Analysis on Trends and Features in Spotify Tracks

Kevin Louie, Terence Tang

Computational Data Science

Summer 2024

Introduction

We will be using the Spotify API to work with song data. A song track contains many features including metadata-related ones as well as audio features. With the goal of curating high-ranking tracks on well-known song charts, producers would benefit from knowing which audio features correlate the most with popularity. Additionally, knowing how well collaborations between multiple artists perform compared to solo tracks in terms of popularity would be beneficial. We will be performing statistical tests and training machine learning models to tackle these topics.

Getting Data

We collected track and audio feature data from Spotify using Spotipy, a Python library designed for retrieving data from Spotify's Web API. In total, we acquired 1000 random tracks per year from the search endpoint, spanning from 2000 to 2023, and retrieved their audio features from the audio-features endpoint from Spotify's API. For each track, we collected 6 features that uniquely identify each track, and 12 audio features that provide insights into various aspects of the track's sound profile. These features include:

- `id`: The unique Spotify ID for each track.
- `name`: The title of the track.
- `popularity`: The numerical score representing the track's popularity, ranging from 0 to 100, with higher scores indicating greater popularity.
- `artists`: The list of artists who performed or contributed to the track.
- `release_date`: The date when the track (album) was released.
- `duration`: The length of the track in milliseconds.
- `acousticness`: The confidence measure of whether the track is acoustic, with values ranging from 0.0 (not acoustic) to 1.0 (highly acoustic).
- `danceability`: The suitability of the track for dancing, with values ranging from 0.0 (not danceable) to 1.0 (highly danceable).
- `energy`: The intensity and activity level of the track, with values ranging from 0.0 (low energy) to 1.0 (high energy).
- `instrumentalness`: The likelihood that the track contains no vocals, with values ranging from 0.0 (not instrumental) to 1.0 (with no vocal content).
- `key`: The key in which the track is played, represented as an integer from 0 to 11, corresponding to musical keys from C to B. If no key was detected, the value is -1.
- `liveness`: The measure of audience presence in the recording, with values ranging from 0.0 (not live) to 1.0 (highly live).
- `loudness`: The overall volume of the track in decibels (dB), with typical values ranging from -60 (very quiet) to 0 (very loud).
- `mode`: The modality of the track. Major is represented by 1 and minor is 0.
- `speechiness`: The presence of spoken words in the track, with values ranging from 0.0 (not speech-like) to 1.0 (highly speech-like).
- `tempo`: The speed of the track in beats per minute (BPM).

- `time_signature`: The number of beats in a measure. It ranges from 3 to 7, indicating time signatures of “3/4” to “7/4”.
- `valence`: The musical positiveness or happiness of the track, with values ranging from 0.0 (sad, depressed) to 1.0 (happy, cheerful).

Problem 1: Are collaborative tracks more/less popular than solo tracks on average?

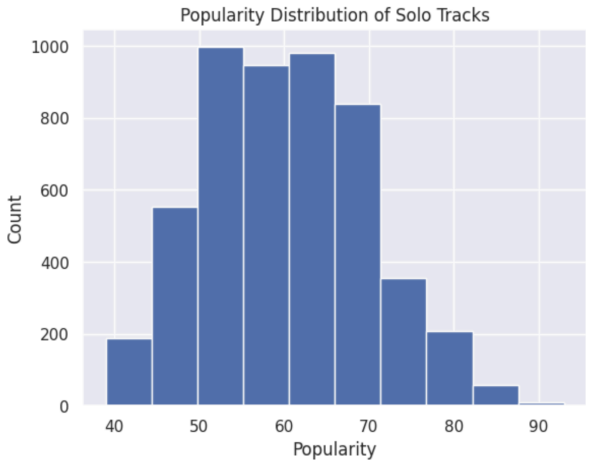
We want to perform a T-test with the following hypotheses:
Null hypothesis (H_0): Collaborative tracks are just as popular as solo tracks on average.
Alternative hypothesis (H_a): Collaborative tracks are more/less popular than solo tracks on average.

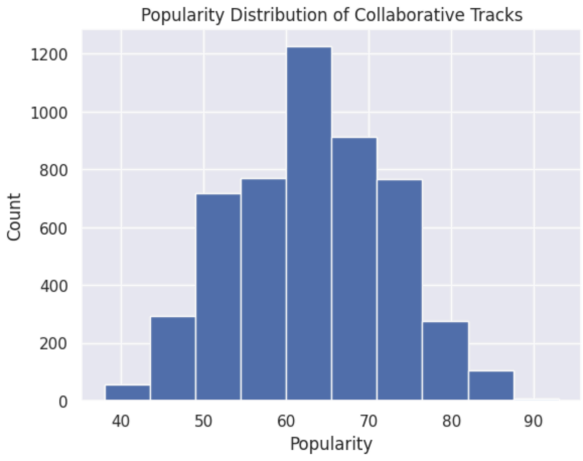
Cleaning Data

Firstly, we want to split all the tracks based on artist count. If there is only one artist, we consider it a solo track. Otherwise, it is a collaborative track. Since we are only interested in the popularity and the artists of a track, we remove all other audio features. Lastly, because there are inherently more collaborations than solos, we re-balance the data by sampling n random solo tracks, where n is the amount of collaborative tracks. Then, we use that as our new solos data frame.

T-Test Assumptions

Next, we want to ensure that the popularities of both groups (solos and collaborations) are normal and have equal variance. Both groups fail the normality test with $p \approx$ 6.28e-24 and $p \approx$ 6.80e-22 respectively. Judging by the plots below, however, the Central Limit Theorem tells us that the popularities look close enough to normal since $n \geq 40$. Additionally, performing Levene’s test on the two groups results in $p \approx 0.41$, so they have equal variance.





Perform the T-Test

Since all the test assumptions have been satisfied, we can perform the T-test, which results in $p \approx$ 9.31e-49. Thus, we reject the null hypothesis and conclude that there is a difference in the average popularity between the two groups. Comparing the means of the two groups, we get $\mu \approx 62.93$ for collaborations and $\mu \approx 60.16$ for solo tracks. Therefore, the popularity levels of collaborations are slightly higher than that of solo tracks on average.

Problem 2: How have the number of collaborations changed from the year 2000 to 2023?

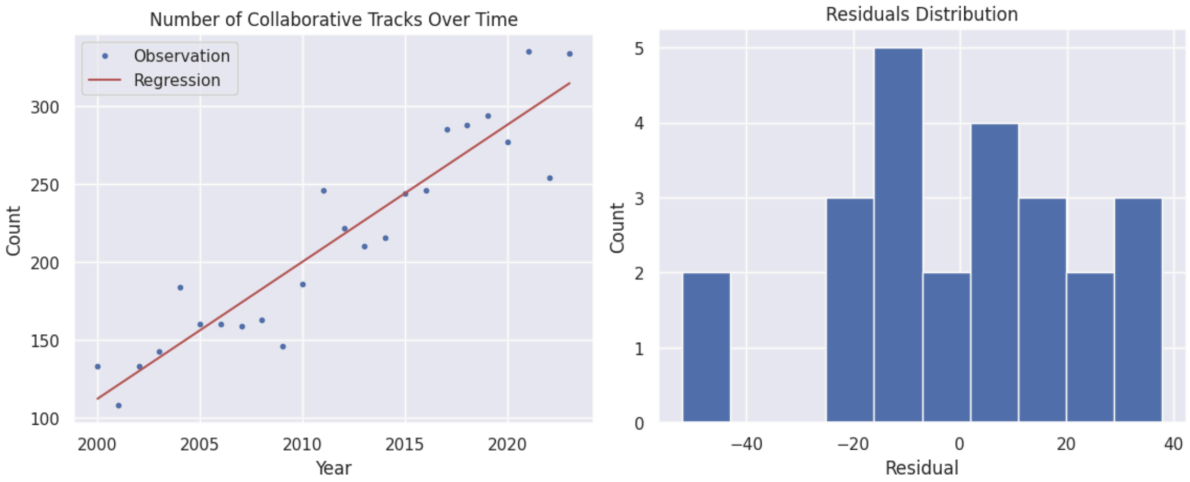
Cleaning Data

Re-using the same data from above, we now want the `release_date` column in a consistent format. It mostly consists of dates in the format YYYY-MM-DD but there are often just years given (YYYY). We split both data frames by its `release_date` format (date vs. year), and convert all the dates into their years. After joining back the split data frames, we group the collaborations by year and aggregate by counting the rows, which gives us the amount of collaborations per year as a data frame.

Linear Regression

Null hypothesis (H_0): The amount of collaborations does not depend linearly on its release year.
Alternative hypothesis (H_a): There is a linear relationship between the amount of collaborations and the release year.

We fit a regression line through the data as seen below, but we want to see how accurate it is using the Ordinary Least Squares (OLS) test.



The residuals data ends up passing the normality test with $p \approx 0.79$, so all the assumptions are satisfied. OLS gives us $p \approx$ 1.96e-11, and we reject the null hypothesis and say there is a linear relationship between the amount of collaborations and the release year. Since the regression

line has a positive slope of m = 8.80, we conclude that the amount of collaborations has increased from the year 2000 to 2023.

Problem 3: Which audio feature is more likely to impact tracks' popularity?

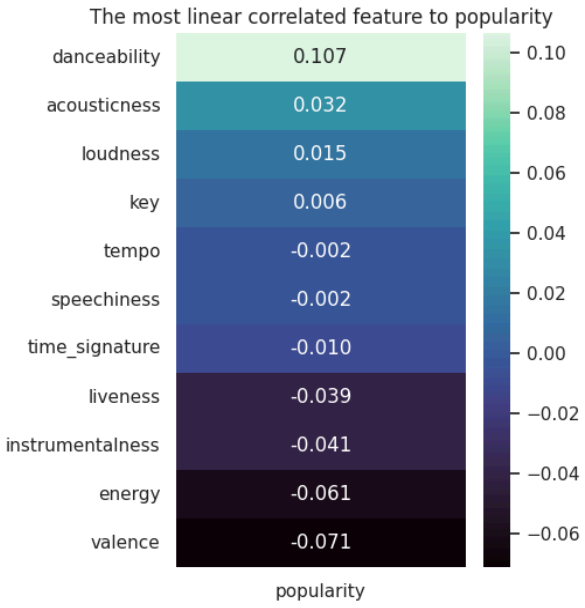
Cleaning Data

Since we are only interested in the audio features and popularity of each track, we first removed features like `id`, `name`, `artists`, `release_date`, and `duration`. The `mode` attribute only has 2 values, and serves similar purposes to `key` (but with a wider range), so we removed that. We also removed tracks with missing values, duplicates, and outliers that could affect the analysis. Specifically, we focused on finding tracks with invalid tempo and time signature data, like those with 0 BPM or time signatures outside the range of 3 to 7, as defined by Spotify's API.

Analysis by Statistics

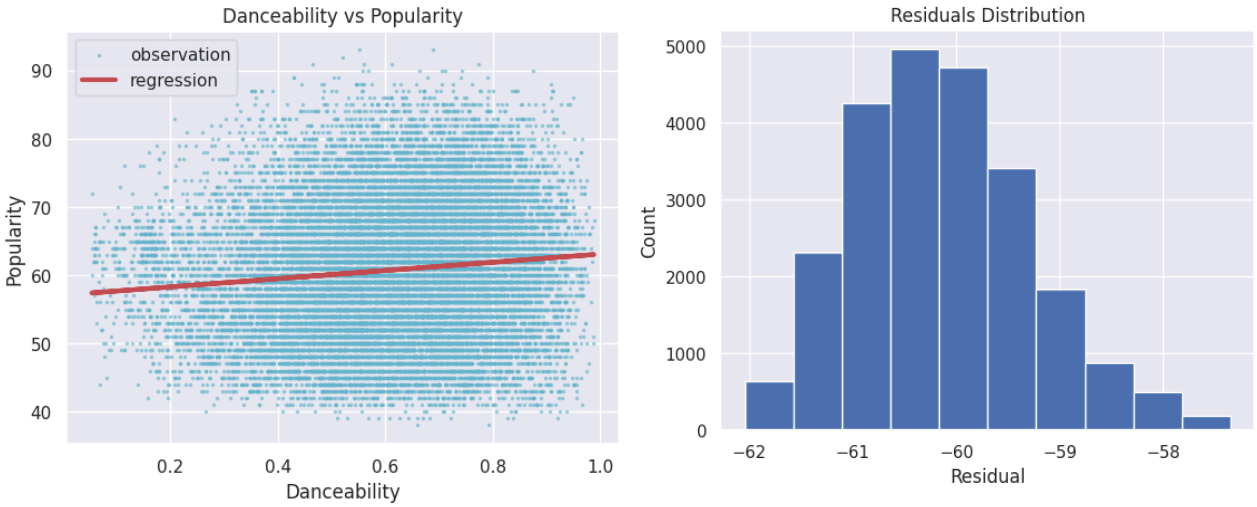
To analyze the impact of audio features on track popularity using statistics, we redefine the question to focus on the correlation between each feature and popularity, so the problem we focus on is now "which audio feature has the biggest linear relationship with popularity?"

We first look at all correlations between all features using Seaborn's heatmap, and specifically look at the column that compares the correlation between each audio feature and track popularity. We sorted the correlation in descending order to better identify one feature with the highest correlation with popularity:



By the heatmap, `danceability` seems to have the highest correlation with popularity. We now want to determine if it actually has a linear relationship with track popularity by doing the Ordinary Least Squares (OLS) test. We define the null and alternative hypotheses as follows:

Null hypothesis (H_0): Popularity does not depend linearly on danceability.
Alternative hypothesis (H_a): There is a linear relationship between danceability and popularity.



The residuals distribution is close to normal, and by the central limit theorem, we could assume normality since we have enough data points ($n \geq 40$). By the OLS test, we have $p \approx$ 7.76e-61, which is less than 0.05. Therefore, we can conclude that there is a linear relationship between danceability and popularity. By plotting the red regression line to the scatterplot, we can see the linear regression is positive too (with slope $\approx$ 6.02). Therefore, by using statistical tests, we conclude danceability is most likely to impact the track's popularity.

Analysis by Machine Learning

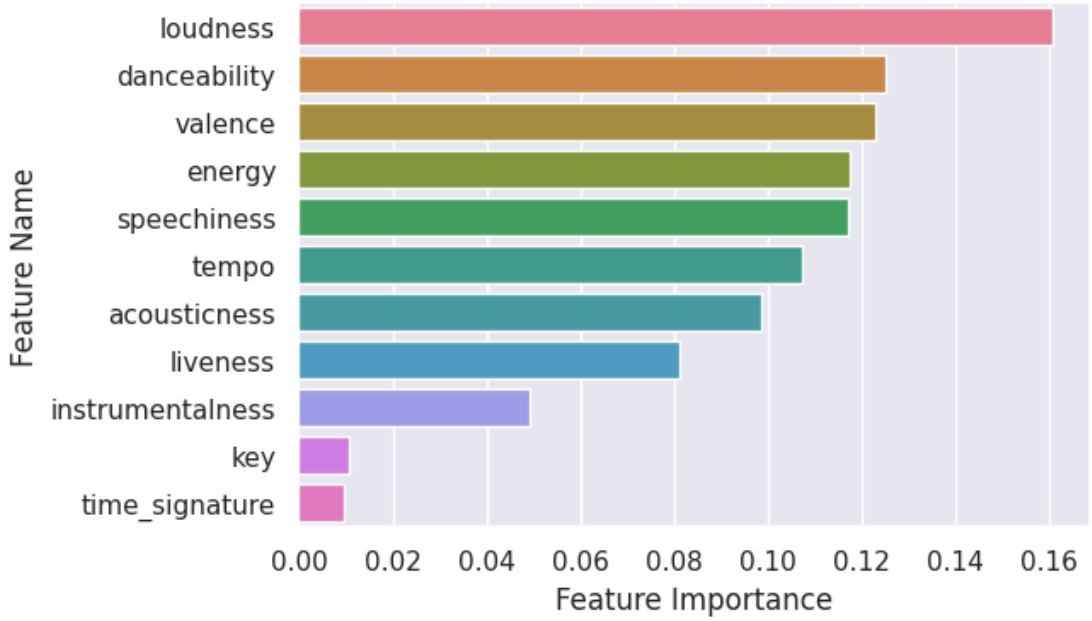
Machine learning models are used to make predictions, and we realized some models have the attribute `feature_importances_` which could possibly guide us in identifying one audio feature that impacts tracks' popularity the most. We now redefine the question to "which feature has the most importance in predicting popularity?"

We first approach this problem by training multiple regression models to predict popularity, since the popularity from the Spotify data is numerical. We only focus on models that provide the `feature_importances_` attribute, which include the Decision Tree, Random Forest, Gradient Boosting, and AdaBoost regressors. However, all regression models perform poorly in predicting popularity, with low scores on the validation data.

Model	Training Score	Validation Score
Decision Tree Regressor	0.06563	0.03724
Random Forest Regressor	0.10963	0.07286
Gradient Boosting Regressor	0.17203	0.08489
AdaBoost Regressor	0.04484	0.04306

Therefore, we can now turn this into a classification problem. Instead of predicting the numerical popularity, we categorize the popularity into three distinct groups: low, medium, and high. After ensuring that our data is evenly distributed, we apply classification models to predict the popularity by assigning each track to a certain group. This time, we get much better results:

Model	Training Score	Validation Score
Decision Tree Classifier	0.41465	0.39265
Random Forest Classifier	0.50467	0.41373
Gradient Boosting Classifier	0.59271	0.41980
AdaBoost Classifier	0.40554	0.39130



The Gradient Boosting Classifier gives the highest score on validation data, so we use that as our model to identify feature importance on predicting popularity. By the barplot shown above, `loudness` has the most significant importance on the track's popularity.

Limitations

Our project faced several limitations and challenges that impacted its progress. Firstly, during data acquisition, we frequently encountered rate limits when fetching audio features from Spotify's Web API, which blocked us from obtaining crucial data for analysis, further delaying the project. Additionally, our initial plan was to use StockX's Web API for price and market analysis. However, access to the API was granted on an application basis, and our application was not approved.

While training the machine learning models, we should have dedicated more time to hyperparameter tuning, since we observed overfitting on the training data and low accuracy on validation data, indicating that the models still did not generalize well to unseen data. This may impact the reliability of feature importance evaluations and yield an inaccurate assessment of audio features' influence on a track's popularity.

Project Experience Summary

Kevin Louie

- Split track data retrieved using the Spotify API into collaborative and solo tracks and transformed data using Pandas by removing unused features and re-balancing, preparing data for analysis.
- Conducted T-tests on track data with SciPy to identify significant differences in the popularity of collaborative and solo tracks on average.
- Investigated trends in the number of collaborative tracks per year using linear regression and visualizations with SciPy and Matplotlib respectively, revealing the evolution of collaborative tracks over time.

Terence Tang

- Retrieved track details using Spotipy, a lightweight Python library for Spotify's API, to perform analysis on music trends and audio features.
- Visualized correlations between audio features and popularity using Seaborn's heatmap to identify the feature with strongest linear relationship to popularity.
- Trained multiple machine learning models from scikit-learn to predict track popularity and identify the most important feature in the best-performing model.