Jacob Kurstedt

**Final Report:**

**Music Recommender**

**Problem Statement (How well a Player will do in the NFL):**

I have always really enjoyed listening to music and I wanted to be able to find recommendations for songs that I like. Also, it would be a way for me and others to discover new songs and artists which would expand our musical palette

**Data Wrangling/Cleaning:**

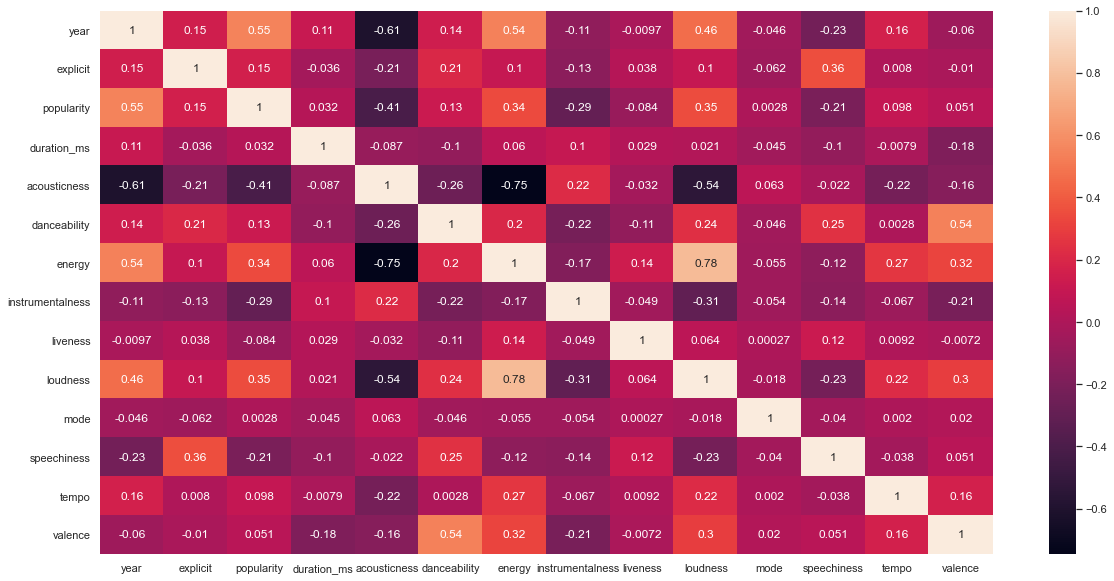
The first thing I had to do was find a song dataset. I went to Kaggle and there was a dataset that had over 150 thousand songs which was exactly what I was looking for. <https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks>. After getting the data I had to clean it I first stripped unnecessary characters from the artists column and merged the genre data set to show the genre for each song. I then changed the key column to have the actual key strings instead of being numeric. After that I made all the song names lower case so when people input in song names, they could type in a song anyway they wanted. Lastly, I had to drop duplicate songs.

**Exploratory Data Analysis:**

The main parameters that I was looking at were the metrics that Spotify uses which are:

* Acousticness
* Danceability
* Energy
* Instrumentalness
* Liveness
* Loudness
* Speechiness
* Tempo
* Valence

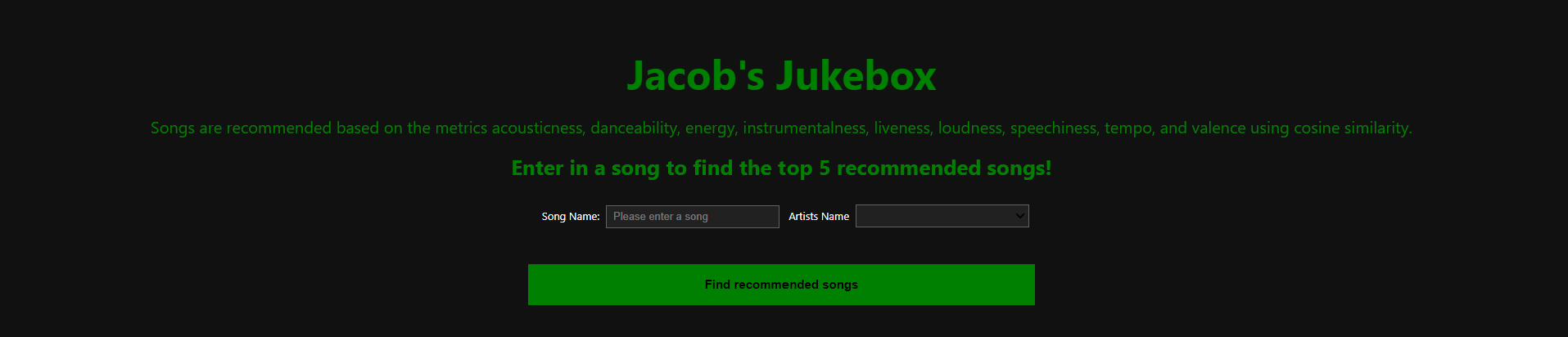
I explored each of these features individually to see if there was anything interesting about them. I created graphs for each feature and filtered the data by if they were minor or major and if the song was explicit or not. The most interesting things that I discovered was that explicit songs had a lot more danceability to them than other songs. I found this interesting because with house and edm music there is not much explicitly in them and I feel that those types of genres are way more dancy. Also, it was interesting to see that explicit songs were overall more popular. I think this is due to the popularity being rated based of recent popularity of the song and not peak popularity, and songs today are more explicit than in other years. Thus, the explicit songs of today are more popular than older songs.

I then looked at the correlations between the features and the things that popped out to me the most were that energy and loudness had the best correlation when looking at the popularity of the song. While acousticness and instrumentalness were the worst. This is probably because songs today are more upbeat than songs in the past and since popularity is based on recent popularity the higher energy songs of today are more popular. Also, that danceability and energy had the best correlation for valence which is the positivity of the song. This makes sense to me because upbeat songs that make people happy need to be of high energy and somewhat dancy. 

**Preprocessing**

During preprocessing the first thing I had to do was change the metric values from float64 to float32 in order to save memory. I then used a MinMaxScaler on the metrics. Then I preformed Cosine Similarity, Euclidean Distance, and Pearson Correlation using different recommendation functions that I created. All of them preformed similar and recommended similar songs however, I decided to go with using Cosine Similarity because I thought it gave the most accurate recommendations. After choosing Cosine Similarity I had to run two separate cosine similarities (reset kernel ran one each time) one that had songs from 1980s and up and one 1979 and down. The reason I had to do this was because of memory. After preforming this I created two new dataframes that held the id, name, artists, and the top 5 recommended songs(dictionary) for each run through which allowed me easily extract the top 5 recommended songs without having to run my functions each time. This saved a lot of time and memory.

**Modeling/Application:**

There was not much I had to do when it came to modeling the only thing I had to do was read in both the 1980s\_up dataframe and the 1980s\_down dataframe and merge those together so I could have all the songs I needed. Once this was done, I was able to create the application. Using ipywidgets I was able to create an input box where users could plug in the song, a dropdown menu with artists names who had similar song names. Then a button which would display the output of the top 5 recommended songs when clicked. After this using voila and Heroku I was able to deploy the application. Here is the link to the website [<https://jacobs-jukebox.herokuapp.com/>](https://jacobs-jukebox.herokuapp.com/).

**Conclusion:**

I believe using Cosine Similarity was the best choice when making the recommendations. When I was testing out the different recommenders the song it recommended I thought were the best. I had a lot of fun creating this project and hope to continue to build on it.

**Moving Forward:**

In the future I will continue to add songs to the dataset so that there are more types of songs to be recommended. I will try to make it so people can create playlists based of the songs that are recommended. Also create another button which will produce the next top 5 songs. I will also test out using less metrics or figure out which metrics I can drop to see if the recommender becomes more accurate. Lastly, create a drop down list so people can also filter the songs that were recommended by genres