***Music genre classification: a comparison between machine learning classifiers***

# Introduction

Classifying music tracks into genres is crucial for music information retrieval tasks and an essential input for recommendation systems in streaming services. Music genres group tracks based on shared characteristics, aiding in efficient music organisation and enhancing user experience. According to IFPI (2024), the music industry saw a 10.2% revenue growth in 2024, with streaming services contributing over 67%. Accurate genre identification helps streaming services refine recommendations, driving user engagement and financial growth.

Categorising music is complex due to the constant evolution of styles, genre overlap, and the subjective nature of genres. Early successful attempts include Tzanetakis and Cook (2002), who used KNN and GMM classifiers for genre identification. Recent studies have employed methods like SVM (Lidy & Rauber, 2005) and deep learning techniques (Choi et al., 2017) to improve classification performance. However, these studies often focus on complex audio feature extraction, which is beyond this project's scope.

Instead, this project uses the Spotify dataset with pre-calculated features for training the models. The project aims to answer:

1. Which machine learning algorithm is the most effective in differentiating genres?

2. Which features in the Spotify dataset are most meaningful for genre classification?

# Dataset Overview

## About the dataset

The dataset, obtained from Kaggle, was compiled and cleaned using the Spotify Web API and Python. The original data consists of 89,000 tracks and multiple columns (attributes) that can be divided into three groups:

* Track Identification: Columns describing the track, including 'track\_name', 'artists', 'album\_name', and 'track\_genre'.
* Track Discrete Attributes: Categorical or discrete features extracted by Spotify, such as 'explicit', 'key', 'mode', and 'time\_signature'.
* Track Continuous Attributes: Numeric, continuous features extracted by Spotify, including 'popularity', 'duration\_ms', 'danceability', 'energy', and others.

The detailed schema can be seen in Table 1.

Table . Spotify dataset schema.

|  |  |
| --- | --- |
| **Column name** | **Description** |
| artists | The name(s) of the artist(s) associated with the track. (String) |
| album\_name | The name of the album containing the track. (String) |
| track\_name | The name of the track. (String) |
| popularity | The popularity score of the track on Spotify, ranging from 0 to 100. (Integer) |
| duration\_ms | The duration of the track in milliseconds. (Integer) |
| explicit | A boolean value indicating whether the track contains explicit content. (Boolean) |
| danceability | A score ranging from 0 to 1 that represents how suitable a track is for dancing based on various musical elements. (Float) |
| energy | A measure of the intensity and activity of a track, ranging from 0 to 1. (Float) |
| key | The key of the track represented by an integer value. (Integer) |
| loudness | The loudness of the track in decibels (dB). (Float) |
| mode | The tonal mode of the track, represented by an integer value (0 for minor, 1 for major). (Integer) |
| speechiness | A score ranging from 0 to 1 that represents the presence of spoken words in a track. (Float) |
| acousticness | A score ranging from 0 to 1 that represents the extent to which a track possesses an acoustic quality. (Float) |
| instrumentalness | A score ranging from 0 to 1 that represents the likelihood of a track being instrumental. (Float) |
| liveness | A score ranging from 0 to 1 that represents the presence of an audience during the recording or performance of a track. (Float) |
| valence | A score ranging from 0 to 1 that represents the musical positiveness conveyed by a track. (Float) |
| tempo | The tempo of the track in beats per minute (BPM). (Float) |
| time\_signature | The number of beats within each bar of the track. (Integer) |
| track\_genre | The genre of the track. (String) |

Following the classification structure proposed by Tzanetakis and Cook (2002) in Figure 1, the dataset categorizes music into 10 main genres, such as 'classical', 'country', and 'disco', with each genre having 1,000 sampled tracks.

A diagram of a music chart

Description automatically generated with medium confidence

Figure 1. Audio classification hierarchy suggested by Tzanetakis and Cook (2002)

## Exploratory Data Analysis

To understand the dataset's characteristics, an Exploratory Data Analysis (EDA) was conducted on the filtered dataset.

Figure 2 shows histograms of discrete features. The explicit content feature reveals that most tracks are non-explicit. The key feature displays a relatively even distribution, with keys 7 and 0 being the most common. Tracks are predominantly in the major key (mode 1), and the standard 4/4-time signature is the most frequent.

A graph of different sizes and colors

Description automatically generated with medium confidence

Figure 2. Histograms for discrete features.

A group of graphs showing different colors

Description automatically generated

Figure 3. Density plots for continuous features by labeled genres.

Figure 3 presents density plots of continuous features by genre. Popularity scores vary, with pop and hip-hop being more popular. Track durations are consistent across genres. Danceability is higher in dance and pop genres, while classical and rock are lower. Energy levels are higher in metal and disco. Loudness follows expected patterns, with rock and metal being louder. Speechiness is very low across genres, acousticness in classical and jazz, and instrumentalness in classical. Valence scores, indicating musical positiveness, are higher in pop and reggae, while tempo varies with faster tempos in dance and electronic genres.

Based on these visualisations, the feature importance can be intuitively inferred. Features like instrumentalness and speechiness, which exhibit a more uniform distribution across genres, are likely to contribute less to classification tasks. In contrast, features such as danceability and energy, which show distinctive differences across genres, are expected to be more significant in differentiating between genres.

# Methodology

Table . Detail modelling steps in genre classification.

|  |  |
| --- | --- |
| **Step** | **Description** |
| Data preparation | Data cleaning and splitting in to feature set and labelled set |
| Perform EDA |
| Data splitting into training and test sets. |
| Feature selection | Perform feature importance using default decision tree. |
| Perform dimensionality - accuracy analysis. |
| Perform dimension reduction if required. |
| Model selection and tuning | Set up pipelines and parameter grid for each algorithm, including K nearest neighbours (KNN), Naïve Bayes, support vector machine (SVM), decision tree, random forest and logistic regression |
| Perform hyperparameter grid search. |
| Perform 10-fold cross validation. |
| Model evaluation | Train best models with best parameters. |
| Evaluate accuracy and F1 score of classified genres. |
| Result interpretation | Overall interpretation of the model and results, including comments on the dataset. |
| Potential improvement points |

A general approach is outlined in Table 2.

## Problem statement and general approach

Working with this dataset requires a supervised and multi-class classification machine learning approach, specifically to categorise 10 classes from labelled data.

An array of classifiers, including K nearest neighbours (KNN), Naïve Bayes, support vector machine (SVM), decision tree, random forest and logistic regression, will be employed and compared to identify the most appropriate algorithm. The best performing algorithm will be used to train the model and its results used for analysis.

The features presented in the dataset have already formatted in numeric forms, however, due to difference is scales of the values, a Min Max Scaler will be applied to all the features before performing any machine learning algorithm (Hale, 2021). All the feature’s values will be scaled to be in between 0 and 1. This approach can be streamlined as a pipeline, illustrated in Figure 4.

A screenshot of a computer

Description automatically generated

Figure 4. A pipeline consists of a Min Max Scaler and a KNN Classifier.

## Feature selection

As observed in the EDA, some features might have a stronger impact on classification performance than others. With 15 numeric features, it is crucial to analyse which features are most needed and which might negatively impact the model.

Two methods are employed to achieve this:

1. Feature Importance from Decision Tree: The data is fitted to a default decision tree model (without tuning) to review feature importance. This method provides an initial insight into which features are considered most significant by the model.

2. Dimensionality Assessment: The data was fitted to three default models—decision tree, Gaussian Naive Bayes, and KNN—with a varied number of features. This approach assesses how the number of features affects model accuracy, helping to determine the optimal feature set for each model.

## Model Selection

When selecting models for genre classification, several key aspects are considered:

* Computational Efficiency: Efficient models are desirable as genre classification often serves as an input for larger, more complex systems like recommendation engines.
* Scalability: The model must handle growing music libraries efficiently, addressing both time constraints and computational resources:
  + Time Constraints: The classifier should operate quickly to avoid becoming a bottleneck in downstream processes.
  + Computational Resources: Frequent retraining is necessary due to the evolving nature of music. Models should balance performance with resource usage.
* Model Transparency: Interpretability is crucial for streaming services and music distributors to extract insights and make informed decisions. Simpler models like decision trees or Naive Bayes are preferred for their transparency over more complex models like SVMs or ensemble methods.

Considering the criteria above, the following models are the focus of this project:

* Naive Bayes: Provides moderate computational efficiency and simplicity, making it suitable for quick classification tasks.
* Decision Trees: Offers a good balance between interpretability and performance, providing clear insights into decision paths.
* Random Forest: Provides robust performance and handles large datasets well, although it requires more computational resources for training.

A graph with blue bars

Description automatically generated

Figure 5. Feature Importance from Default Decision Tree.

* K-Nearest Neighbours (KNN): Simple and effective for classification, suitable for smaller datasets and quick benchmarks.

The following algorithms are not being considered for this project, however still being used for benchmarking:

* Support Vector Machine (SVM): Known for its high accuracy, especially with non-linear data, but is computationally intensive and less interpretable.
* Logistic Regression: Provides a good baseline for comparison, offering simplicity and speed but potentially lower accuracy for complex datasets.

A graph with different colored lines

Description automatically generated

Figure 6. Dimensionality versus Accuracy of Various Classifiers.

## Model Tuning

The model selection process involves creating a pipeline for tuning and evaluating various machine learning models.

The pipeline begins with data preprocessing, where features are scaled using the MinMaxScaler to normalise the data into the 0 and 1 range. This ensures that all features contribute equally to the model training process and prevents any single feature from disproportionately influencing the results. Next, the data is fed into different classifiers for training and evaluation.

Grid Search with cross-validation (GridSearchCV) is employed to systematically explore and identify the optimal hyperparameters for each model. This method helps in assessing the model performance across different configurations, ensuring that the best combination of hyperparameters is selected based on accuracy scores. Since the sample counts are even between genres, accuracy scores are comparable to F1 scores.

## Evaluation Method

The model with the best performance is evaluated using 10-fold cross validation, using accuracy as the scoring benchmark.

A graph with black and red lines

Description automatically generated

Figure 7. Box plots comparing 10-fold CV results of 7 selected algorithms.

A math equations with black text

Description automatically generated

Figure 8. Accuracy, precision, recall and F1 score definitions.

# Result and Evaluation

## Feature Selection

The results for feature importance and dimensionality analysis, which presented in Figure 5 and Figure 6, shows that although explicit, mode and time\_signature features contribute little in the overall accuracy, they are still contributing to small increases of accuracy, with no negative impact. Therefore, all 15 features will be kept for further operations.

## Model Implementation

The GridSearchCV algorithm is deployed to search through the hyperparameters for each of the six algorithms presented in Table 3, and uses a 10-fold cross validation for model selection. The results, including the best parameters and the best accuracy score achieved using the X\_train set, are presented in Table 4.

Table 3. Tuning ranges for hyperparameters using GridSearchCV

A screenshot of a computer

Description automatically generated

The algorithms, with their best estimated parameters, are evaluated using 10-fold cross validation on the whole dataset and compared in Figure 7.

Based on these results, Random Forest is the best performing algorithm with consistently higher results. This can be explained by Random Forest ability to capture non-linear relationship effectively, especially with only 15 features and 10 classes. Moreover, the usage of multiple trees averages out the result and avoid overfitting.

KNN come second in the comparison. KNN, although performing considerably better in speed, both in training and testing, is hindered by the curse of dimensionality, especially when all the features are normalized into a specific range, making distances between vectors harder to differentiate.

Naïve Bayes algorithms score significantly lower in accuracy. This can be explained by the base assumption that all features are independent from each other, which rarely holds true, specifically in music data, where features are often interrelated.

Table . Best Hyperparameters by GridSearchCV

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Parameter** | **Best Value** | **Best CV score** | **Notes** |
| Gaussian Naive Bayes | var\_smoothing | 1.00E-06 | 0.39 |  |
| Multinomial Naive Bayes | alpha | 1.00E-05 | 0.4 |
| Support Vector Classifier | C | 100 | 0.6 | Extremely slow, often reaches 20 minutes. For reference only. |
| gamma | 1 |
| kernel | 'rbf' |
| Decision Tree | criterion | 'gini' | 0.53 |  |
| max\_depth | 20 |
| max\_features | 'log2' |
| min\_samples\_split | 2 |
| Random Forest | n\_estimator | 200 | 0.59 |  |
| max\_depth | 30 |
| max\_features | 'sqrt' |
| min\_samples\_split | 2 |
| Logistic Regression | C | 10 | 0.49 | Having multiple overflows for max iterations. For reference only. |
| solver | 'sag' |

A screenshot of a music chart

Description automatically generated

Figure 9. Training, testing results and confusion matrix for Random Forest.

## Model Evaluation

Random Forest is chosen for further evaluation. The training and testing results, in the form of confusion matrix, can be seen in Figure 9.

Overall, the accuracy of this model is 65.8%, with balanced Precision and Recall scores, which suggests a good starting point for a 10-class classification problem and implies room for improvements.

## Result Interpretation

The Random Forest model demonstrates a nuanced performance across different music genres, with classical music emerging as a standout in terms of model accuracy. With a recall of 0.93 and precision of 0.89, leading to an F1-score of 0.91, classical music's distinct features are effectively captured and recognized by the model, indicating a strong match between model capabilities and the genre's characteristics.

Conversely, genres such as blues experience lower precision, recall, and F1-scores, with frequent misclassifications between each other and with jazz, highlighting a difficulty in distinguishing between these similar categories. Jazz itself, while achieving a reasonable precision of 0.71, suffers from a lower recall of 0.64, suggesting the model often misses true instances of jazz. Other pairs exhibit the same confusion behaviour are pop – hip-hop and metal – rock.

This pattern of confusion across genres implies a need for enhanced feature engineering and model tuning to better differentiate between genres with overlapping musical elements.

# Conclusion

This project aimed to determine the most effective machine learning algorithm for music genre classification using a dataset from Spotify. The Random Forest algorithm emerged as the best performer, achieving the highest accuracy in differentiating between the ten music genres. The good performance can be attributed to its ability to handle non-linear relationships and reduce overfitting by averaging results across multiple trees. In contrast, algorithms like Naive Bayes underperformed due to their inherent assumption of feature independence, which does not align with the interrelated nature of music features.

The model's detailed evaluation revealed that certain genres, such as classical music, were classified with high precision and recall, indicating clear feature distinctions within the dataset. However, other genres, like blues and jazz, showed significant overlap, leading to frequent misclassifications. These results highlight the complexities of genre classification, where overlapping musical characteristics challenge even sophisticated models.

# Further Study

While the Random Forest model provides a strong foundation for music genre classification, there are rooms for further improvement.

The current model has difficulty differentiating between similar genres, such as blues - jazz or metal - rock, mainly due to the lack of control in feature extraction in this project.

For the limited scope of this project, only available, extracted features are used for training. Future implementation could focus on enhancing feature engineering to better capture the nuanced differences between acoustically similar genres. This could involve additional audio feature extractions or experimenting with NLP on lyrics classification. Advanced techniques, such as deep learning models, can also be employed to automatically learn feature representations from raw audio data.

Moreover, fine-tuning the model's hyperparameters further could potentially improve classification performance. Employing more sophisticated ensemble methods or integrating different types of classifiers might also yield better results.

Additionally, increasing the dataset size and diversity could help the model generalise better to newly updated data.

##### References

Choi, K., Fazekas, G., Sandler, M. and Cho, K. (2017) *Transfer learning for music classification and regression tasks* [online]. Available from: http://arxiv.org/abs/1703.09179 [Accessed 16 May 2024].

Géron, A. (2019) *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: concepts, tools, and techniques to build intelligent systems*. Second edition. Beijing [China] ; Sebastopol, CA: O’Reilly Media, Inc.

Hale, J. (2021) *Scale, Standardize, or Normalize with Scikit-Learn* *Medium*. 13 December 2021 [online]. Available from: https://towardsdatascience.com/scale-standardize-or-normalize-with-scikit-learn-6ccc7d176a02 [Accessed 16 May 2024].

IFPI (2024) *IFPI Global Music Report 2024* [online]. IFPI. Available from: https://ifpi-website-cms.s3.eu-west-2.amazonaws.com/GMR\_2024\_State\_of\_the\_Industry\_53c7eb914a.pdf [Accessed 14 May 2024].

Lidy, T. and Rauber, A. (2005) *Evaluation of Feature Extractors and Psycho-Acoustic Transformations for Music Genre Classification* [online]International Society for Music Information Retrieval Conference. Available from: https://www.semanticscholar.org/paper/Evaluation-of-Feature-Extractors-and-for-Music-Lidy-Rauber/bcd2eb19b1f764c2e2e17718a73e6c04cb4d45ee [Accessed 16 May 2024].

maharshipandya (2023a) *maharshipandya/spotify-tracks-dataset · Datasets at Hugging Face*. 2023 [online]. Available from: https://huggingface.co/datasets/maharshipandya/spotify-tracks-dataset [Accessed 16 May 2024].

maharshipandya (2023b) *Spotify Tracks Genre*. 2023 [online]. Available from: https://www.kaggle.com/datasets/thedevastator/spotify-tracks-genre-dataset [Accessed 16 May 2024].

sklearn (no date) *sklearn.ensemble.RandomForestClassifier* *scikit-learn* [online]. Available from: https://scikit-learn/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html [Accessed 16 May 2024a].

sklearn (no date) *sklearn.linear\_model.LogisticRegression* *scikit-learn* [online]. Available from: https://scikit-learn/stable/modules/generated/sklearn.linear\_model.LogisticRegression.html [Accessed 16 May 2024b].

sklearn (no date) *sklearn.model\_selection.GridSearchCV* *scikit-learn* [online]. Available from: https://scikit-learn/stable/modules/generated/sklearn.model\_selection.GridSearchCV.html [Accessed 16 May 2024c].

sklearn (no date) *sklearn.naive\_bayes.GaussianNB* *scikit-learn* [online]. Available from: https://scikit-learn/stable/modules/generated/sklearn.naive\_bayes.GaussianNB.html [Accessed 16 May 2024d].

sklearn (no date) *sklearn.naive\_bayes.MultinomialNB* *scikit-learn* [online]. Available from: https://scikit-learn/stable/modules/generated/sklearn.naive\_bayes.MultinomialNB.html [Accessed 16 May 2024e].

sklearn (no date) *sklearn.neighbors.KNeighborsClassifier* *scikit-learn* [online]. Available from: https://scikit-learn/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html [Accessed 16 May 2024f].

sklearn (no date) *sklearn.svm.SVC* *scikit-learn* [online]. Available from: https://scikit-learn/stable/modules/generated/sklearn.svm.SVC.html [Accessed 16 May 2024g].

sklearn (no date) *sklearn.tree.DecisionTreeClassifier* *scikit-learn* [online]. Available from: https://scikit-learn/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html [Accessed 16 May 2024h].

Tzanetakis, G. and Cook, P. (2002) Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing* [online]. 10 (5), pp. 293–302. Available from: https://ieeexplore.ieee.org/document/1021072 [Accessed 16 May 2024].