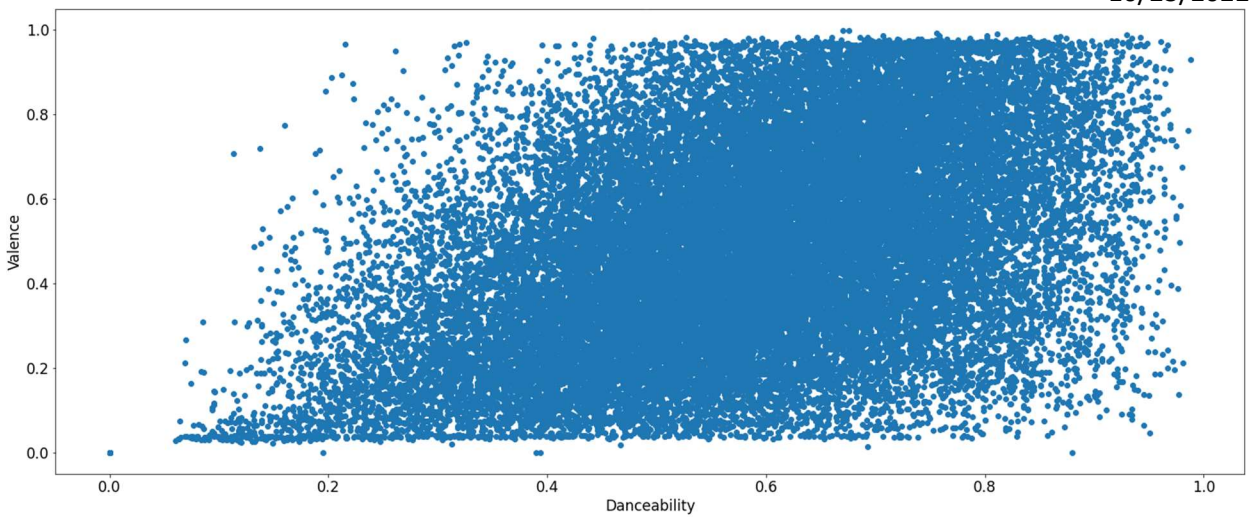


Note. As energy increase, the songs get quieter.

Danceability and valence show a similar relationship, as both can be seen increasing with one another in Figure 4.

Figure 4

Danceability vs Valence

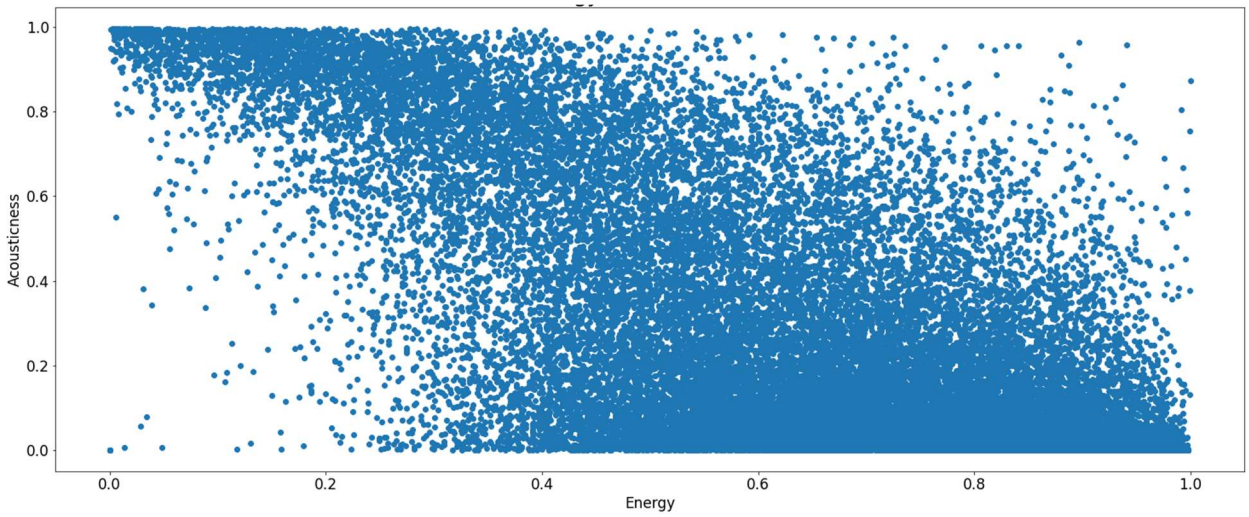


Note. As songs get more danceable, the valence increases.

Lastly, Figure 5 shows a negative relationship in the data. The more acoustic a song is, the less energy is has. This meets a assumption that acoustic songs are less energetic than non-acoustic songs.

Figure 5

Energy vs Acousticness



Note. As energy increases, the song becomes less acoustic

After I had a solid data understanding I tested out the cosine similarity method. I was able to generate suggested songs that were over .90 similar to the playlists they were suggested for.

Conclusion

After completing a simple recommendation system I found that there is not an easy way to test how well it is working. This is because, accuracy would need to be measured based on if the users ended up adding any of the recommended songs to their playlists. With that in mind, I still feel confident that looking at track audio features will generate some similar suggestions that users would want to add. I also assume there will be a mix of good and bad recommendations in the top 50 songs.

Assumptions

Since this data was collected and provided by Spotify, I am assuming that it is of high quality. I found no mistakes while exploring the data so I feel as though this is a safe assumption. I also am assuming the pool of songs to use as recommendations can come from the songs found in the provided data. Eventually, this would need to change so that the song pool can include all songs within Spotify and not just those found in the same playlists.

Limitations

The main limitation I faced in this project was the overall size of the data. Working with a dataset of this size is very compute intensive. Right away I found that my personal laptop was taking a lot of time to even prepare the data, let alone the time it would take to train with this

many records. Limiting the data to a 1,000 playlist sample was the best way to deal with this in the short four week time frame. I also feel that the final dataset used had some limitations. The only features I ended up using were audio features, which can be similar across many genres of songs whereas the playlists may focus on one genre.

Challenges

One challenge I faced was not capturing user behavior. Users may edit their playlists overtime and that can take it in a different direction. The recommendations would need to change with the user which might be difficult for a model to recognize in the beginning. Additionally, I have never prepared data that was stored in such a way before. It has been a personal challenge to prepare the JSON data in a more workable format.

Future Uses

The main goal of future use for this model would be to improve the user experience. This is because, if Spotify can recommend songs that their users think fit in perfect with their playlists, they are going to have a better experience with the platform as a whole. This model could also be used to inform Spotify of their user behavior which can help within other areas like marketing and product development.

Recommendations

One way to improve this model would be to update the training data. Playlists from 2010 are likely very different from playlists today. While 2017 was not that long ago, the music industry can change drastically in that time. Updating the training data can improve the quality of the predictions because it can include more recent music. Another way to improve the model would

be to bring in more metrics. Spotify has more descriptive data on the artists, albums, and songs that could really improve the recommendations. Even just adding genre could go a long way in improving the system.

Implementation Plan

I will implement this model in the form of a report that describes the process and the outcomes of a model. In a real world scenario, this model should be implemented in a way that can provide real time predictions to the users. It would also require methods that track the accuracy of predictions so that it could be quickly retrained when needed based on playlist changes and new songs.

Ethical Assessment

Recommendation systems can pose a number of ethical concerns. The main considerations are with security and privacy. For a recommendation system to provide good results, it needs to have a good understanding of the individual the recommendation is for. Some people find that this can cause concerns for that persons privacy. That same idea can lead to security concerns because the model is using data that is personal to a specific individual. That information could be leveraged in negative ways if not kept secure and used for the right reasons.

References

7 Critical Challenges of Recommendation Engines. Appier. (2021, February 2). Retrieved October 2, 2022, from https://www.appier.com/blog/7-critical-challenges-of-recommendation-engines

C.W. Chen, P. Lamere, M. Schedl, and H. Zamani. Recsys Challenge 2018: Automatic Music Playlist Continuation. In Proceedings of the 12th ACM Conference on Recommender Systems (RecSys '18), 2018.

Lam, S.K.“., Frankowski, D., Riedl, J. (2006). Do You Trust Your Recommendations? An Exploration of Security and Privacy Issues in Recommender Systems. In: Müller, G. (eds) Emerging Trends in Information and Communication Security. ETRICS 2006. Lecture Notes in Computer Science, vol 3995. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11766155_2

Prabhakaran, S. (2022, April 20). *Cosine similarity - understanding the math and how it works? (with python)*. Machine Learning Plus. Retrieved October 23, 2022, from https://www.machinelearningplus.com/nlp/cosine-similarity/

Spotify Million Playlist Dataset Challenge: Challenges. AIcrowd. (n.d.). Retrieved October 2, 2022, from https://www.aicrowd.com/challenges/spotify-million-playlist-dataset-challenge

Web API reference: Spotify for developers. Home. (n.d.). Retrieved October 23, 2022, from https://developer.spotify.com/documentation/web-api/reference/#/operations/get-audio-features

What is a Recommendation System? NVIDIA Data Science Glossary. (n.d.). Retrieved October 2, 2022, from https://www.nvidia.com/en-us/glossary/data-science/recommendation-system/#:~:text=Recommender%20systems%20are%20highly%20useful,data%20gathered%20about%20their%20interactions.

Sadie Harper
DSC680 – T301 Applied Data Science
Professor Williams
10/23/2022

Appendix A

Data Dictionary

Feature	*Description*
Playlist Name	User created name of playlist
Collaborative	T/F: Identifies if playlist is collaborative or not
PID	Playlist ID
Modified At	Time playlist was modified
Number of Albums	Total albums in playlist
Number of Tracks	Total tracks in playlist
Number of Followers	Total followers playlist has
Number of Edits	Total times user has edited playlist
Duration (ms)	Length of playlist
Number of Artists	Total artists in playlist
Tracks Data:	
Position	Position of song within playlist
Artists Name	Name of artist
Track URI	Spotify URI for song
Artist URI	Spotify URI for artist
Track Name	Song name
Album URI	Spotify URI for album
Duration (ms)	Length of song

Sadie Harper
DSC680 – T301 Applied Data Science
Professor Williams
10/23/2022

Album Name	Name of album
Acousticness	How acoustic a song is on a scale of 0 – 1
Danceability	How suitable the song is for dancing on a scale of 0 – 1.
Energy	A measure of intensity and activity on a scale of 0 – 1.
Instrumentalness	A measure of vocals included on a scale of 0 – 1, close to 1 meaning no vocals included
Liveness	Depicts audience noise on a scale of 0 – 1, 1 being the most audience noise
Loudness	Average decibels across the entire song
Speechiness	The presence of spoken word on a scale of 0 – 1, 1 being all spoken word
Tempo	The overall beats per minute in the song
Valence	Describes how positive a song is on a scale of 0 – 1, 0 being sad/angry and 1 being happy/euphoric
Key	The key of the track using standard pitch class notation
Mode	Major or minor of the track, 1 is major and 0 is minor
Time Signature	Estimated measure of beats in each bar

Sadie Harper
DSC680 – T301 Applied Data Science
Professor Williams
10/23/2022

Appendix B

Cosine Similarity

Cosine similarity is a mathematical way to determine how similar two elements are. It measures the distance between two values using the cosine of the angle between the two vectors in a multi-dimensional space. These vectors are usually arrays of numbers that describe an object. In the instance of this recommendation system, they were arrays of different audio features. This measure also focuses on the orientation of the angle versus the magnitude, or the more traditional way of calculating distance. This reason is why cosine similarity is a good option at finding similar objects when comparing against multiple elements.