Music Genre Detector

Radoslav Vasilev, Hans Markus Haug, Lars Erik Solbakken, Erlend Stensbø

[01.11.25]

1: DESCRIBE THE PROBLEM

SCOPE

The idea behind this project is to make a small machine learning system that can predict the genre of a song based on different audio features, like how fast or energetic it is. The result will be a simple web app where a user can choose a song or load an .mp3 file and the model will guess what genre it belongs to.

We chose this project because we have always liked music and thought it would be interesting to see how computers can recognize patterns that define a genre – like what makes something "rock" or "pop". It is also a good beginner-friendly way to learn about how machine learning models can work with numerical data.

Machine learning fits this problem well because it can find patterns in the data that are hard to describe manually. Without it, you would need someone with musical knowledge to listen to every song and classify it, which is both slow and subjective.

There are already big platforms like Spotify that do this on a large scale, but this project is meant as a simplified educational version – something smaller and easier to understand.

The system is meant for:

- *People curious about how ML models can be used in music.*
- *Students or developers who want a small example of ML in action.*

If it were a real product, it could be used in apps that organize playlists or suggest songs.

Business Objective

For us, the main goal is educational, but in a broader sense, this kind of system could be part of a music recommendation engine or something that helps users discover songs similar to the ones they already like.

Resources

- Software: Python, scikit-learn, Pandas, NumPy, and Gradio for deployment.
- Hardware: Just a laptop.
- Time and personnel: Us, working regularly with it until the submission deadline.

METRICS

To measure how well the model performs, we will use these metrics:

- Accuracy – how often the model predicts the correct genre.
- F1-score – to see how well it balances between precision and recall, especially if some genres have more songs than others
- ROC AUC (Receiver Operating Characteristic - Area Under the Curve): A measure of the model`s ability to distinguish between classes. Specifically, it plots the True Positive Rate (Recall) against the False Positive Rate across all possible classification thresholds.

If the model reaches at least 70% accuracy, we will consider that good for a first version. From a user's perspective, the model should feel "mostly right," meaning that the predictions make sense most of the time.

2: DATA

The data will come from Kaggle's Dataset of songs in Spotify. Each entry represents a song with values like:

- *Danceability, energy, valence, acousticness, etc.*
- *The label is the genre.*

The dataset already contains many songs (tens of thousands), which is plenty for a project like this. The labels are reliable since they come from Spotify's own metadata.

We focused on 10 major genres by grouping subgenres into broader categories such that for example "hard-rock" and "metalcore" belong simply to "rock" and "cantopop" and "k-pop" are in the "pop" genre. This is to reduce noise and improve model generalization.

Data handling and preparation:

We will use Pandas to clean the data – remove missing values, maybe drop some irrelevant columns. Then normalize the features using StandardScaler, since some things are on different scales. The label (genres) will be converted into numbers using a LabelEncoder.

Ethical/Privacy considerations:

There are no personal or sensitive data here – only audio features of songs, so It is safe and ethical to use.

3: MODELING

We will treat this as a supervised classification problem, where the model learns to map numerical features to genres.

We will start with a Logistic Regression model to set a baseline. Then we will try more advanced models like:

- *Random Forest (since it usually performs well on tabular data),*

- *SVM, and maybe*
- *LightGBM for comparison.*

We will compare their accuracy using train/test splits and cross-validation.

To understand the model better, we will check feature importance to see which features matter most – for example, whether "tempo" or "danceability" has a stronger influence on predicting pop vs. classical.

If the results are not great, we will try improving by:

- *Tuning hyperparameters using GridSearchCV.*
- *Making sure classes (genres) are balanced.*

4: DEPLOYMENT

We plan to deploy the model as a Gradio web app because it is simple and easy to share.

How it will work:

- *The user selects a song from a small list.*
- *The model predicts the genre and shows it on screen.*
- *The app can also display a top 5 list of most probable genres.*

After deployment, we can test how it performs with new songs. In theory, the system could later be expanded to use Spotify's API to get real song data automatically.

Maintenance and improvement:

If we continue working on it later, we could retrain it with more up-to-date data or add deep learning models for better accuracy.

5: REFERENCES

- *Kaggle – Spotify Tracks Genre Dataset:* *https://www.kaggle.com/datasets/thedevastator/spotify-tracks-genre-dataset/data*
- *Scikit-learn Documentation: https://scikit-learn.org/stable*
- *Gradio Documentation: https://www.gradio.app/docs*
- *Example Kaggle notebooks for music genre classification.*
- *"GTZAN Dataset: Music Genre Classification" – a well-known dataset in this field.*