ebtezcan / **dsc-phase-3-project**

generated from jirvingphd/osemn-project-template

GPL-3.0 License

0 stars **0** forks

Star

Unwatch

Code Issues Pull requests Actions Projects Wiki Security

master

...

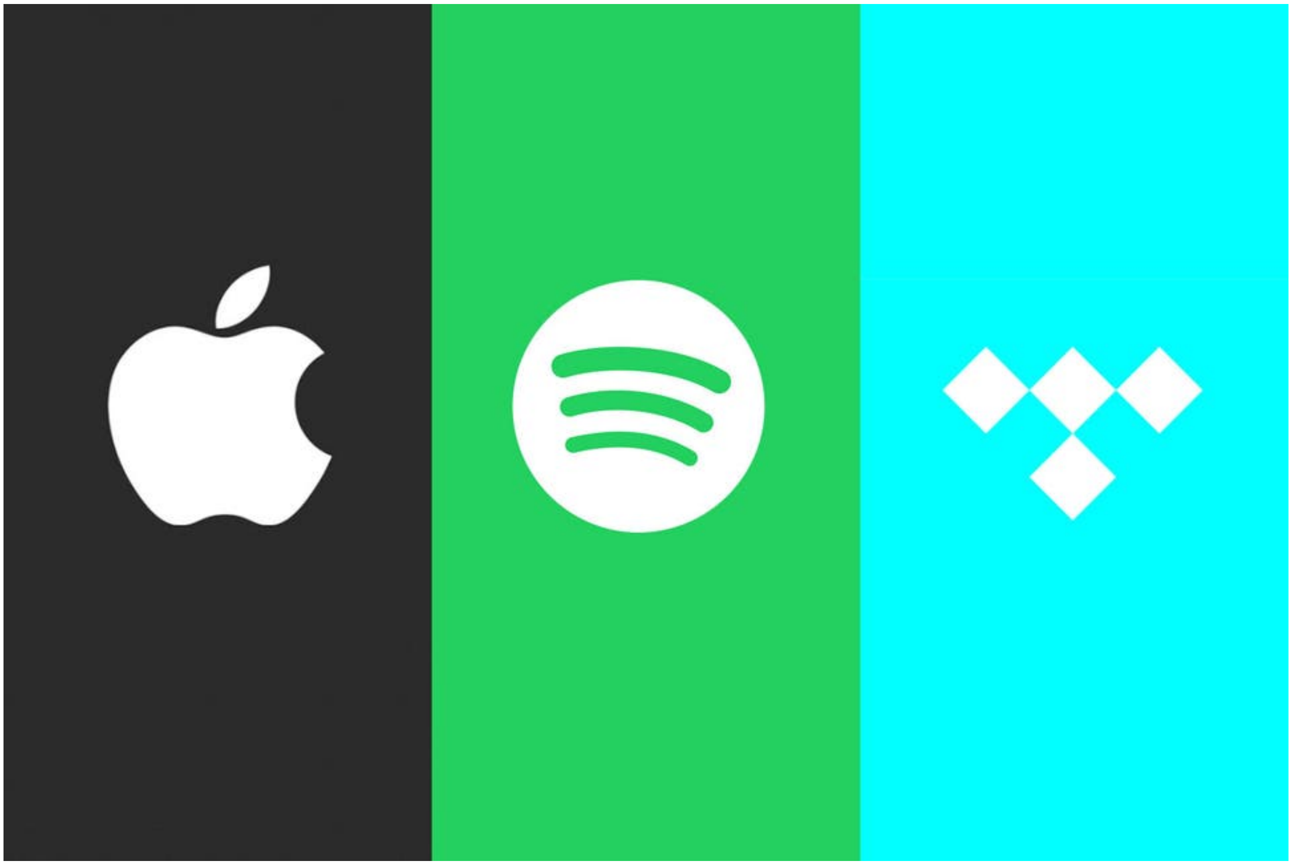
ebtezcan Minor changes to README, notebook finalized, initial draft ... ... 18 seconds ago 17

View code

Music Streaming Wars: Song Popularity Prediction

Training and testing machine learning models on music data from Spotify to predict a song's popularity.

Author: E. Berke Tezcan



> Image from https://static.standard.co.uk/s3fs-public/thumbnails/image/2015/06/30/15/apple-music-spotify-tidal.jpg?width=968&auto=webp&quality=75&crop=968%3A645%2Csmart

Introduction

With Apple Music announcing on May 17th that they will be providing lossless audio along with spatial audio by Dolby Atmos for their subscribers and Tidal continuously providing exclusive content from artists, the competition among audio streaming platforms is heating up. Spotify would like to stay competitive by being able to predict which songs are going to be popular ahead of time so that they can curate even better playlists and sign deals with up-and-coming artists to have exclusivity on their content. This would not only help retain the current subscribers but also help market the platform to new subscribers as well.

For this project, we were hired by Spotify to train and test a machine learning model that can accurately predict whether a song is going to be popular or not. In order to achieve this, we will be testing out different machine learning models and will look at what attributes of a song are the most important for determining its popularity.

Data

We used a dataset that has approximately 232,000 tracks from 2019 that was collected from Spotify's API and published on Kaggle for public use. The dataset can be found in the following link:

Spotify Tracks DB - https://www.kaggle.com/zaheenhamidani/ultimate-spotify-tracks-db

Since we ran classification models we had to feature engineer a column that required us to use additional data to determine whether a song was popular or not (see below Methods section for more information). The data used for this purpose was also gathered from Spotify's API and is available on Kaggle:

- Top 50 Songs - 2019: https://www.kaggle.com/leonardopena/top50spotify2019
- Top 100 Songs - 2019: https://www.kaggle.com/reach2ashish/top-100-spotify-songs-2019

Methods

We started off with exploratory data analysis (EDA) to understand the columns and the dataset overall. The detailed explanation of each column can be found in Appendix A. During the EDA, we found that the "Children's Music" genre was duplicated due to different characters used in the dataset. After addressing this discrepancy by renaming this genre, we saw that approximately 56,000 tracks were duplicated. We proceeded to group the data by their unique id numbers, and kept the maximum values of each column among the duplicated values (refer to the notebook for more information). Additionally, we checked to see if there were any missing values in the dataset and did not find any.

We then had to feature engineer our target column by using the popularity scores column. We binarize this column by establishing a cutoff popularity score. After looking at the range of popularity scores within Top 50 and Top 100 songs from the same year, we established the cutoff point to be greater than or equal to 58 out of 100.

After this, we proceeded with one hot encoding the categorical columns "key", "mode" and "time_signature". The dataset was then ready to be split into training and testing sets and used to train our models. Our modelling process started with training a baseline dummy classifier model for comparison and continued with training/testing a Random Forest model, XGBoost model and a Logistic Regression model. We additionally used grid searches that optimized for the recall scores to tune the hyperparameters of these models.

Due to a class imbalance problem, we additionally had to use SMOTENC on our data prior to training the models. The outliers in the data were also removed and the dataset as a whole was scaled prior to training and testing the Logistic regression models.

Results

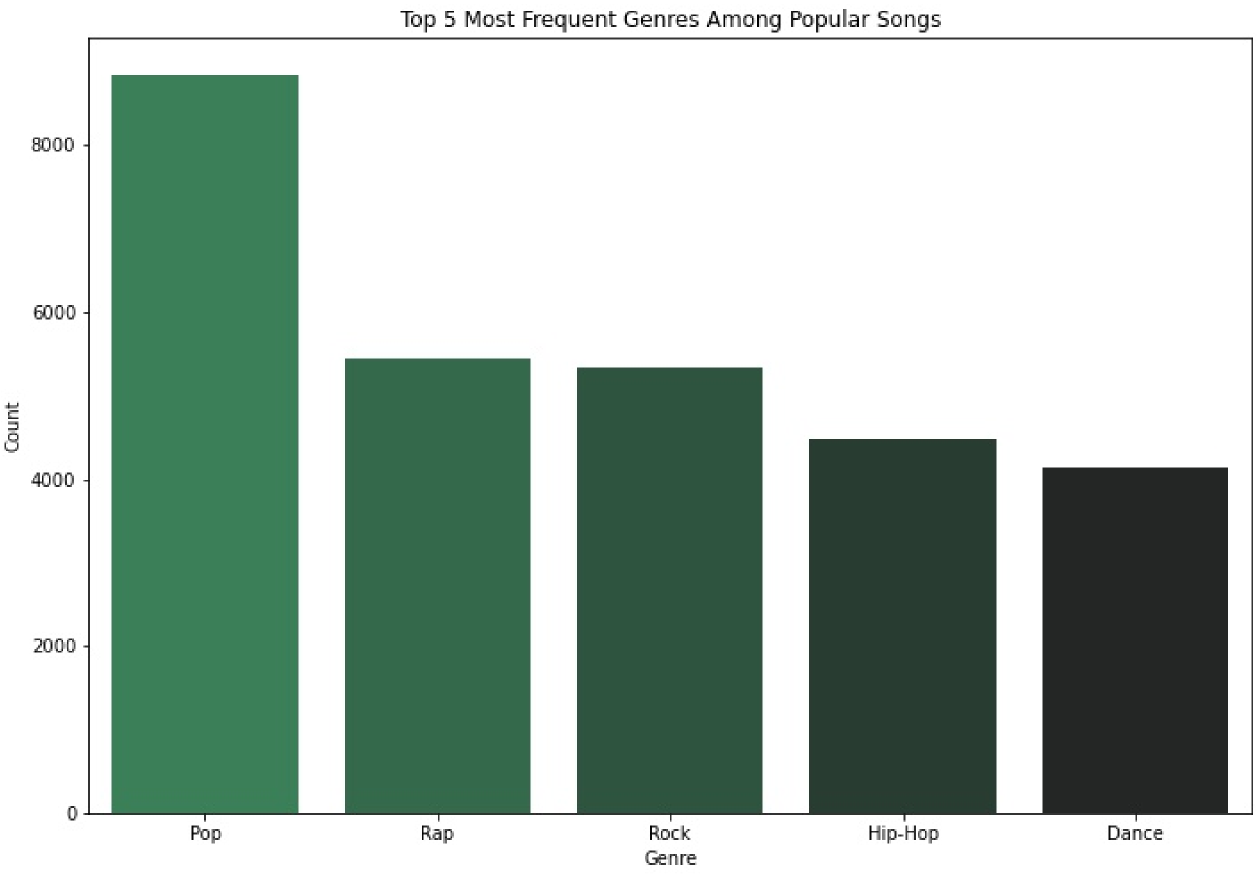
Below are the ranked results for our models:

Model Type	Recall Score
XGBoost	0.66
Logistic Regression	0.65
Random Forest	0.59
Dummy Classifier	0.12

The best model we trained, XGBoost, performed 54% better compared to the dummy classifier at 66% correct prediction for a song being popular.

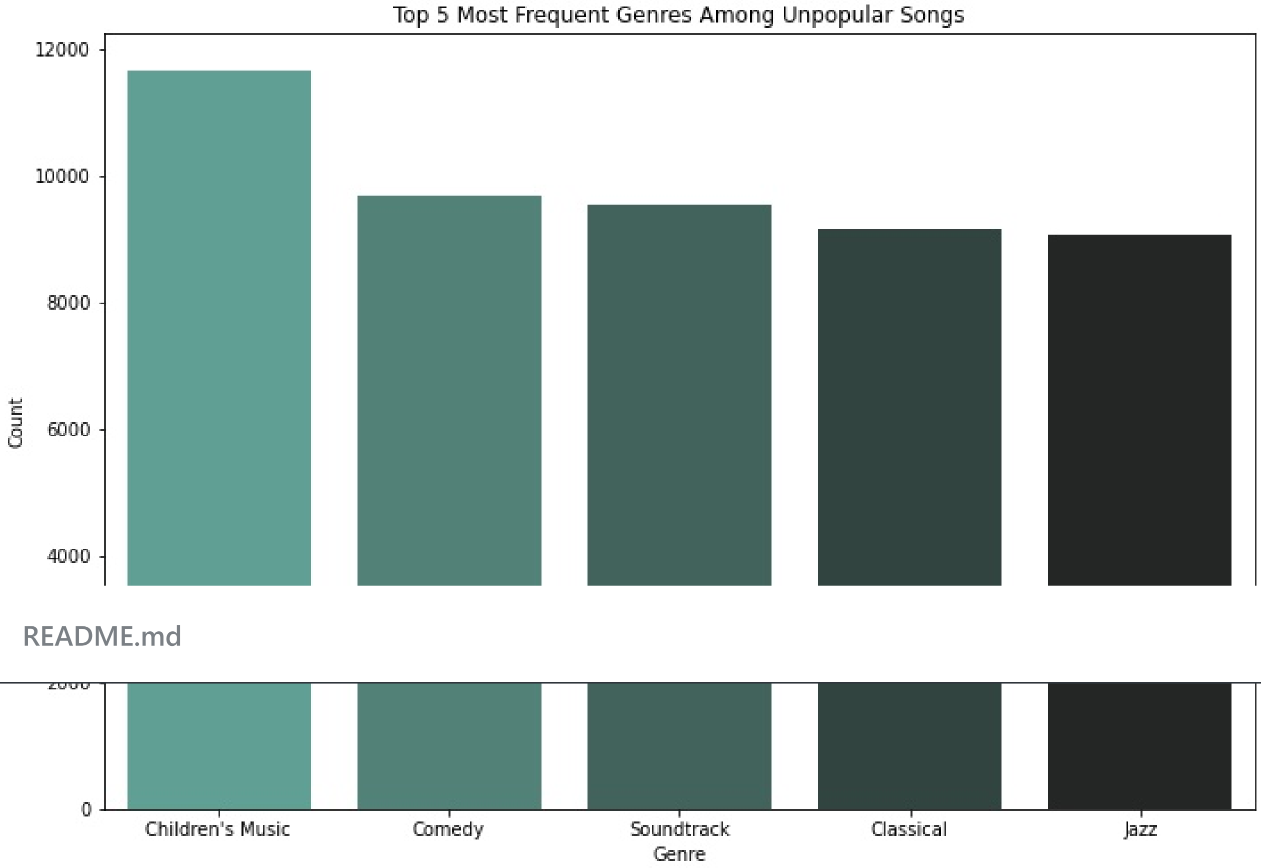
Aside from the models, we additionally explored different attributes within the popular songs and the unpopular songs separately.

Top 5 Genres within Popular Songs



> Above bar graph shows us the most frequent genres among popular songs. Most frequently a popular song had Pop as their genre followed by Rap, Rock, Hip-Hop and Dance. These results make sense and are in-line with a survey conducted by IFPI (https://www.statista.com/chart/15763/most-popular-music-genres-worldwide/).

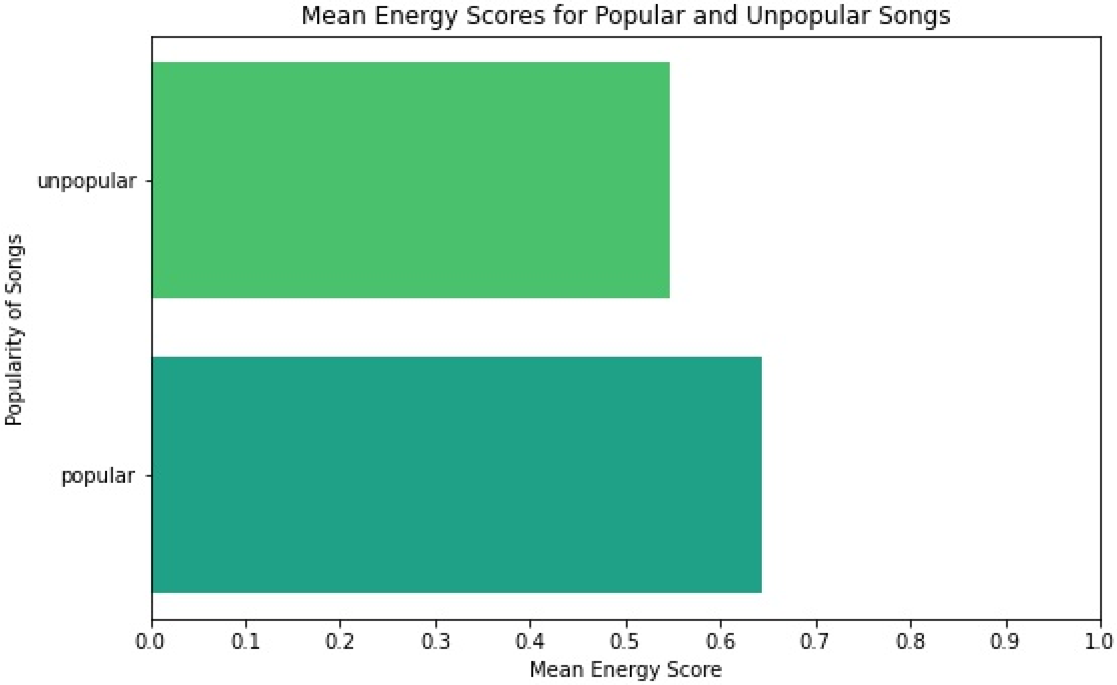
Top 5 Genres within Unpopular Songs



README.md

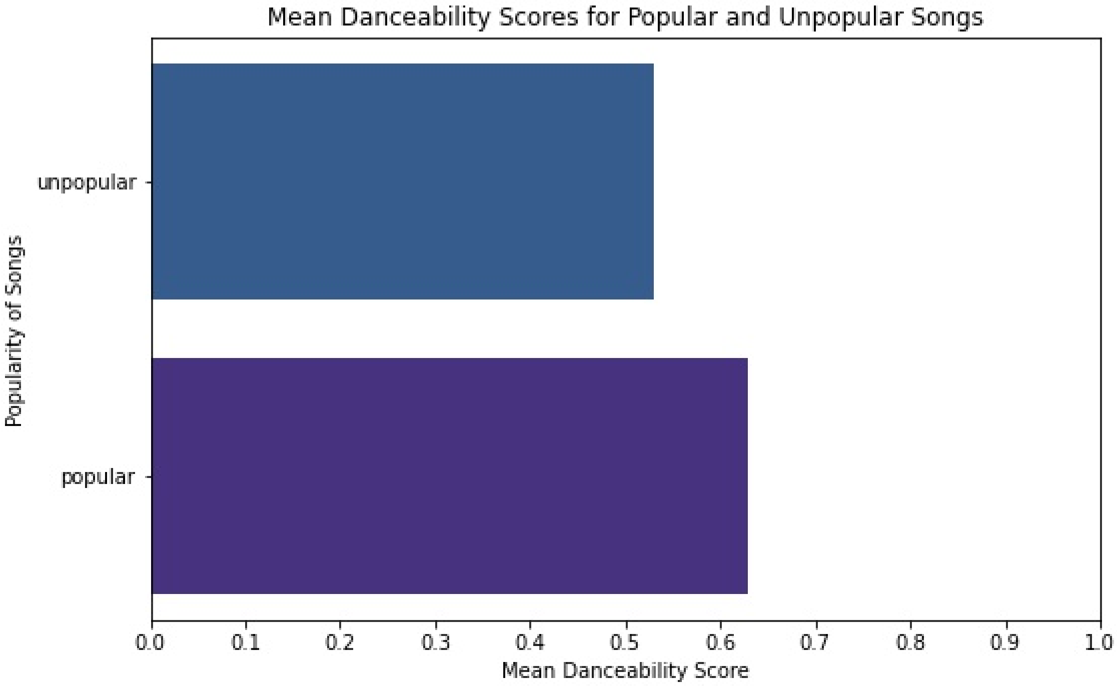
> The most frequent genres of unpopular songs can be seen above. The results make sense as these genres tend to have a more niche fanbase or are not represented in the mainstream media.

Energy Scores of Songs by Popularity



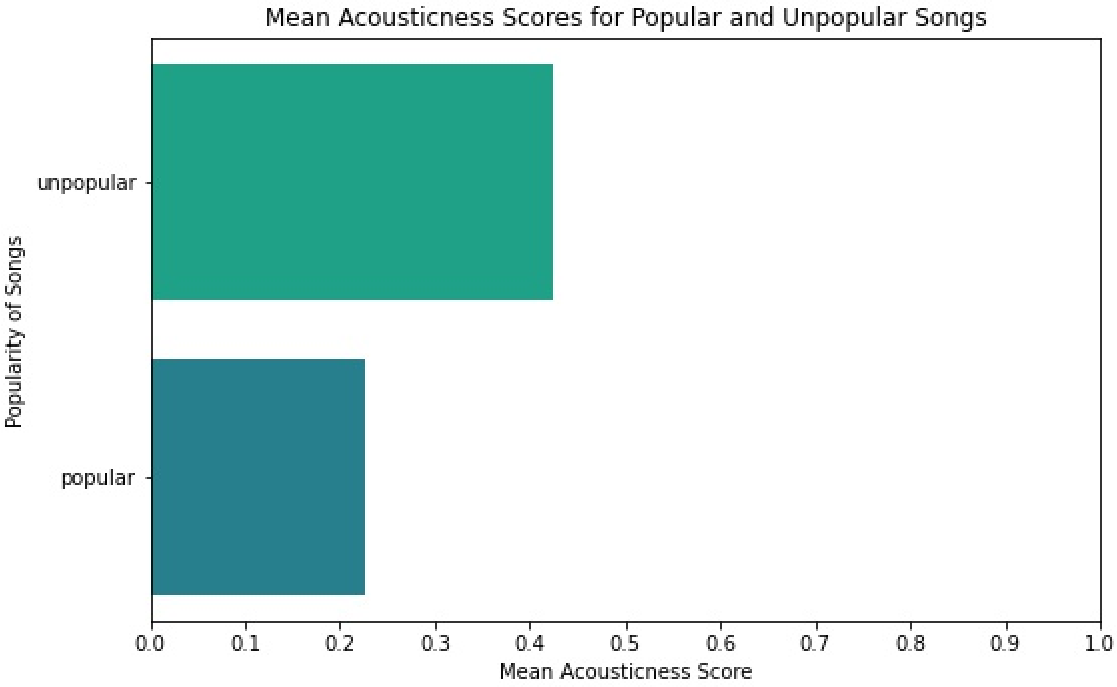
> As we can see above, popular songs tended to be more energetic compared to unpopular songs. This makes sense since the most frequent genres we explored tend to also be energetic genres.

Danceability Scores of Songs by Popularity



> Above, it is clear that the popular songs tended to have a higher danceability score compared to unpopular songs. This follows the same trend as the energy scores where majority of the popular songs are high energy and danceable (refer to Appendix A for definition of "danceability": high tempo, high beat strength etc.)

Acousticness Scores of Songs by Popularity



> Similar to the energy and danceability scores we see that the popular songs tended to have a lower acousticness score. Since acoustic songs are usually lower energy and rarely danceable this follows the same trend we've been observing.

In a competitive environment like the music streaming market, it is vital to retain current subscribers and add new subscribers over time. By accurately predicting which song will be popular next, companies like Spotify can leverage this information to create better playlists and find and sign exclusivity deals with established and up-and-coming artists more easily. To sum up, our analysis of approximately 176,000 songs from 2019 showed the following:

- Popular songs tend to have Pop, Rap, Rock, Hip-Hop and Dance as their genres.
- More niche genres such as Children's Music, Comedy, Soundtracks, Classical and Jazz tend to be unpopular.
- Generally, popular songs are higher energy, danceable, and therefore less acoustic.

Limitations & Next Steps

One of the major limitations we faced during this project was computing power. All training and testing of ML models were completed on a dated quad-core CPU which caused long processing times (sometimes in excess of 4 hours). Due to the limited amount of time we had, we were unable to iterate on our gridsearches and optimize our models to potentially perform better. Additionally, these hardware limitations also caused processing times of SHAP explainers to be too long, which we then excluded from our notebook as they never finished running.

In the future, instead of using pre-collected data from 2019, we would like to use more current data by utilizing the Spotify API to analyze current trends and build models.

For More Information

Please review my full analysis in my Jupyter Notebook or my presentation. For any additional questions, please contact Berke Tezcan at berketezcan@gmail.com.

Repository Structure

```
├── README.md               <- The top-level README for reviewers of this
project.
├── final_notebook.ipynb    <- Narrative documentation of analysis in jupyter
notebook
├── notebook.pdf            <- Narrative documentation of analysis in PDF
├── presentation.pdf        <- PDF version of project presentation
├── images                  <- Both sourced externally and generated from code
└── data                    <- Externally sourced data
```

Appendix A: Explanation of Features

Since the data was collected by using Spotify's API, each column has detailed descriptions to be used as reference. Below is the table that was used to interpret each column.

Key	Description
acousticness	A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
danceability	Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.
duration_ms	The duration of the track in milliseconds.
energy	Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy. For example, death metal has high energy, while a Bach prelude scores low on the scale. Perceptual features contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy.
instrumentalness	Predicts whether a track contains no vocals. “Ooh” and “aah” sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly “vocal”. The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.
key	The key the track is in.
liveness	Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.
loudness	The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track and are useful for comparing relative loudness of tracks. Loudness is the quality of a sound that is the primary psychological correlate of physical strength (amplitude). Values typical range between -60 and 0 db.
mode	Mode indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is 0.

Key	Description
speechiness	Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks.
tempo	The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.
time_signature	An estimated overall time signature of a track. The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure).
valence	A measure from 0.0 to 1.0 describing 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).
popularity	The popularity of a track is a value between 0 and 100, with 100 being the most popular. The popularity is calculated by algorithm and is based, in the most part, on the total number of plays the track has had and how recent those plays are. Generally speaking, songs that are being played a lot now will have a higher popularity than songs that were played a lot in the past. Duplicate tracks (e.g. the same track from a single and an album) are rated independently. Artist and album popularity is derived mathematically from track popularity. Note that the popularity value may lag actual popularity by a few days: the value is not updated in real time.

Releases

No releases published
Create a new release

Packages

No packages published

Publish your first package

Languages

- ● **Jupyter Notebook** 100.0%