Content-Based Music Recommender

Alex Condotti
Department of ECE
Carnegie Mellon University
Pittsburgh, PA 15213
`acondott@andrew.cmu.edu`

Prateek Gaddigoudar
Department of ECE
Carnegie Mellon University
Pittsburgh, PA 15213
`pgaddigo@andrew.cmu.edu`

Vamsi Ghorakavi
Department of ECE
Carnegie Mellon University
Pittsburgh, PA 15213
`vghoraka@andrew.cmu.edu`

1 Introduction

Companies like Spotify, Apple, and Amazon provide music streaming and customized recommendations to users. These recommendations take into account music labels such as genre, and artist, along with audio features such as timbre. However, these companies primarily use collaborative filtering methods to recommend songs. Collaborative filtering is based on user models, where users in the system are recommended songs based on the listening history of similar users. Our goal is to make a music recommendation system that only uses audio features of songs listened to by the current user to recommend new songs, thus performing content-based filtering rather than user-based filtering. Our primary goal is to investigate if using only audio features will build a working song recommendation system. An audio feature based system may have the advantage of recommending songs outside of a user's typical comfort zone or demographic zone, as it might allow for genre-independent suggestions. Additionally, we want to investigate if building a recommendation system using only audio features allows for the discovery for new music that may have different genres, or artists than the listener normally prefers, but still is acoustically similar to songs that the user likes listening to.

2 Related Work

We examined existing papers and implementations in order to establish a baseline for our models. The foundation of our project is based on *Content-Based User Models: Modeling the Many Faces of Musical Preference* [1] which contains information about several models and APIs that provide data about music preferences and the features in the track, based on the LFM-1b music database. In addition, we looked at existing implementations of music recommendations described in *Item-Based Collaborative Filtering Recommendation Algorithms* [2] that recommends popular artists that a user could listen to based on the listening history of other users with similar profile.

3 Methods

We initially pulled data from the LFM-1b dataset provided by Last.fm music service, which provided tags such as song title, genre, and artist data. This data was used as input to the Spotify Web API, in order to get descriptive features for each track. The Spotify Web API provides conventional features such as loudness, tempo, and musical key. Spotify also provides its own subjective features, such as danceability, energy, and acousticness. These features are defined by spotify; as an example, danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. To determine which features to use, we examined the distributions of each feature, as shown in Figure 1. We then clustered the features using K-Means Clustering and examined the output clusters to see if the selected features resulted in good separation between clusters. For example, we determined that musical key was not a good feature due to its distribution, as well as due to the fact that musical key has little to no correlation with genre. We selected acousticness, danceability, energy, instrumentalness, liveliness, loudness, tempo, valence, and speechiness as our features for analysis. To determine the optimal amount of clusters, we used the elbow method to view the error associated with the number of clusters, and used the silhouette method to examine how well objects were matched to their own clusters. Using K-Means, we visualized the resulting clusters as a heatmap, as shown in Figure 2. We then used PCA for dimensionality reduction to visualize the clusters further. An example of resulting clusters using K-means is shown in Figure 3, where each cluster consists of combinations of features and are relatively distinct from each other.

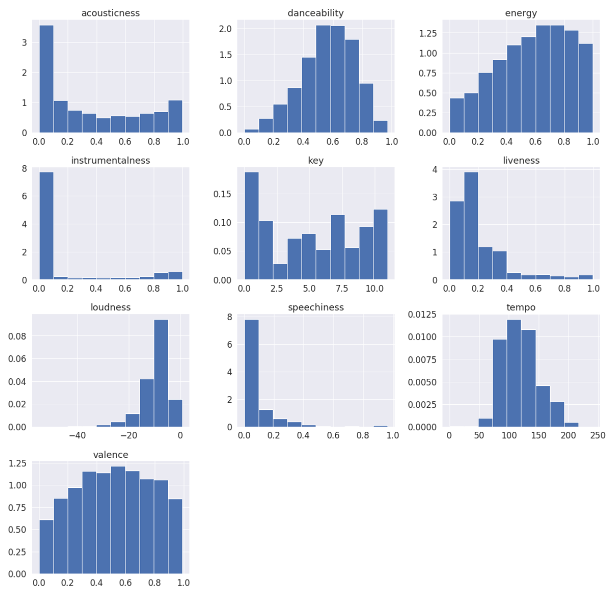

Figure 1: Spotify Features

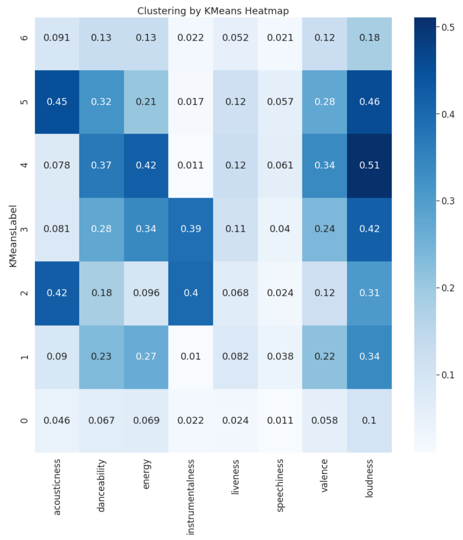

Figure 2: Heat map Visualization of K-Means Clusters

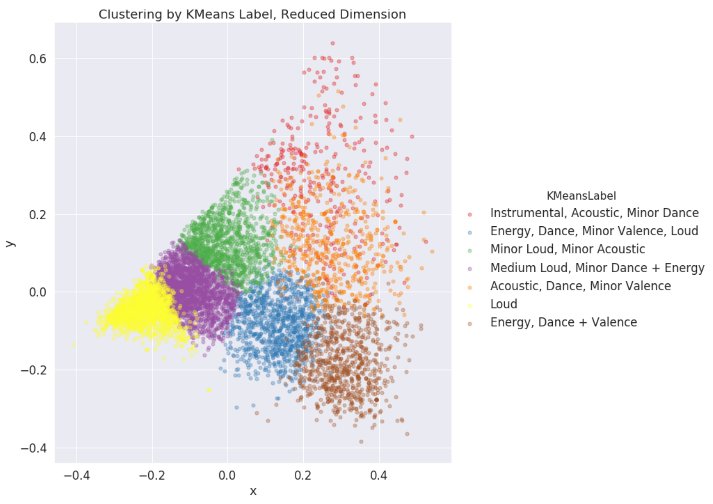

Figure 3: Example K-means Feature Clusters

After determining the ideal features, we then began to build the model for our data. As a starting point, we used the Last.fm user preference data, which comprised of lists of users and songs that they prefer- we call these user preference lists. We investigated different ways to cluster the user preference data to find the optimal number of clusters. The number of songs in a user preference list varies from user to user, and the amount of variance also is different between their song preferences. Due to this variability, it is necessary to determine an optimal number of clusters for each user input to the system. Originally, we attempted to threshold the maximum distance for each cluster, but found this method to be sub-optimal, since the variability of the user data caused the optimal maximal distance to change for each user preference list, thus making it impossible to find a true "optimum" distance. Another approach involved examining where the rate of change of the distances would start to slow down, but this method resulted in a small number of clusters, which was far less than the optimal number of clusters and did not give good results. To finally determine the number of clusters, we used a variable called the gap statistic [5], which provided a heuristic version of estimating the number of clusters for each user. The gap statistic examines the average dispersion for a set of cluster amounts, and compares it to the average value of dispersion within clusters over that range. We get the optimal number of clusters using the gap statistic [5].

Our overall model works as follows:

0. Generate spotify statistics for Base Dataset
1. Gather playlist or preference information
2. Generate statistics from spotify
3. Select and normalize the features
4. Generate the gap statistic to determine the number of features
5. Cluster the normalized data
6. Measure the distance from our base dataset to the centers of each cluster
7. Choose songs that are the closest to each cluster

4 Datasets

In order to model and test our music preference models, we used four different datasets:

1. A 10,000 song Subsample from Last.fm
2. Listening events of 120,000 users recorded into LFM-1b dataset by Last.fm music service
3. Audio features for music generated by Spotify API

4. Various Spotify playlists

The subsample of the Last.Fm dataset [1] served as our reference data from which we would provide music recommendations. It is a subset of 10,000 songs containing tags like genre, artist, and song title. Our second dataset was the user preference lists provided by LFM-1b [2]. It contains nearly a billion listening events of 120,000 users that we used as a validation dataset for our methods, and to provide objective results. This sample was around 40 GB of listening data from all around the world. So, in order to reduce the size of the data, we only looked at preference lists from users in the United States. The third dataset we used was the audio features provided by the Spotify API [3] which provides information from the acoustical features of tracks available on Spotify. We used these features as the primary way of clustering. Finally, our last dataset were several Spotify playlists that we used to subjectively validate that our model worked on a smaller scale because we could build the playlists, and see if the songs recommended sound similar.

5 Results

Due to the nature of our project, we looked at both objective and subjective results of both our proposed model, along with a more traditional collaborative filtering model. In order to objectively evaluate both models, we looked at the average Euclidean-distance from the recommended song to the center of its cluster. We also removed ten songs from our preference lists, added them to our reference dataset, and counted how many of those songs were returned by each model. To subjectively evaluate our system, we used several Spotify playlists containing jazz [4], a single album [5], 90s hip hop [6], and world music[7].

Objectively looking at the results of our system, we found that the average normalized distance from the center of the cluster to the song we recommended was 0.087, and for the furthest song, it was 0.98. For our reference model, the distance from the center of the cluster to the song recommended was 0.120, and for the worst song it was 0.980. Additionally, after the removal of 10 songs, our model would recommend 32% of the songs removed, and the collaborative filtering model would recommend 85% of the songs removed.

Subjectively, looking at the songs recommended by our feature based system, we found that the system would often recommend songs that are acoustically similar to the songs in the input playlist, but some of these recommended songs were in different languages, from different genres, or from artists that were not as famous. Inputting a playlist of hip-hop and rap artists from the 90's resulted in songs emblematic of the hip-hop genre, but spanned songs ranging from classic hip-hop to underground modern hip-hop to foreign language rap. On the contrary, the traditional recommendation system would provide songs that were acoustically less similar, but were from more established artists, and had the same genre and language.

The final metric that we looked at was the number of clusters that each user preference list and playlist would provide. Since the number of clusters varies based on its gap statistic, the average number of clusters that we found was 12.67. The fewest clusters that we found was 2 for the blues playlist, and the largest we found was 15 for the world music playlist.

6 Discussion and Analysis

Looking at the objective results for our model, we found that it would provide acoustically similar songs to the songs in the playlist. Also, it would provide songs from the playlists that were different to the songs that we removed. We found many cases where the model recommended songs from

[1]Million Song Subsample: http://millionsongdataset.com/lastfm/
[2]Listening history: http://www.cp.jku.at/datasets/LFM-1b/
[3]Spotify API: https://developer.spotify.com/documentation/web-api/
[4]Jazz Playlist: spotify:playlist:06CDn0IMasuAK0dZs1LNrk
[5]Miles Davis' Kind of Blue: spotify:album:1weenld61qoidwYuZ1GESA
[6]90s Hip Hop Playlist: spotify:playlist:37i9dQZF1DX186v583rmzp
[7]World Music Playlist: spotify:playlist:0VT4PIZ4W4FprvHLxgVMWX

different genres than the input tracks, but upon listening we were surprised to find that the output tracks contained stylistic and musical elements that could place them firmly within the original track's genre. For example, inputting a playlist of classic acoustic jazz resulted in recommendations of more classic jazz songs, but also the song "Can You Hear the Angels from contemporary gospel group Virtue, and "Steve Biko (Stir it Up)" by 90's hip-hop group A Tribe Called Quest. Examining these songs further revealed typical jazz instrumentation and rich jazz chord progressions in the gospel song, and we found that "Steve Biko (Stir it Up)" was built around a trumpet riff sampled from jazz legend Woody Shaw. Although this method of evaluation is very standard for a music recommendation system, we think that this result highlights how our model allows users to discover new songs and artists that they would not expect to hear otherwise, which adds to our original goal of music discovery.

In addition, because we are using a variable number of clusters, we can see the overall variance of the songs in a playlist. Looking at the number of clusters in a uniform playlist like the "Kind of Blue" album, we found the optimal number of clusters was 2, while a more varied playlist like "world music" had an optimal number of 15 clusters. Since the number of clusters seems to directly correlate with the types of songs within a playlist, we found that the average user would listen to between 12 and 13 types of songs.

Since this model is inherently limited because it only uses acoustical features, there are many ways to improve how the model works. One more interesting way to look at the data would be to view it as a time series version of the data, where songs that were heard more recently would be weighted more, or if songs from newer artists would be weighted more. Additionally, we could use a different or more recent dataset to recommend songs, and to also view how music preferences have changed over time.

To further evaluate the model, ideally we would create an application that can be used by the general population, and sample if users like the recommendations that we provide. Our evaluation methods cannot reliably capture how well people enjoy a song because of many confounding variables. However, since the model seems very useful for music discovery, a possible use would be to create a dataset of new tracks, and use our method to recommend songs to users that they might like.

References

[1] Zangerle E, Pichl, M. (2018). Content-Based User Models: Modeling the many Faces of Musical Preference. *Proceedings of 19th ISMIR Conference*. Paris, France.

[2] Savar, B, Karypis, G, Konstan, J, Riedl, J. (2001). Item-based Collaborative Filtering Recommendation Algorithims *WWW10*. Hong Kong

[3] Schedl, M, Ferwerda, B. Large-scale Analysis of Group specific Music Genre Taste From Collaborative Tags.

[4] Soleymani, M, Aljanaki, A, Wiering, F, Veltkamp, R. Content-Based Music Recommendation using Underlying Music Prefrence Structure.

[5] Tibshirani, R. Walther, G. Hastie, T. Estimating the number of clusters in a data set via the gap statistic. *J. R. Statst. Soc. B*. Stanford University, USA