Categorize Music with PCA and K-Means Clustering

Introduction

The Spotify feature dataset certainly helps us to get desirable songs according to one's choices. Clustering and dimension reduction help us to identify specific song lists to suggest clients based on the features. It also gives you access to information that is not available on the app, such as danceability', 'valence', 'tempo', 'liveness', 'speechiness'.

Data Set

Spotify uses a series of different features to classify the tracks. The descriptions of the features are:

- **Acousticness**: [0–1] Confidence measure of whether the track is acoustic.
- **Danceability**: [0–1] Describes how suitable a track is for dancing based on musical attributes including tempo, rhythm, stability, beat strength, and overall regularity.
- **Energy**: [0–1] Perceptual measure of intensity and activity. Energetic tracks feel fast, loud, and noisy (e.g. death metal: high energy, Bach prelude: low energy).
- **Instrumentalness**: [0–1] Predicts whether a track contains no vocals (values above 0.5 represent instrumental tracks whereas rap songs would have a score close to 0).
- **Liveness**: [0–1] Detects the presence of an audience in the recording.
- Loudness: [-60–0 dB] The average volume across an entire track.
- **Speechiness**: [0–1] Detects the presence of spoken words in a track (values above 0.66 describe tracks that are probably made entirely of spoken words, 0.33–0.66 describe tracks that may contain both music and speech, and values below 0.33 most likely represent music and other non-speech-like tracks).
- **Valence**: [0–1] Describes the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).
- **Tempo**: [0–300 BPM] The speed or pace of a given piece, as derived from the estimated average beat duration.

Objectives

In this project the following works have been highlighted:

1. **Data Collection:** With "Spotipy" library we can get full access to all of the music data provided by the Spotify platform. Spotipy supports all of the features of the Spotify Web API including access to all endpoints, and support for user authorization. Creating an app in Spotify developer, it allows me to collect data on audio features.

2. **Cleaning and Indexing:** To get the appropriate information we must drop down the rows having null data. Then we need to create a unique index for each song by joining "track name" and "track id" columns. We also need to pick useful features by which we can differentiate one from another. For the purpose of that, we are dropping several columns.

3. **Standardization:** We need to standardize the data before clustering is important because clustering algorithms are sensitive to the scale of the variables being used. If the variables have different scales, the clustering algorithm may give more weight to variables with larger scales, which can skew the results.

4. **Dimension Reduction:** Principal Component Analysis (PCA) is useful as it can help to identify the most important features or variables that contribute the most to the overall variability in the data. By reducing the number of features, it becomes easier to visualize and interpret the data and identify patterns and relationships between songs. It is a good rule of thumb to consider an 80% variance. If we plot our cumulative sum of variance for features we can see that if we select 5 components it gives us enough variance for the whole dataset.

5. **Optimal Cluster Number:** In order to implement k-means clustering, we must select a number of clusters, k, which distinctly splits the data. The inertia_ function in Python is an attribute of the KMeans clustering algorithm. This function returns a single value that represents the sum of squared distances for all clusters. The lower the value of inertia_, the better the clustering algorithm has performed, indicating that the data points are tightly clustered around their respective centroids. We are determining how many clusters would be optimal to get a minimum within-cluster sum of squares (WCSS). Our optimal number of clusters is 3. To evaluate the quality of clustering silhouette score is a metric often used. It measures how well each data point fits into its assigned cluster based on both its similarity to other data points within the same cluster and its dissimilarity to data points in other clusters. The silhouette score ranges from -1 to 1, with a score closer to 1 indicating that the data point is well-matched to its assigned cluster. Through the elbow method we find out, the best score comes for 3 clusters which also match our desired number of clusters. Our Hopkins Score is also higher than 90% which is the degree to which a dataset can be considered as having clusters that are distinct from each other.

6. **Visualization:** The InterclusterDistance visualizer helps to interpret the clustered data situation by showing the distances between the centroids of different clusters.By analyzing the distances between the centroids, we can determine whether the clusters are well-separated or overlapping. If the distances between the centroids are large, it suggests that the clusters are well-separated and distinct from each other. On the other hand, if the distances between the centroids are small,

it suggests that the clusters may be overlapping or poorly separated. Here we can see the 3 centroids are well separated from each other.

7. **Analyzing the Cluster:** By now as we have an optimal number of clusters and PCA, we can go forward with analyzing our songs. To get to know which songs belong to which cluster we have to label each song according to its cluster number. Then we will extract the information about each cluster to know better about each song's features. We can make some bar charts, plotting a particular feature for each track in a given cluster in order to see if this feature is, indeed, characteristically high with respect to the average feature value in the entire dataset.

   In order to get the real value of features now we have to take the clean data frame in account as the "Cluster_pca_kmeans" dataset is standardized. Also, we have to add the "cluster" column from the "Cluster_pca_kmeans" dataset. As their index isn't the same so we have to set it to the same index first and then copy the column from the "Cluster_pca_kmeans" dataset and paste it to the “sample_input_df” dataset. From the bar charts, we can observe some features for each cluster. For example, cluster 0 has high acoustics, low instrumentalness.

In project I have added comments for each part of the code analyzing the output for better understanding.

Conclusion

Through this project we can see how clustering can save us a lot of time in selecting a song of our choice. Now based on our choice of features, we can choose our song faster!