Music Genre Detector

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

[08.11.25]

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 Spotify Tracks Genre dataset. 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 treated this as a supervised classification problem, where the model learns to map numerical features to genres.

Model Selection and Baseline: We first established a baseline using **Logistic Regression,** which achieved an accuracy of 41%. The model was underfitting, which meant that it could not grasp the complexity of the genre classification. The ROC_AUC score was 80%.

We then trained and evaluated several other models to find the best performer. The results were gathered using a 3-fold cross-validation and are summarized below:

Model	Accuracy	Macro F1-score	ROC_AUC	top_k_categorical _accuracy (excluded)
Logistic Regression (Baseline)	41%	35%	80%	NOT MEASURED
SVM	54%	51%	85%	NOT MEASURED
Random Forest Classifier	61%	58%	90%	by memory: top_2: 75% top_3: 82%
LightGBM	60%	57%	89%	NOT MEASURED
Ensemble	NOT NOTED	NOT NOTED	NOT NOTED	NOT MEASURED

```
------------------------------
Validation Set Report, Best RF
              precision    recall  f1-score   support

   classical       0.69      0.79      0.74       600
     country       0.62      0.63      0.62       600
  electronic       0.73      0.63      0.67      1000
        folk       0.61      0.65      0.63      1000
     hip-hop       0.39      0.34      0.36       200
        jazz       0.62      0.44      0.51       400
         pop       0.53      0.51      0.52      1000
       r-n-b       0.50      0.39      0.44       400
      reggae       0.55      0.65      0.60       800
        rock       0.63      0.69      0.66      1000

    accuracy                           0.61      7000
   macro avg       0.59      0.57      0.58      7000
weighted avg       0.61      0.61      0.60      7000
```

[ ]

```
from sklearn.metrics import roc_auc_score
y_val_proba = best_rf.predict_proba(X_val_scaled)
roc_auc_score(y_val, y_val_proba, multi_class='ovr', average=
```

```
np.float64(0.8980917748805956)
```

Final model and Tuning: The **Random Forest** model clearly performed the best, nearly achieving our accuracy goal of 70% with a final score of 61%. We improved this model by using RandomizedSearchCV to tune its hyperparameters.

Model Analysis: To understand our final model, we analyzed its **feature importance.** This showed that speechiness, danceability, and acousticness were the three most important features for predicting a genre.

4: DEPLOYMENT

We successfully deployed the final model as a Gradio web app.
Final Model Choice: *Based on our analysis in the modeling phase, we selected the* ***Random forest model*** *(with an accuracy of 61%) to be the engine for the web app.*

How it will work:

The apps allow a user to either select a song from a list or upload a song under 5mb. The second web app uses the Reccobeats API (Spotify no longer allows developers to use their Web aPIto extract the features and if using the second method) which leads to poorer performance.

The app then displays the predicted genre and a list showing the model's confidence probability for each possible genre. The system was successfully tested and performs as expected.

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.*