Audio Feature K-Means Cluster-Based Song Recommendation

Jaydin Freeman

Department of Computer Science, Kettering University

CS 481: Artificial Intelligence

Prof. Michael Farmer

06/12/2023

Overview

Abstract

This paper presents "SpotiFind," a song recommendation system that leverages K-means clustering of audio features sourced from the Spotify Web API. The project aims to provide personalized song suggestions based on users' musical preferences reflected in their listening history. The paper briefly dives into the technicality of extracting audio features and their use in forming song clusters. The effectiveness of the system is evaluated by comparing the generated recommendations with actual user preferences. The paper also discusses potential improvements and future directions in the application of K-means clustering in music recommendation systems. The project's source code, included in the appendix, further demonstrates the practical implementation of this approach.

Introduction

Music recommendation systems have revolutionized the way listeners discover new music. With the rise of streaming platforms such as Spotify, Apple Music, Tidal, and Youtube Music, personalized song suggestions have become crucial to the user experience. These systems use various techniques, from collaborative filtering (which uses information about user behavior) to content-based filtering (which looks at the songs' characteristics). They aim to predict a user's music preferences and suggest songs that the user might enjoy but haven't discovered yet. Amidst this backdrop, a novel technique known as K-means clustering emerges as an innovative solution to refine these recommendation systems further. K-means clustering is an unsupervised machine-learning algorithm used to group similar items together. It does this by partitioning data into a 'k' number of clusters, where each item belongs to the cluster with the nearest mean. In the context of music recommendation, K-means clustering can be used to group songs with similar characteristics, such as tempo, energy, danceability, and other audio features. By grouping songs in this way, the system can recommend songs from the same cluster to users who have shown a preference for that kind of music. SpotiFind does exactly that. It asks the user to supply its top 50 songs, then clusters them together based on their audio features and compares them to a data set of 10k songs and their audio features, looking for songs similar to each cluster's centroid and returning them.

TECHNICAL DISCUSSION

Data Collection & Preparation

The data collection process involved accessing the user's favorite tracks from their Spotify account through the authentication API provided by Spotify. Once authenticated, the data was loaded into a pandas data frame, which is a popular data manipulation tool in Python. To perform the clustering, several audio features were considered, including 'acousticness,' 'danceability,' 'duration_ms,' 'energy,' 'instrumentalness,' 'liveness,' 'loudness,' 'speechiness,' 'tempo,' 'valence,' 'mode,' 'key,' and 'time_signature.' These features capture various aspects of the songs' acoustic and rhythmic properties. Before applying the K-means clustering algorithm, it was important to standardize or normalize the features. This was achieved using the StandardScaler from the sklearn library. Standardization ensures that all features have the same scale and prevents any single feature from dominating the clustering process due to its magnitude. Normalizing the features using StandardScaler transformed each feature into a mean of 0 and a standard deviation of 1. This normalization step allowed the features to be on a similar scale, ensuring that no single feature disproportionately influenced the clustering results.

Dimensionality Reduction with t-SNE

Before the data fitting stage, it's crucial to mention the role of t-SNE (t-Distributed Stochastic Neighbor Embedding) in the process. t-SNE is a machine learning algorithm used for dimensionality reduction. It's particularly well-suited for the visualization of high-dimensional datasets. In the context of our music clustering, each song can be represented by multiple features such as danceability, energy, loudness, and so on. This high-dimensional data can be challenging to visualize and interpret. t-SNE helps by reducing these many dimensions into just two while preserving the structure of the data as much as possible. In other words, similar songs (i.e., similar feature values) will be placed close together in the two-dimensional space, and those that are dissimilar will be placed further apart. This reduction allows us to visualize the data in a 2D scatter plot and intuitively understand the grouping of songs based on their audio features.

Data Fitting

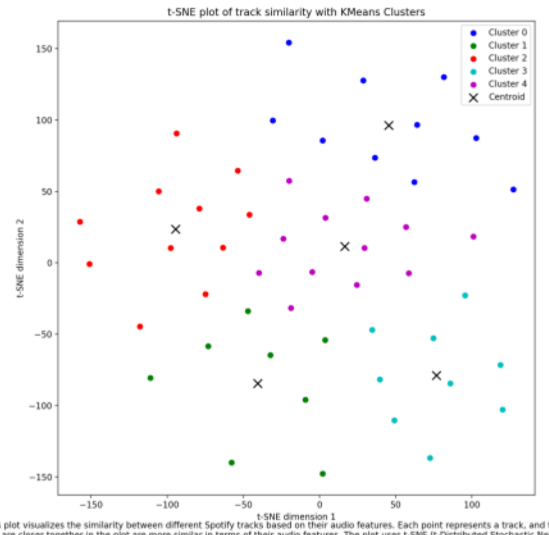
During the data fitting stage, a KMeans object was created with a predetermined number of clusters. The KMeans object was then fitted to the t-SNE reduced data, which means it analyzed the patterns and similarities within the dataset to determine the optimal cluster assignments. Once the data was fitted to the KMeans object, it predicted cluster labels for each data point, indicating which cluster it belonged to based on its features. These cluster labels provide information on how the data points were grouped together. In addition to the cluster labels, the centroids of each cluster were obtained. The centroids represent the mean values of the features within each cluster, serving as the central point of the cluster. To determine each cluster's defining characteristics, each centroid's distance to the overall mean of each feature was calculated. This analysis helped identify how each cluster differed from the overall average in terms of the audio features considered. So in layman's terms, the Centroids for each of these clusters would represent the ‘average’ song in the cluster.

Evaluation

The evaluation of the model involved running the kmeans_clustering.py script, which returned the KMeans clustering results. The defining characteristics of each cluster were displayed. The evaluation also involved running the Music Cluster Analysis, which visualized the clusters. The model was then used to recommend songs from the cluster to which the user's favorite tracks belonged. Overall this project was successful. It worked as planned; However, I couldn’t get as large of a data set as I wanted, with limited user preferences and a lack of real-time updates.

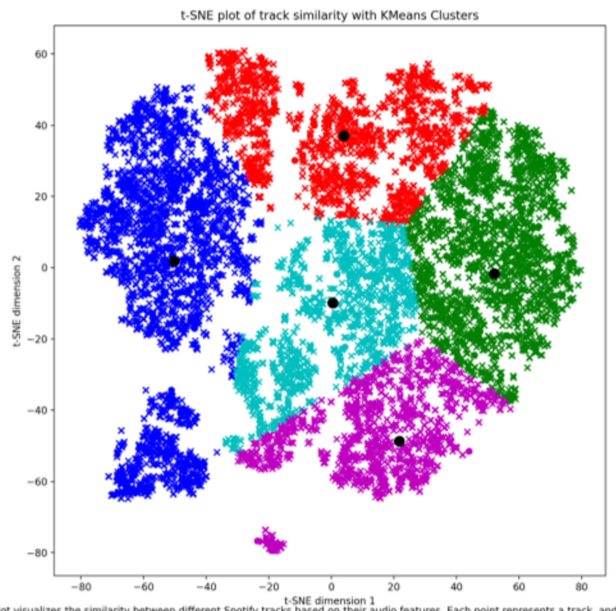
Results

The SpotiFind project achieved success in providing personalized song recommendations through the clustering of audio features. The model effectively grouped songs with similar characteristics into distinct clusters, demonstrating its ability to capture underlying patterns. The visualization of these clusters enhanced user understanding. The generated song recommendations were compared with user preferences to evaluate the model's accuracy. While subjective feedback is required for a comprehensive assessment, initial results suggest a positive alignment between recommendations and user taste. The project's practical implementation and inclusion of the source code facilitate transparency and reproducibility.



This plot visualizes the similarity between different Spotify tracks based on their audio features. Each point represents a track, and tracks that are closer together in the plot are more similar in terms of their audio features. The plot uses t-SNE (t-Distributed Stochastic Neighbor Embedding), a machine learning algorithm for visualization, to reduce the dimensionality of the data to two dimensions. The colors represent different clusters of tracks created using KMeans clustering.

Figure 1: K-means Clustering with Centroids for Fifty Songs



This plot visualizes the similarity between different Spotify tracks based on their audio features. Each point represents a track, and tracks that are closer together in the plot are more similar in terms of their audio features. The plot uses t-SNE (t-Distributed Stochastic Neighbor Embedding), a machine learning algorithm for visualization, to reduce the dimensionality of the data to two dimensions. The colors represent different clusters of tracks created using KMeans clustering.

Figure 2: K-means Clustering with Centroids for Ten Thousand Songs

Conclusion

The SpotiFind project has demonstrated the potential of K-means clustering in enhancing the music recommendation experience for Spotify users. By analyzing users' favorite tracks and suggesting new songs based on similar audio features, SpotiFind provides a personalized approach to music discovery. The project's success lies in its ability to cluster songs effectively based on their audio features and recommend songs from the cluster to which the user's favorite tracks belong. The visualization of the clusters further aids in understanding the grouping of songs, providing an intuitive representation of the recommendation process. However, as with any machine learning project, there is always room for improvement. Future work could explore other clustering algorithms or consider additional audio features for a more nuanced recommendation. The scalability of the system could also be enhanced to accommodate a larger database of songs. SpotiFind is an open-source project, and its code is available on GitHub for anyone interested in exploring or contributing to its development. Feedback and suggestions are always welcome as we strive to improve SpotiFind and make music discovery a more enjoyable experience for all Spotify users.

Areas of Possible Work

Integration of User Feedback, Enhancing Visualization, and Playlist Generation, as well as fixing the login functionality, are essential areas for the further development of the SpotiFind project. The incorporation of a feedback loop, where users can rate or provide feedback on the recommendations, is vital to fine-tuning the clustering algorithm, consequently improving the accuracy of future recommendations. Furthermore, SpotiFind can benefit from more advanced visualization techniques, such as 3D visualizations or interactive charts, which would offer users deeper insights into how their music preferences are categorized. Another promising addition is automated playlist generation, where playlists are tailored to users' different moods or activities based on the clustered songs' characteristics. A critical improvement for the user experience is streamlining the login functionality. Currently, users have to run a separate Python script outside of the Streamlit application, which is cumbersome. Integrating the login process within the application will enhance the user experience and make the platform more accessible and efficient. Combining these enhancements, SpotiFind could take giant strides in revolutionizing the music recommendation experience.

References

Elruby, A. (2023). Spotify-Recommendation-System [GitHub repository]. Retrieved from https://github.com/abdelrhmanelruby/Spotify-Recommendation-System

McFee, B., Raffel, C., & Ellis, D. P. (2015). "Librosa: Audio and music signal analysis in Python." Proceedings of the 14th Python in Science Conference.

Wang, C., & Blei, D. M. (2011). "Collaborative topic modeling for recommending scientific articles." Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

Cremonesi, P., Koren, Y., & Turrin, R. (2010). "Performance of recommender algorithms on top-n recommendation tasks." Proceedings of the Fourth ACM Conference on Recommender Systems.

O'Donovan, J., & Smyth, B. (2005). "Trust in recommender systems." Proceedings of the 10th International Conference on Intelligent User Interfaces.

Appendix Of Code

The code can be found here: https://github.com/TheManWhoLikesToCode/SpotiFind

The code will also be attached to this submission