
# Spotify, Play Something

## A new mood-based recommendation engine

07.20.2021

Valentina Rizzati
Data Scientist
Spotify

## Opportunity

I am a Data Scientist at Spotify and I've been asked to come up with an idea to increase retention and Daily Active Users (DAUs). Building upon the past success of personalization-based strategies, I have decided to build a recommendation engine that provides a curated Top 10 playlist based on the user's daily mood.

For the purpose of this prototype, I will create a recommendation engine that is built upon the user's **mood** at the time of logging into the app.

The idea of building a recommendation engine based on mood feeds into a larger vision of creating a matrix between user personas (i.e. mirroring a user's musical taste) and occasion/mood. The applicability of such a vision is of course tied to access to a large pool of user data.

## Impact Hypothesis

By providing the user with a more personalized experience with relevant content on a daily basis, Spotify will likely succeed in increasing app stickiness, retention and the number of DAUs. The objective is to make listening to music on Spotify part of a user's daily habits and to streamline their path to daily recommendations.

## Data

I have collected data on Spotify artists, albums and tracks via the Spotify API. Thanks to spotipy, a lightweight Python library for the Spotify API, I did not have to store json files on MongoDB but I could interact with them directly in the jupyter lab interface.

I obtained the mapping between personality traits and music genres from Adrian C. North's study on Individual Differences in Musical Taste. It's important to note that Spotify assigns music genres at the artist level. To gather the data for all the genres that were mentioned in the study, I had to first pull a table of artists by genre, then a table of albums by artist and finally a table of tracks by albums. I saved these tables locally in a SQL database for easier access.

After preprocessing and cleaning the data, I have obtained a total of **~ 256,000 tracks** across 15 music genres. Finally, I realigned the genre nomenclature between the study and Spotify to enable the mapping between music genre and moods, derived from the study.

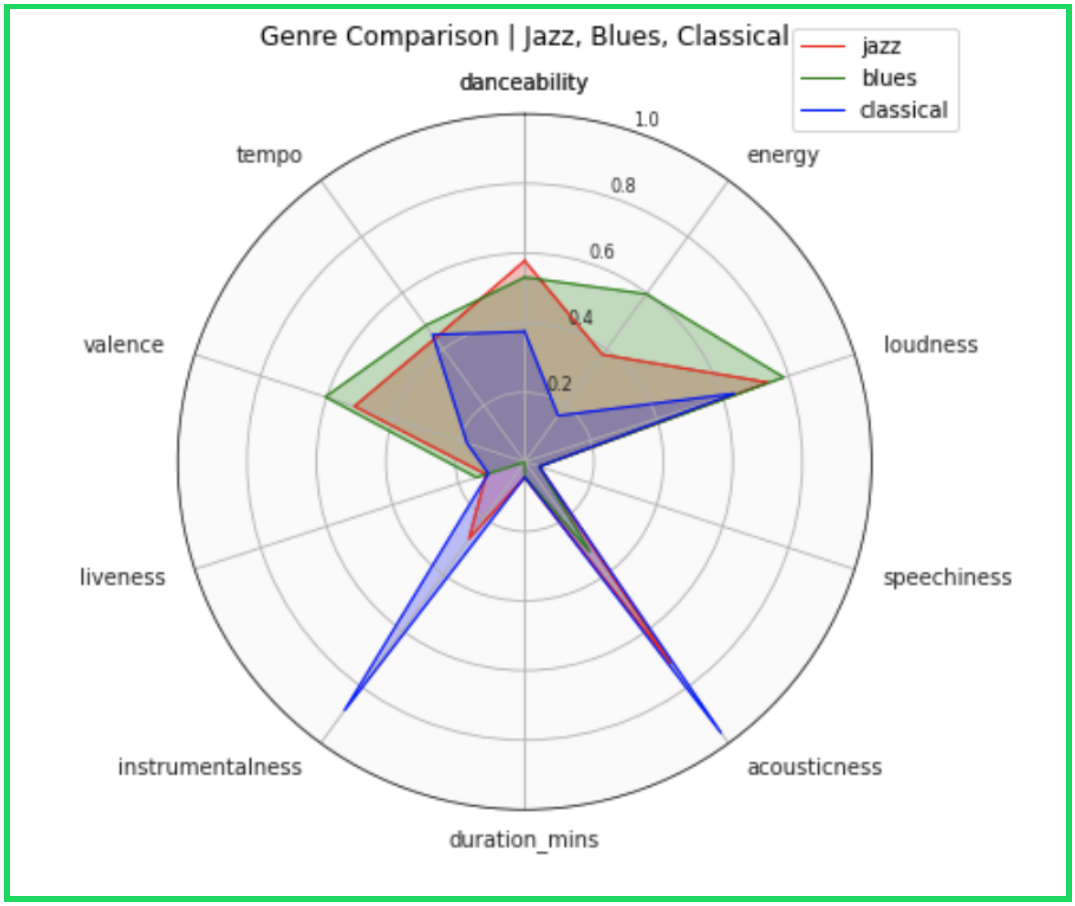
As an additional step, to make the personality traits more relatable and actionable, I merged the insights from the study with my domain knowledge to assign a user persona to every music genre. This user persona will constitute the first choice that a user will make on the final app.

# Algorithm

### DATA LOADING & PREPROCESSING

In this section I pulled data from the Spotify API through spotipy. I structured my data across the *artists*, *albums* and *tracks* tables and I saved these in a SQL *spotify* database. This step took some time because the Spotify API kept serving me a *timeout error*. After some testing, I found out that a combination of Jupyter Lab and Google Colab would allow me to pull all the data I needed in a relatively timely fashion.

As a second step, I preprocessed the data and segmented the tracks by music genre. I also conducted EDA on the music features and compared different music genres on a radar graph. See an example below.



### MODELING THE RECOMMENDATION ENGINE

My recommendation engine is based on three main steps:

**Step 1**

- **Action on App:** User select user persona (mood) they identify with
- **Recommendation**: music genre (not visible to the user)

**Step 2**

- **Recommendation**: top 3 songs that most align with music genre in step 1 (visible to the user)

**Step 3**

- **Action on App:** User select most attractive song that is served in step 2
- **Recommendation (final)**: top 10 songs that most align with the song selected by the user in step 2 (visible to the user)

These recommendations will be based on cosine similarity.

### DEPLOYING THE RECOMMENDER ON STREAMLIT

I will deploy my 3-stage recommender on streamlit with the flow as described in the section above.

## Tools

For general data manipulation, modeling and visualization, I have pandas, numpy, matplotlib, scikit-learn, and spotipy.

To reduce the computational cost I have used the GPU and high-RAM on Google Colab.

To store data I have used a SQL database.

Finally, I will use Streamlit to deploy the app.

## Next Steps

To finalize the project,  I will need to finalize the recommender and build the app on Streamlit.