Spotify Data Analysis

Viet Anh Duong - anhd@sfu.ca
Nguyen Dang Khoa Pham - ndpham@sfu.ca

1. Problem Statement

In this project, we wanted to explore multiple problems in the music streaming industry, Spotify, to be more specific in this case. We find that songs' genres and their popularities would be quite intriguing to be investigated since it probably would provide us with some interesting insights about why a specific song and genre became popular in a particular year. In particular, we tackled the following questions for further analysis:

1. How accurate can we predict a song's popularity based on its audio features?
2. How accurate can we predict a song's genre based on its audio features?
3. Which genre has the highest popularity in the explored time period?
4. Can we recommend a song based on a set of audio features?

We approached the tasks by performing multiple data cleaning procedures and statistical analysis to examine the correlation between a song's audio features or genre, and its popularity. For the classification tasks, we applied multiple machine learning models for predictions. Meanwhile, for the most popular genre, we used an ANOVA test among groups with most popular genres. Notice that all these mentioned approaches were performed after refining the dataset with some initial preliminary analyses.

2. Methodology

2.1. Data Extraction, Cleaning, and Transformation

We initially acquired two datasets from two online competitions:

- challenger_set.json: This dataset consists of 1,000,000 playlists including playlist titles and track titles. We then extract the unique song ids from these playlists and feed them to Spotify Developer API to get the metadata like artist, genre, release_date as well as the audio features of the song.

- SpotifyFeatures.csv: This dataset consists of 232,725 tracks that already have the metadata and audio features. However, this dataset is missing the "release_date" field, which is necessary for analysis. Therefore, we feed the song ids from this dataset to Spotify Developer API to get the "release_date" field and merge it back to the initial dataset.

We then merged these two datasets on unique song ids since there is a noticeable amount of repeated song id entries, at 19,480. Some minor data cleaning and transformation were necessary as well.

- The duration of a song is in milliseconds, so we convert it to seconds.

- Some of the entries have invalid date formats, so we have to drop those.

- All of the songs released range from 1899 up to 2020 but the ones in the 2020s or less than the 1960s are very minimal compared to the whole dataframe so we dropped those entries, limiting the range to [1960, 2019].

- Some of the audio features like “key”, “mode” and “time_signature” are categorical and they have a combination of integer and string values. For example, the “mode” has two categories Minor (=0) and Major (=1) but the data column has a mixed combination of these 4 values, so we have to convert them to integers accordingly based on how Spotify defines them. However, we have to drop the “time_signature” column as there is no defined way to convert them to integers.

2.2. Data Analysis Techniques

2.2.1. Analysis of song popularity based on its audio features

We look at the audio features when investigating song popularity. A correlation analysis is used to filter out audio features which are not related to the dependent variable popularity because they are insignificant to the analysis later on and it might be heavy for computation since there are various audio features.

We then use multiple regression models to predict the songs’ popularity. The models we used are KNeighborsRegressor, RandomForestRegressor, GradientBoostingRegressor and LinearRegression(). After this, we output the R2 score of each model.

2.2.2. Analysis of a song genre based on its audio features

We first investigated the number of genres in the dataset and quickly realized that 1369 genres seem to be overwhelmed for further analysis, hence we decided to trim it down and filter for just significant genres. We initially observed that there are some redundant genres such as “winconsin indie” - which is just “indie”, or “indian edm” while “edm” is just enough, we suspect that the amount of songs with these repeated genres is insignificant and not worth converting into the appropriate genres for analysis, hence we decided to remove these rows to reduce the redundancy.

Next, we also filtered out the outliers by identifying the genres which have the amount of songs lower than 500. Our rationale is that genres that have less than 500 songs are insignificant in the dataset of roughly 250,000 songs.

After performing all of these mentioned fine-tuning, we archived a dataset of 30 unique genres of significantly large enough size of songs.

We then performed multiple machine learning models on these unique genres to see how accurately the models can predict the songs' genres. The models we were using are KNeighborClassifier, RandomForestClassifier, GaussianNB, and Neural Network (tensorflow.keras.Sequential). We then output the exact match score (EM) and F-score (F1) of each model.

2.2.3. Analysis of song popularity for different genres using ANOVA

We applied the same genre splitting process like when analyzing song genre based on its audio features, but more work is needed before doing the ANOVA test. From 30 genres, we filtered them again to 10 by checking which genre has more than 9000 songs since this ensures that the genre has enough data points that spread out continuously over the years to be used for the Central Limit Theorem (CLT). We used CLT by grouping popularity by each year and average it here to make the genres more normally distributed. We filtered the data by year range [1984, 2019] since this is the only range where all filtered genres have continuous data points. After the normality check, we have 7 genres ready for the ANOVA test.

2.2.4. Analysis of song recommendation based on song audio features

Initially, we observed that there exist rows with duplicate 'id' (different 'name' and 'genre', but same 'id'). By closely investigating these cases, we believe that the duplicated 'id' was pointing to different variations of an original song, the 'popularity' scores of these duplications are also roughly similar, hence, we decided to remove all of the redundancies and only kept the original songs with unique ids. We then normalized all of the numeric columns for values from 0 to 1 for consistency in Manhattan's calculation.

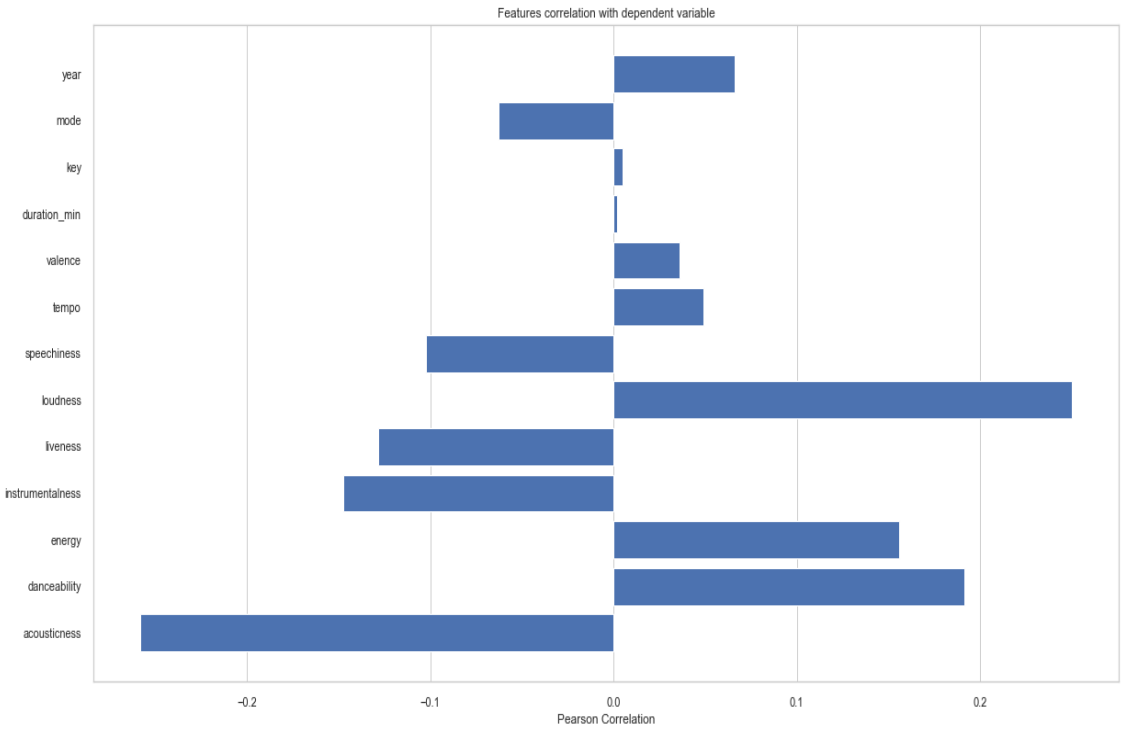
After the data preprocessing steps, we utilized the KMeans model for the song recommendation system given a particular song. We customized the k number of centroids n_clusters=25 and we believe that it is sufficient enough for our model. Our rationale for using KMeans is that it is one of the fastest clustering algorithms available with an average complexity of O(knT) with k being the number of centroids, n being the number of samples, and T being the number of iterations.

We finally trained our model and retrieved a list of recommended songs based on a given song by implementing the function

get_recommendations(). The number of recommendations can also be customized by utilizing the parameter “n_top”.

3. Results

3.1. Song popularity based on its audio features



Based on the feature correlation visualizer, we picked out the important features as being >0.05.

The evaluation scores are presented in the table below:

Model	R2 score
KNN	0.973
Random Forest	0.595
Gradient Boosting	0.538
Linear Regression	0.355

The KNN regressor performed significantly better than any other model being examined. We believe that this behavior occurs because we filtered the data quite clean and might result in overfitting. However, such an assumption needs further investigation on the data.

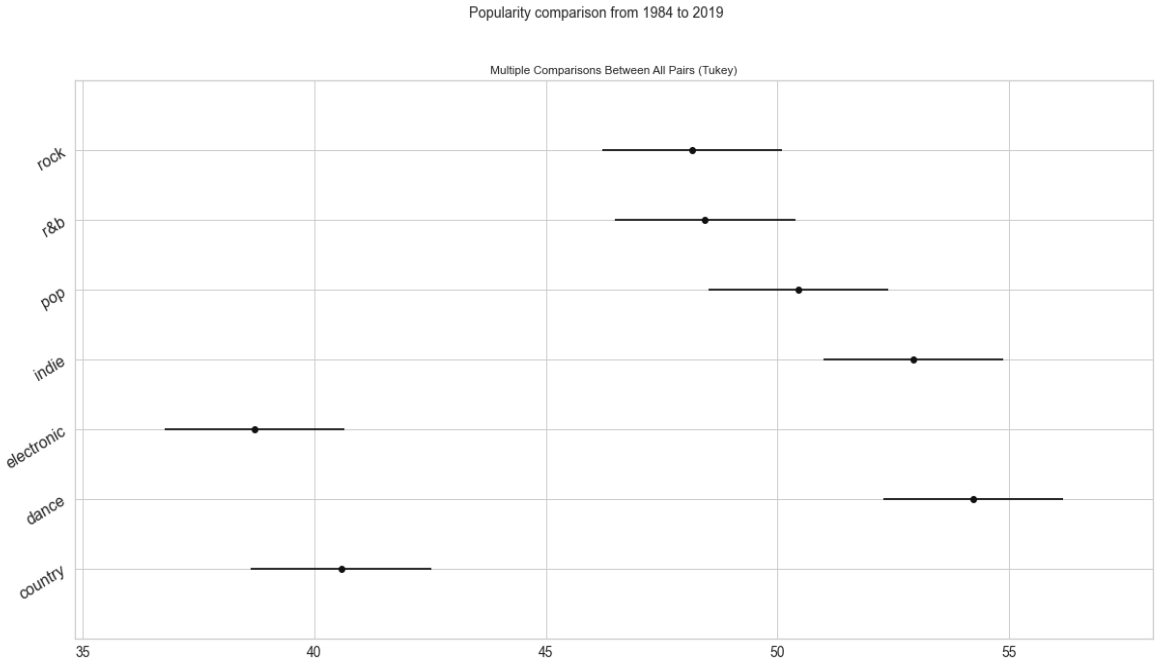
3.2. Song genre based on its audio features

Based on the mentioned models above, we retrieved the evaluation scores for each model:

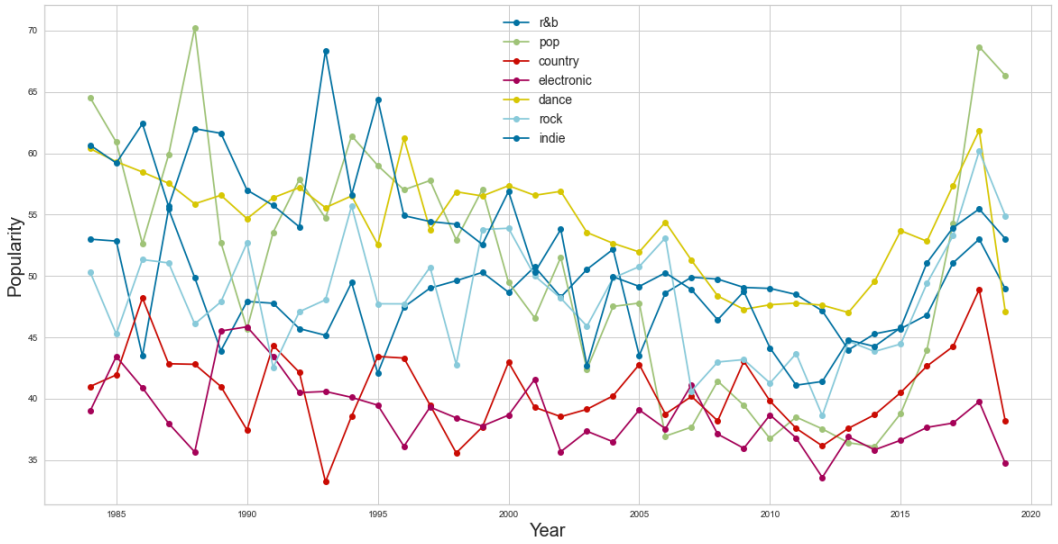
Model	EM	F1
KNN	0.695	0.691
Random Forest	0.376	0.343
Naive Bayes	0.291	0.261
Neural Network	0.296	N/A

Notice that the k-nearest neighbors model performed the best and is expected to do so for such multiclass classification.

3.3. Song popularity for different genres using ANOVA



With p-value = 2.31019954e-09, we can conclude that there is a statistical difference in the yearly average popularity among all genres and we can see that “dance” has been the most popular genre throughout this time period.



However, when we are only looking at the trend, then pop turns out to be the most popular in recent years.

3.4. Song recommendation based on song audio features

Based on the implemented song recommendation system, we made a sample for better visualization for the song “Tommy’s Party” and took the output as a list of 10 recommendations:

Observe that all of the output has a similar popularity score, hence, we suspect our model weighted popularity higher than other audio features.

```
get_recommendations('Tommy\'s Party', 10)
```

	name	artist	popularity	danceability	loudness	speechiness	acousticness	year
137799	Tommy's Party	Peach Pit	0.62	0.571862	0.804436	0.032161	0.12249	0.983051

name	artist	popularity	danceability	loudness	speechiness	acousticness	year
4 Seasons	Rex Orange County	0.660	0.511134	0.817802	0.040538	0.217871	0.966102
Asylum	Chloe Foy	0.590	0.441296	0.788074	0.031334	0.122490	0.983051
we fell in love in october	girl in red	0.670	0.572874	0.740336	0.028956	0.113454	0.983051
Cold Showers	Chelsea Cutler	0.610	0.631579	0.846589	0.033195	0.102410	0.983051
John My Beloved	VÉRITÉ	0.575	0.398785	0.781987	0.047570	0.205823	0.966102
Spirit Lead Me - Live	Influence Music	0.580	0.517206	0.802036	0.031127	0.222892	0.983051
When The World Was At War We Kept Dancing	Lana Del Rey	0.610	0.623482	0.827780	0.035471	0.097390	0.966102
Poetry	Wrabel	0.620	0.581984	0.854716	0.032472	0.020884	0.966102
Wildfire	Seafret	0.620	0.594130	0.817850	0.034023	0.253012	0.949153
Wish You Were Here	Avenged Sevenfold	0.545	0.578947	0.808986	0.030817	0.001777	0.966102

4. Limitations

We could have extracted more data for better precision but the bottleneck of the data collection process is the Spotify API. We tried to optimize the API calling function to process multiple song ids at a time resulting in much fewer requests sent to Spotify and much better performance. However, the process of collecting data from the API is still very time-consuming.

Our genre splitter function is not working properly in different cases. It is not capable of classifying “hip-hop”, “hip hop”, “american hip-hop”, “american hip hop” as the same genre. This is important because misclassification in genres can lead to different results of the analysis later on.

When examining the dataset for ANOVA test, we observed that there exist genres that have the data not normally distributed. We suspect that it occurs probably because our dataset is not sufficiently large enough and the entries are also insufficiently randomized to be representative for the whole population.

Another limitation when exploring the task of classifying songs’ genres is when utilizing the Neural Network approach, we suspect that it would outperform the other models, however, the output surprisingly behaves in a completely contrary way. We believe that it happened because we did not customize the hyperparameters of the hidden layers optimally, which resulted in significantly poor performance.

5. Future Work

If we have more time, we could alter the Python scripts to search for the song ids based on year or come up with different ways to generate our own song ids list instead of extracting these ids from other datasets. With a much bigger number of unique song ids, we could evaluate our models better. However, that requires a significant amount of time for the scripts to scrape data.

Introduce a better genre splitter function to ensure our results are more reliable. This could be a Natural Language Processing task to classify different variations of the same genre.

We plan to fine-tune the hyperparameters of the Neural Network approach to achieve the most optimal model. We believe that by doing so we would achieve a significant increase in the performance of the model.

6. Project Experience

Viet Anh Duong

- Structured and organized the ideas and tasks to do
- Helped in implementing the data scraping script
- Contributed to implementing and testing the experiments

- Analyzed in-depth for the experimental results for meaningful interpretations
- Implemented Neural Network approach in the genre classification task

Nguyen Dang Khoa Pham

- Formed ideas and defined problem domains for the project
- Wrote Python scripts to scrape data from Spotify API
- Learnt to use Neighborhood Collaborative Filtering and calculate Manhattan Distance to write a song recommendation algorithm
- Utilized matplotlib Python library to visualize and plot data
- Applied machine learning concepts using scikit-learn to tackle various classification/regression problems

7. References

https://towardsdatascience.com/predicting-popularity-on-spotify-when-data-needs-culture-more-than-culture-needs-data-2ed3661f75f1

https://www.kdnuggets.com/2021/04/song-popular-analyzing-top-songs-spotify.html

https://towardsdatascience.com/visualizing-spotify-songs-with-python-an-exploratory-data-analysis-fc3fae3c2c09

https://www.kaggle.com/danielsheen/spotify-song-genre-classification/notebook

https://github.com/tgel0/spotify-data