# Music Genre Classification and Song Recommendation System

*Final Project Report*

*ECS 170 Fall 2024*

Devon Streelman, Ameen Salim, Sarbani Kumar, G Krupaa No Name, Ansh Chhabra, Jesus Delgado-Perez, Loc Nguyen, Lorenzo Fernandes

December 10, 2024

# 1 Introduction

The objective of our project was twofold: to develop a music genre classification model and a song recommendation system that leverages machine learning. Our approach to classification utilized an ensemble learning method based on the gradient boosting framework, enabling us to explore how combinations of these audio features shape the unique qualities of a song. For recommendations, focusing on audio features such as tempo, danceability, and acoustics, our system seeks to enhance the music discovery experience for users.

The motivation behind this project stems from the significant impact that music has on daily life and the increasing role of AI in transforming user experiences. Music recommendation systems are a cornerstone of platforms like Spotify and Apple Music. This project builds on that foundation by incorporating user input directly into the recommendation system, rather than solely predicting the user's mood.

# 2 Background

## 2.1 Genre Classification in the Music Industry

Music genres often overlap in characteristics such as tempo, instrumentation, and vocal style, making accurate classification challenging. Automated systems analyze audio features quantitatively to identify subtle patterns beyond human perception. Genre classification has real-world applications, particularly in streaming platforms where it powers personalized listening experiences. For example, Spotify combines editorial curation and algorithm-driven recommendations, integrating human expertise with machine learning to deliver tailored, engaging content for users (*Understanding Recommendations on Spotify*).

## 2.2 Significance of Our Model

While Spotify excels in recommendations based on listening habits, its genre classification systems often lack user-provided inputs like mood, preferences, or specific activities. Our model addresses this gap by using a balanced dataset to ensure fair representation of all genres and tailoring recommendations based on user input. This approach not only enhances the user experience with personalized music discovery but also supports smaller, overlooked artists by increasing their visibility through our recommendation system.

# 3 Methodology

## 3.1 Data Selection

### 3.1.1 Data Selection for Genre Classification

Our initial dataset, sourced from Spotify data, included approximately 2,000 songs with comprehensive audio features such as tempo, danceability, and energy, along with corresponding genre classifications. This dataset seemed promising due to its detailed features extracted from the Spotify API. However, during preliminary exploratory data analysis (EDA), significant limitations became evident.

First, the small size of the dataset raised concerns about its generalizability for machine learning tasks. Moreover, univariate analyses and visualizations revealed severe class

imbalances, with genres like pop and rock being heavily overrepresented while others, such as classical, had very few instances. This imbalance risked introducing bias, where the model might disproportionately favor dominant genres, leading to poor performance on minority classes.

Recognizing these limitations, we sought a more robust dataset. Our final dataset, sourced from a Kaggle dataset called “Spotify Tracks Datasets”, addressed these issues with a larger size and balanced representation (Maharshi Pandya, 2022). After data cleaning including the removal of non-predictive features, like track ID and time signature, and filtering out niche genres we refined the dataset to include five major genres: pop, rock, jazz, classical, and country. Each genre was represented by 1,000 data points, ensuring an equitable training process, as seen in Figure 1. This balanced structure provided a solid foundation for training our model, minimizing bias and improving classification accuracy.

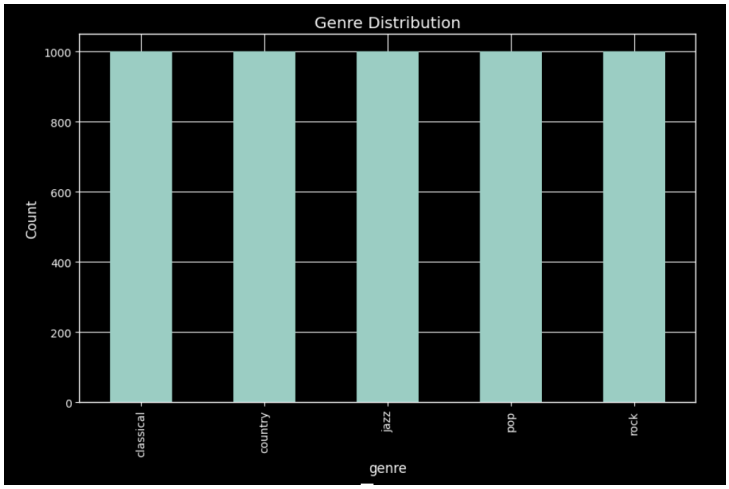


*Figure 1: The distributions of different features across genres after scaling*

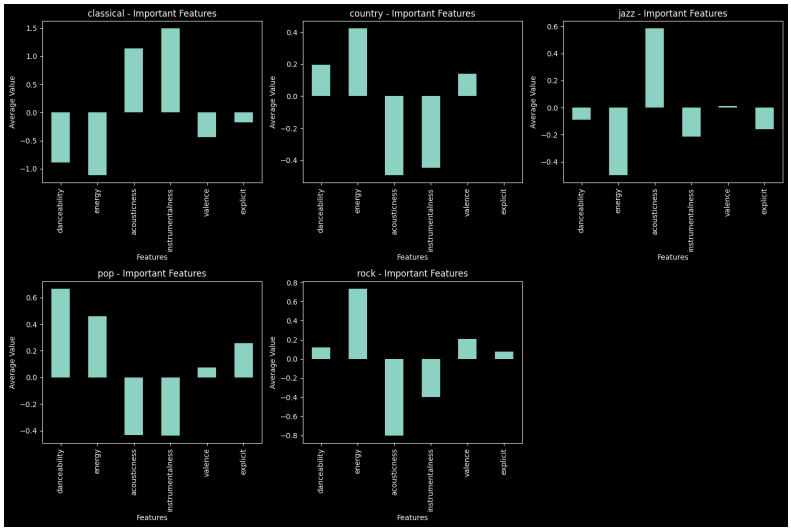


*Figure 2: Important features and each of their average values per genre*

### 3.1.2 Data Selection for Song Recommendation

Initially, we planned to utilize the Spotify API as the data source for the song recommendation system, leveraging its extensive catalog of millions of songs. This approach would have ensured highly relevant and up-to-date recommendations. However, due to the deprecation of key Spotify API endpoints, we pivoted to an alternative solution. After an extensive search, we selected a dataset containing over 1,000,000 songs, which included features comparable to those provided by Spotify, such as tempo, danceability, energy, valence, and genre. This dataset allowed us to maintain the quality and relevance of recommendations while addressing the API's limitations.

To facilitate efficient data filtering required in later stages of development, we converted the extensive dataset into a SQL database. This database included a table containing all the essential features for song recommendations, such as popularity, danceability, energy, and genre. During initial exploration, SQL queries were used to identify anomalies and refine the dataset. For instance, we found that certain popular songs with low energy levels were disproportionately represented as white noise. To ensure the database remained balanced and relevant, we filtered out excess white noise entries while retaining a representative selection to accommodate users seeking sleep music. Similar adjustments were made for other overrepresented song types, refining the dataset for more effective recommendations.

## 3.2 Model Development

### 3.2.1 Genre Classification

Through model selection, we trained and evaluated Logistic Regression, SVM, Decision Trees, Random Forest (RF), and XGBoost, ultimately settling on an ensemble of XGBoost and Random Forest due to their superior accuracy. Individually, RF achieved an accuracy of 82.7% with weighted average precision, recall, and F1-scores of 83%, as shown in the RF Classification Report in figure 3. Similarly, XGBoost delivered comparable results, achieving an accuracy of 82.6% with weighted averages also at 83% in figure 4,. Both models performed best on classical music, achieving precision, recall, and F1-scores of 94%, but struggled most with rock music, where these metrics fell to approximately 74%. Combining RF and XGBoost in an ensemble leveraged their strengths and mitigated individual weaknesses, particularly in challenging genres like rock.

RF Accuracy: 0.827

RF Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| classical | 0.93 | 0.92 | 0.92 | 216 |
| country | 0.80 | 0.73 | 0.76 | 190 |
| jazz | 0.80 | 0.83 | 0.82 | 192 |
| pop | 0.81 | 0.90 | 0.86 | 210 |
| rock | 0.78 | 0.74 | 0.76 | 192 |
| accuracy | | | 0.83 | 1000 |
| macro avg | 0.82 | 0.82 | 0.82 | 1000 |
| weighted avg | 0.83 | 0.83 | 0.83 | 1000 |

*Figure 3: Random Forest Classification Report*

XGBoost
Accuracy: 0.826

Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.94 | 0.94 | 0.94 | 216 |
| 1 | 0.76 | 0.73 | 0.75 | 190 |
| 2 | 0.83 | 0.86 | 0.85 | 192 |
| 3 | 0.84 | 0.84 | 0.84 | 210 |
| 4 | 0.73 | 0.74 | 0.74 | 192 |
| accuracy | | | 0.83 | 1000 |
| macro avg | 0.82 | 0.82 | 0.82 | 1000 |
| weighted avg | 0.83 | 0.83 | 0.83 | 1000 |

*Figure 4: XGBoost Classification Report*

Note: 0 - 4 represent classical, country, jazz, pop, and rock respectively

The lower accuracy for rock and country genres is likely due to feature variance and overlap with other genres. Rock and country have inconsistent features—some songs are slow, others uptempo—making it harder for the model to identify genre-specific traits. Additionally, feature similarities with pop, particularly in energy, valence, and acousticness (Figure 2), contribute to misclassifications. To address this, we applied grid search for hyperparameter tuning, but it yielded minimal improvement. This prompted us to adopt an ensemble model, as discussed in the final model section.

### 3.2.2 Song Recommendation System

To enhance the intelligence of our song recommendation system, we integrated LangChain with OpenAI's GPT-3.5 Turbo model, enabling it to dynamically generate SQL queries based on user-input for real-time filtering and display of results. Prompt engineering played a key role, with a detailed system message providing clear instructions to ensure accurate and task-specific outputs. For example, `"You are a helpful assistant that converts user song preferences into an SQL query. No other responses are needed. The features include... An example SQL query might be..."`, `etc.` This context was vital because GPT-3.5 relies on precise instructions. Without them, the model could generate incorrect queries or stray from its task. By clearly defining the task and expected output, we ensured consistent, accurate SQL queries, which demonstrated how effective LLMs can be in database operations.

To further refine accuracy, we fine-tuned GPT-3.5 Turbo with specific query examples and expected outputs. This process reduced ambiguity, improved the model's ability to address nuanced user preferences, and bridged the gap between general-purpose AI and our system's specific needs. Despite these improvements, initial queries to the LLM often prioritized popular tracks, resulting in repetitive artist recommendations. While this approach enhanced relevance for users, it introduced a limitation we aim to address by increasing diversity in future recommendations.

## 3.3 Final Model

The final project consists of two components: genre classification and dynamic song recommendations. For classification, an ensemble of Random Forest and XGBoost models achieved an accuracy of 83.6%, as seen in Figure 5, excelling in classical music with scores of

94% but struggling with rock at 74%. These results highlight strengths in distinct genres and challenges with overlapping features like those between pop and rock. The recommendation system leverages GPT-3.5 Turbo via LangChain to generate SQL queries based on user input. Fine-tuned prompts ensure accurate and relevant queries, dynamically filtering the database to deliver tailored recommendations.

```
Ensembled Accuracy: 0.836
Ensembled Classification Report:
              precision    recall  f1-score   support

   classical       0.95      0.94      0.94       216
     country       0.80      0.76      0.78       190
        jazz       0.82      0.87      0.85       192
         pop       0.85      0.86      0.85       210
        rock       0.75      0.74      0.74       192

    accuracy                           0.84      1000
   macro avg       0.83      0.83      0.83      1000
weighted avg       0.84      0.84      0.84      1000
```

*Figure 5: Classification report for ensemble*

Examples of LLM generated SQL Queries for the song recommendation system include:

1. **User Input:** "Give me some high energy rock and roll songs to keep me motivated!"
   **Generated SQL Query:** SELECT * FROM songs WHERE genre = 'rock' OR genre = 'rock-n-roll' AND energy > 0.7 AND tempo > 0.6 AND valence > 0.5 ORDER BY popularity DESC LIMIT 10;
2. **User Input:** "How about some songs for my underground rave in Germany."
   **Generated SQL Query:** SELECT * FROM songs WHERE genre = 'techno' AND energy > 0.7 AND tempo > 0.6 AND loudness > 0.5 ORDER BY popularity DESC LIMIT 10;
3. **User Input:** "Songs for my St Patrick's day party!"
   **Generated SQL Query:** SELECT * FROM songs WHERE genre IN ('irish', 'folk', 'celtic') AND energy > 0.5 AND tempo > 0.5 AND valence > 0.5 AND loudness > 0.5 ORDER BY popularity DESC LIMIT 10;

These examples demonstrate the system's ability to interpret user preferences and provide targeted, meaningful recommendations. By prioritizing features such as energy, tempo, and valence, and ranking by popularity, the system enhances user satisfaction.

## 4 Results

Overall, we successfully achieved the primary objectives of this project. For Genre Classification, our model attained an 84% accuracy across five genres—classical, country, jazz, pop, and rock—based on their audio characteristics. For Song Recommendation, the system effectively generates curated lists of popular songs based on user-input moods and activities.

Despite its strengths, the recommendation system faces challenges with highly specific prompts and a lack of diversity in suggested artists. These limitations highlight areas for improvement, such as refining the underlying language model and enhancing training to increase adaptability and diversity in recommendations.

## 5 Discussion

We successfully developed independent Genre Classification and Song Recommendation System models, enabling users to explore music preferences and discover new songs. The system stands out for its ability to suggest tracks tailored to specific moods or situations using a custom prompt feature that provides a personalized and dynamic listening experience. This project offered valuable insights into the characteristics of different music genres and the features that define songs. Leveraging these features, we created an innovative tool to enhance the music discovery experience. Additionally, this deeper understanding of genre classification and recommendation system design has broadened our expertise and prepared us to address similar challenges in the future.

## 6 Conclusion

While working on this project, we analyzed song features and their correlations with genres using heatmaps and univariate analysis. Among various machine learning techniques, an ensemble of Random Forest and XGBoost delivered the best results, evaluated using accuracy, precision, and recall metrics. This led to the implementation of the two independent components: Genre Classification Model and a Song Recommendation System, allowing users to explore music tailored to their mood or activity.

## 7 Github

Here is the Github link to our project: https://github.com/ameensalim1/the-greatest-team

## 8 Contribution

**Devon Streelman**: ML model development, LLM song recommendation system, fine tuning, EDA, and helped with the methodology section. **Ameen Salim**: ML model development, Created web app implementation for song recommendation system, helped add info for presentation, **Sarbani Kumar**: Researched new datasets on Kaggle and suitable ML models, wrote script and slides, wrote parts of the report and project check-in, helped to organise internal deadlines and meetings. **G Krupaa No Name**: Researched datasets on Kaggle, hyperparameter tuning, helped with presentation, and worked multiple sections of the report and edited the report. **Ansh Chhabra**: Finding dataset, creating slides for presentation, helped with model description section on the project report, unused work on front-end. **Jesus Delgado - Perez**: Front end development, helped write script, and presentation slides as well as edited and proofread the report. **Loc Nguyen**: Helped with researching models, tuning, and writing a part of the report. **Lorenzo Fernandes**:I helped create our Presentation and wrote the Conclusion and Discussion section of this report, also helped write parts of the script we used during our presentation.

# Works Cited

Pandya, Maharshi. "🎹 Spotify Tracks Dataset." *Kaggle*, 2022, https://doi.org/10.34740/KAGGLE/DSV/4372070.

"Understanding Recommendations on Spotify." *Safety & Privacy Center*, www.spotify.com/us/safetyandprivacy/understanding-recommendations. Accessed 10 Dec. 2024.