BopBuddy:

Your ultimate companion for discovering playlists that match your unique taste!

Abbie Maemoto, Belinda Yeung, Robert Igbokwe, Jasmyn Lopez

Introduction

Discovering new music can be overwhelming due to the vast number of songs available. Current recommendation systems often struggle to capture individual tastes, leading to less personalized suggestions. This issue stems from a limited understanding of individual user preferences and the complex nuances of musical diversity. Our project aims to address this issue by using real world playlist data to train our model that can generate more tailored recommendations. By analyzing patterns in user-created playlists and the attributes of included songs, our approach aims to discern and predict user preferences with greater accuracy than conventional recommendation systems. Overall, we hope to simplify the process of finding new, enjoyable music for users!

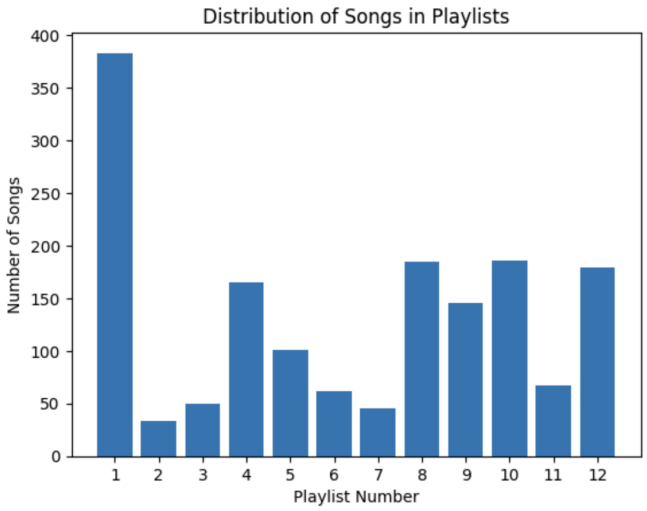
Literature Review

Recently, there has been significant research conducted on the applications of machine learning and reinforcement learning to generate personalized playlists. These approaches explore a range of models with varying degrees of success in real-world implementations. One such approach uses unsupervised K-means clustering to organize songs into groups based on similar moods (Shah 2020). They used the “Elbow method” to choose an optimal number of clusters which involves plotting the inertia of different choices of K and choosing a value where the inertia decreases and resembles an elbow shape. After choosing their K value, they used 24 different features to group similar songs to the same cluster. This approach is similar to ours in the sense that we will also be extracting song attributes with the goal of finding songs that have similar moods to create a cohesive playlist. However, their approach differs because it is K-means clustering to find similar songs, however we plan to use cosine similarity as a similar evaluation metric. We are currently exploring two main approaches, a search-based approach and a machine

learning approach. The search-based approach will be different from the implementation above as it would resemble a greedy algorithm searching for an optimal solution rather than clustering songs into groups. The machine learning approach would be more similar, as it extracts features from the songs to maximize similarity by predicting the most likely song to fit into the user's music taste and the songs already added to the playlist. Thus, our approach will be fairly orthogonal, yet not contradictory to the approach above, but we hope to use similar frameworks of extracting song attributes from our dataset and using them to rate how similar songs are.
An approach more similar to ours that does not constrain songs to genres but rather to meaningfully relate songs to each other.(Dargavel Smith 2023) This approach is more similar to ours since our goal is to create a playlist that is multidimensionally cohesive, meaning songs from different genres may be in the same playlist. Additionally, this approach found that the best scoring metric was using cosine similarity which was how we plan to evaluate the cohesiveness of our playlists. This paper also discusses the potential of future approaches such as Siamese networks and two dimensional convolution networks, both of which we are exploring as potential networks to most meaningfully capture our data.

Dataset

The dataset has 1,614 total song entries, and each song is part of at least one of the 12 playlists that we scraped from Spotify. The 12 playlists are of varying size, and thus songs are unevenly distributed across the playlists; however, this is intentional to account for the fact that playlists can vary widely in song length. The distribution of songs across the 12 playlists is pictured below:



We will be doing data pre-processing to maintain only relevant features of the dataset. We will be dropping all column names except “TrackName” (string), “Artist Name” (string), “Explicit” (bool) and “Popularity” (int). These will be the X_train features. There are several pre-processing steps needed to transform the X_train categories. First, all text data will need to be converted to numerical data using TF-IDF encoding. Next, the “Explicit” boolean will need to be converted to numerical 0’s and 1’s. All of these processed features will then need to be synthesized into a singular feature set for each song. We will be adding a column called “Playlist Number” that will include the playlist that each song is a part of, representing the y_train values. We will use a multi-label binarizer to convert the playlist labels into a binary matrix suitable for multi-label classification.

Baseline

The baseline for this project takes in a list of input song entries from a user, predicts which of the 12 playlist(s) the input songs most likely belong in, and returns a list of 15 songs to create a playlist based on the input. We implement this baseline using multi-label classification via the MultiOutputClassifier from sklearn. To do this multi-label classification, we first had to “binarize” the 12 playlists into binary matrix form using MultiLabelBinarizer(). We then combined all the features into a single feature set for each song entry, and split the data into train and test data. We then fit the MultiOutputClassifier to the X and y train data, and evaluated the

model's prediction performance using the classification_report from sklearn.metrics detailing the precision, recall, and F-1 scores. The evaluation of these metrics can be found in our Results section.

Main approach

The inputs to our algorithm are a list of songs (represented as strings) provided to the user. Various features of the songs (such as their title, artists, and album name) are vectorized using scikit-learn's "TfidfVectorizer" function. Then using sk-learn's "MultiOutputClassifier," we performed logistic regression and retrieved an array of playlists (a subset of the playlist in our database) whose songs were most similar to the inputted song in the input. We then randomly returned a list of 15 songs from the predicted most similar playlist to recommend to the user. For example, consider the input list of songs:

Song Name	Artist	Album
Fragile	Kygo, Labrinth	Fragile
Silence	Khalid	Silence
Winter	Khalid	American Teen
Sober Up	AJR, Rivers Cuomo	The Click
Dazed and Confused	Ruel	Dazed & Confused

All these parameters would be converted into vectors which themselves are part of a larger feature vector. After performing the described transformations and logistic regression, the most compatible playlists in our database of playlists were "Playlist 2" and "Playlist 7". We then randomly returned 15 songs from these playlists, generating the output:

“Here are songs recommended for your new playlist: ['redrum', 'In Camera', '100 Bad Days', 'Rainbow Cadillac', 'DANCING ELEPHANTS', 'Hefner', 'Mishaps Happening - Prins Thomas Edit', 'All Night - Cash Cash Remix; Radio Edit', 'In Camera', 'Weekends', 'Pyramids', 'Dark Red', 'Leave The Door Open', 'BabyWipe', "She's So Nice", 'Pyramids']”

Evaluation Metric

To evaluate the overall performance of the model, we can use the quantitative metrics of area under the ROC curve (AUC ROC) and the confusion matrix (true positives, false positives, false negatives, true negatives) which can be used to calculate precision, recall, and the F-1 score. All of these metrics are embedded in sklearn packages which we will use to calculate and evaluate our model. From a qualitative perspective, we can ensure that the recommended songs are not just repeats of the input songs.

Results & Analysis

In our project, the baseline model using multi-label classification showed a mix of performance across the various playlists and showed some playlists achieving satisfactory precision and recall while others seemed to lag significantly in comparison. The initial outcomes indicated a need for a more nuanced handling of features and perhaps a more robust modeling approach to handle the diversity in music genres and song representation within the dataset. Therefore, following the baseline a combined approach of search-based and machine learning strategies was implemented. By vectorizing song features via TF-IDF and applying logistic regression, the refined model could predict more relevant playlists that matched the characteristics of input songs. However, metrics like AUC ROC and analysis from the confusion matrix showed areas of weakness, specifically in distinguishing between similar playlists and managing songs with lower representation.

However, overall, the results align with our goals with this project in the time frame we were given, which is to just create a more effective and personalized music recommendation system, and our project does recommend songs from playlists that users might listen to. As an extension,

we can also identify patterns of misclassification and areas where our model does perform well to better tailor our system.

Error Analysis

In our error analysis, we conducted some experiments that focused on the ability of the system to correctly classify songs into a playlist. One experiment we did involved adjusting the thresholds for classification, which led to an increase in precision for some playlists as well as a decrease in recall. Therefore, this experiment showed us that there were tradeoffs between capturing relevant items and limiting false positives. Another experiment we did was just analyzing misclassifications, which revealed that certain playlists with similar musical themes were confused by our system. This suggested an overlap in song characteristics that the model was unable to distinctly classify.

From our experiments, we realized that there is a balance required in model tuning to optimize both precision and recall and were challenges associated in handling diverse and overlapping music categories.

Future Work

In the future, we could definitely expand our dataset, which we think will be able to significantly improve our performance since for the scope of this project, we just scraped the Spotify data of one member in our team. However, by incorporating a broader range of songs and playlists, including the more niche genres and less mainstream artists, our model could better understand and predict a wider array of user preferences.

We could also explore other, more advanced machine learning techniques like deep learning that could maybe capture the complexities of musical attributes and user preference better. For instance, neural networks could be used to learn more abstract representations of songs, potentially improving the distinction between similar but distinct genres or moods. We would also need to look at challenges relating to feature selection and model overfitting. We could

implement regularization strategies to include more specific attributes like mood, tempo, content of lyrics, etc. so that our model can differentiate between musical patterns dn user preferences.

Code

Here is the link to our Github:

https://github.com/abbiemaemoto/bopbuddy

Works Cited

Dargavel Smith, Robert. 2023, https://github.com/teticio/Deej-AI. Accessed 23 May 2024.

Shah, Veeral. “Building Mood-Based Spotify Playlists using K-Means Clustering.” *veeral does data*, 20 July 2020, https://veeraldoesdata.com/spotify-mood-playlists. Accessed 22 May 2024.