The Spotify Music Dataset: clustering similar artists

# Problem Identification

### Problem Statement

Have you ever heard an artist you liked and wanted to hear more artists that have a similar sound? You don’t want artists that are just in the same genre, you want artists that sound the same.

### Goal

Using a data driven analysis and machine learning modeling techniques, we will hopefully be able to group artists by audio features and then recommend an artist that has a similar sound. I will combine

# Data

The data comes from [Kaggle](https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks) and contains five csv files. These files contain data for more than 175,000 songs. The data is then grouped by artist, genre, and year in separate files. The dataset is updated monthly and was last updated at the end of January 2021.

## Music Track Data

The data.csv file has different audio features for each track. This is what we will be using for our clustering. The features within this file are:

* Acousticness
* Danceability
* Energy
* Duration in MS
* Instrumentalness
* Valence
* Popularity
* Tempo
* Liveness
* Loudness
* Speechiness
* Year
* Mode
* Explicit
* Key
* Artists
* Release Date
* Name

All of these features are numerical features other than Key, Artists, Release Date, and Name.

## Artists Data

The file called data\_by\_artist.csv has all of the same features, except it is grouped by artist. Each row in the file contains a different artist and each column is the audio feature from the data file. There is an additional field called count. This field contains the number of songs in the main data file.

Since we are most interested in clustering the data by artist, we will primarily use the data by artist file.

# Data Wrangling and EDA

This dataset was very clean. I started out by loading the artists csv file into a dataframe called **df**. After loading the data, I looked for null values and did not find any, which was a great sign. I then looked for duplicate rows to make sure that there were no artists listed twice, and there were none.

Looking at the distribution of the data in the histograms for each feature, nothing stood out as being concerning.

Chart, histogram

Description automatically generatedChart, histogram, box and whisker chart

Description automatically generated

The pairplots and the boxplots also looked good, maybe some outliers but nothing that was concerning. The only thing that stood out was the duration feature, but since this is the total duration for songs, and some artists may have many more songs than other artists, it didn’t seem to concerning. Chart, box and whisker chart

Description automatically generatedEngineering drawing, calendar

Description automatically generated

# Preprocessing and Modeling

## Preprocessing

I will be using KMeans and Agglomerative clustering to do the modeling. I started by creating a new dataframe called df2. This dataframe contained the artist data with the mode, count, and artists features removed. I then stored the values from df2 into X and ran the Standard Scaler on the data.

I created an elbow plot to see what would be the best number of clusters between 1 and 100. The elbow plot showed that the number of clusters should be around 20-40. Since I wanted to recommend similar artists, I chose 25 so that it was closer to the steeper part of the elbow but would allow a good amount of clusters for recommending artists. Chart, histogram

Description automatically generated

## Modeling

Before modeling I needed to identify test cases to test that the clustering was working. I chose artists that would most likely be clustered together and definitely not clustered with others.

Below are the artists I selected:

* 50 – Cent
* Eminem
* Taylor Swift
* Backstreet Boys
* NSync
* Bee Gees
* The Beatles

I started out with the KMeans clustering algorithm. I fit and predicted the data of df2 and created a new feature called cluster that would contain the cluster the artist was assigned to. After clustering I checked the data to see what clusters each of my test cases were in.

|  |  |
| --- | --- |
| Artist | KMeans Cluster |
| 50 – Cent | 23 |
| Eminem | 23 |
| Taylor Swift | 13 |
| Backstreet Boys | 13 |
| NSync | 0 |
| Bee Gees | 6 |
| The Beatles | 16 |

Some of the artists were grouped as I expected, but some weren’t. After thinking about this I thought that maybe the artists that I thought of on my own as test cases didn’t actually have similar musical sound profiles. So I plotted the data to visualize the clusters. I used KMeans and Agglomerative Clustering. I visualized both models pre and post PCA to see how the visualizations differed.

Chart, scatter chart

Description automatically generated

Chart, scatter chart

Description automatically generated

Chart, scatter chart

Description automatically generated

Looking at the scatter plots the post PCA visualizations look significantly better. This is due to reducing the features down to just two. The comparison between Agglomerative and KMeans is very similar. There are some clusters in the Agglomerative scatter plot that are broken out into smaller clusters of data when compared to the KMeans visualization, but the KMeans look more defined.

# Final Visualization

I loaded the KMeans PCA data into [Tableau](https://public.tableau.com/profile/ryan5903#!/vizhome/MusicArtistClusters/Sheet2?publish=yes) and created a cluster plot in order to be able to visualize the artists when hovering over each point.

I focused the plot on the majority of the data so that it was more visually appealing. Below is a screenshot, but please click on the link above for a more interactive experience. A picture containing scatter chart

Description automatically generated