Jon Organ

CS677

Final Project Write Up

The overall goal of the program is to take an input song from a user and generate a song recommendation. To be able to compare the songs with machine learning I used a dataset from Spotify containing various features of over 170,000 songs. These features included a mix of continuous features such as acousticness and danceability as well as categorical features such as popularity and explicit lyrics. Since this dataset doesn’t contain any recommendation related labels, I don’t have any training data that most classifiers would require. However I can compare the features of the input song to find the nearest neighbor, and possibly expand to allow the user to put multiple input songs to find the nearest neighbor to k inputs. Using kNN also still provides flexibility in choosing which features to use, and how to weight those features.

As previously mentioned I don’t have previous recommendation labels to be able to score the program so instead I had to create some. I did this by inputting various songs to the program and judging whether I though the recommendations were good or bad. However, I thought myself being the only input on labels would skew the results towards only my own music preferences so I had my family and friends input songs and judge the output recommendations as well. The result of this was an overall accuracy of 34.57% from 28 out of 81 recommendations being good. I believe the reasons for the low score was due to three areas. The first was improper feature selection. I had chosen the 9 features that seemed to best represent the song in its entirety in order to get a baseline dataset, I didn’t add or remove any. Secondly, I hadn’t yet adjusted weights for the given features to improve the recommendation. Lastly and most importantly, I realized the dataset doesn’t really reflect the way we think about music, viewing songs from a calculated overall value. To fix this, I needed to add a supplementary dataset that focuses not only on music, but also has information on songs we like to listen to together.

The solution to the previous mentioned problem is the playlist dataset. This is a dataset from Spotify as well, containing thousands of playlists, their user’s IDs, and the songs and artists they contain. I incorporated this dataset by taking the input songs and finding every playlist it was on. I saved a list of the previously mentioned playlists and found all the songs on all of those playlists. From that, I found out which song appeared the most often in this list. The song that appears most frequently means it is most often put on a playlist with the given input song.

While this was a good start, it immediately became obvious that there was a bias towards certain songs that appear on many playlists. To counteract this I reincorporated the old song analysis dataset to refine the selection of the most common neighbor classifier. Specifically, I took the ten most common songs, and found them in the song analysis dataset. Then I computed their euclidean distance from the input song and instead returned the song of the ten that was closest in analysis to the input song. This gave a new classifier parameter. Instead of just returning the one most common song, it can now return the k-most common songs to work from. While this does offer more flexibility with the classifier, it also increases the difficulty of finding the parameter values for both k most common neighbors and weights for the music data analysis that minimize the error between the two classifiers to return the best song recommendation. At this point I also realized that from my first attempt, I do have labels with which I could potentially implement other classifiers, however I quickly found the assessment dataset was not large enough to sufficiently improve any performance.

From the new system with the refined playlist classifier the accuracy came out to 75.31% from a sample of 61 good recommendations out of 81 total recommendations. I think this improvement was mostly as a result of the incorporation of the playlist dataset. While there still is improvement that very well could be made with both participating classifiers parameters, the initial set parameters seem to be fairly strong. Overall this music recommendation system seems to be doing alright, and has a decent set of parameters with which the system could be further improved. For example, the Manhattan or Minkowski method could also be used to calculate the distance for the song analysis. In addition, each classifier was contained to its own method so more classifiers could be added on top. This system still has some downsides worth mentioning including the continued difficulty in scoring this system without some additional feedback component to generate labels and calculate score on a larger test set. Additionally, the fact that taste changes over time requires the playlist dataset to also need occasional updating.