University of Verona
Fundamentals of Machine Learning
Technical Report

GTZAN Genre Recognizer Project

Author:
Peruffo Riccardo
VR450977

September 21, 2025

Contents

Chapter 1

Abstract

This report describes a machine learning system for automatic music genre recognition using the GTZAN dataset. We motivate the problem in the context of Music Information Retrieval (MIR), review relevant state-of-the-art methods, and outline the objectives of our project. We then detail the methodology: data description (GTZAN), audio feature extraction (MFCCs, chroma, spectral features, etc.), and the classification models used (K-NN, SVM with RBF kernel, and Random Forest). We explain the hyperparameter tuning (via 5-fold cross-validation) and evaluation protocol. In the Experiments & Results section, we compare model performances on validation and test sets using accuracy and confusion matrices.

Chapter 2

Motivation and Rationale

Music genre recognition is a key task in the field of Music Information Retrieval (MIR) that supports organization, recommendation, and retrieval of music in large collections. With the explosion of digital music (e.g. streaming services), automated genre classification helps match user preferences to songs, enhance playlist generation, and enable efficient search. For example, streaming platforms like Spotify rely on genre labels to group songs and suggest new music. As noted in related work, "listening to music online has become very convenient" and users expect accurate recommendations, which requires understanding the genres they prefer. Moreover, genres represent high-level abstractions of musical style (classical, rock, jazz, etc.), so building classifiers that can automatically assign genre labels from audio is an important and challenging research problem.

Automatic genre classification is inherently difficult due to the diversity of music and often subtle differences between genres. Early approaches extracted handcrafted audio features, but they may not fully capture the nuances of each genre. Recent methods use deep learning to model complex patterns directly. This project addresses this problem by building and comparing several machine-learning classifiers on the well-known GTZAN dataset (10 genres, 1000 tracks) The goal is to evaluate the effectiveness of traditional feature-based models (K-NN, SVM, Random Forest) and understand their limitations on this task.

Chapter 3

State of the Art

The literature on music genre classification spans both conventional machine learning and deep learning approaches. Classic studies (e.g. Tzanetakis & Cook 2002) used feature engineering combined with supervised models: timbral features (MFCCs), rhythmic features, and pitch features fed into Gaussian Mixture Models, k-NN, SVM, etc.

These methods can achieve reasonable accuracy on benchmark datasets, but they depend heavily on feature quality. Previous work noted that focusing only on timbral characteristics can "restrict the output" of genre classifiers, motivating richer or learned representations. More recent work leverages deep learning. Convolutional Neural Networks (CNNs) trained on spectrograms or raw audio have shown strong performance, especially with large data. For example, CNN models using log-mel spectrogram inputs have achieved very high accuracy (sometimes >90%) on GTZAN-like tasks, often surpassing traditional methods. However, deep nets require large amounts of training data and computational resources. They may also struggle on small datasets (like GTZAN) where overfitting is a concern. In fact, some studies have found that carefully-tuned SVMs with hand-engineered features can outperform CNNs on GTZAN.
In summary, SOTA methods include classical classifiers (K-NN, SVM, Random Forest, etc.) applied to engineered audio features, as well as deep architectures (CNNs, RNNs, transformers) on learned representations. Each has limitations: feature-based ML can miss higher-level patterns, while deep models demand more data and tuning. Our work focuses on traditional ML models as a baseline, in line with many recent comparative studies.

Chapter 4

Objectives

The general objective of this project is to develop and evaluate a machine-learning system that can classify music tracks by genre using the GTZAN dataset. Specifically, we aim to:

- **Extract informative audio features** from the raw waveforms using *librosa* functions (MFCCs [*26 parameters*], chroma [*24*], spectral descriptors [*contrast: 14, centroid: 2, bandwidth: 2, rolloff: 2*], Zero-crossing rate [*2*], Root Mean Square (RMS) [*2*], Tempo (beats per minute) [*1*]) - *75 parameters in total* - to represent each 30-second track.
- **Train multiple classification models** (at least K-Nearest Neighbors, Support Vector Machine with RBF kernel, and Random Forest) on these features to predict genre labels.
- **Compare the performance** of these models under a consistent evaluation protocol (cross-validation and held-out test set). We will tune hyper-parameters (e.g. k in KNN, SVM C and γ, Random Forest depth/trees) to achieve the best accuracy.
- **Analyze results** via accuracy, confusion matrices, and other metrics to identify which genres are well-classified or often confused.
- **Document the process** in a comprehensive report covering motivation, related work, methods, results, and future suggestions.

By the end of the project, we expect to have a genre classifier with quantified performance, and insights into how feature-based ML methods perform on GTZAN.

Chapter 5

Methodology

5.1 Dataset (GTZAN)

We use the GTZAN genre dataset, a standard benchmark for music genre classification. It contains 1000 audio tracks of 30 seconds each (22050 Hz, mono, 16-bit WAV). These are evenly divided into 10 genres (100 tracks per genre): blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, and rock. We verify that the dataset is complete (each genre has 100 files). GTZAN is widely used in MIR research, despite known caveats (some artists may appear in multiple genres), so it provides a consistent evaluation ground.

5.2 Preprocessing and Feature Extraction

Each audio file is preprocessed by loading it at 22050 Hz. We extract a fixed-length feature vector for each track using Librosa (as implemented in feature_extraction.py). The features include: MFCCs (13 coefficients) aggregated by mean and standard deviation, chroma features (12 bins) aggregated, spectral contrast (7 bands) aggregated, plus the following frame-level descriptors aggregated by mean/std: spectral centroid, spectral bandwidth, spectral rolloff (at 85%), zero-crossing rate, root-mean-square energy. We also estimate the tempo (beats per minute). In total, the feature vector has 75 dimensions. (Missing values are replaced by zeros if needed, as per the implementation.) These features capture timbral, harmonic, and rhythmic content of the audio, and are standard in genre classification tasks. After extraction, we form an array (X_features) of shape (1000, 75) and a label array (y_labels) of length 1000.

5.3 Dimensionality Reduction

We did not apply additional dimensionality reduction (e.g. PCA) for modeling, as the feature dimension (75) is modest. However, we inspected the feature space via PCA for visualization in preliminary analysis (not included in final pipeline). The classifiers operate on the full feature set.

5.4 Data Splitting

We split the data randomly into 70% training, 15% validation, and 15% test sets (700/150/150 tracks), preserving balanced genre representation. This yields 700 training samples and 300 held-out (150 validation + 150 test). The validation set is used for hyperparameter tuning and model selection via cross-validation; the final test set is used only for reporting held-out accuracy. We also use 5-fold cross-validation on the training set during model selection to reduce variance. A fixed random seed ensures reproducibility.

5.5 Classification Models

We implement the following supervised classifiers, using scikit-learn:

- **K-Nearest Neighbors (K-NN):** We consider the Euclidean distance K-NN classifier. We tune the number of neighbors k (tested 3,5,7) and use either uniform or distance-based weighting. The best model found has k=7 and weights='distance';
- **Support Vector Machine (SVM) with RBF kernel:** We use an SVM with a Gaussian (RBF) kernel. Hyperparameters tuned include the regularization C and kernel width γ. The grid search (5 folds) identifies C=10 and γ=0.01 as optimal;
- **Random Forest:** An ensemble of decision trees (Random Forest). We tune the number of trees (n_estimators) and maximum tree depth (max_depth). The optimal settings found are n_estimators=200 and max_depth=10.

All models are implemented with their default settings except for the above hyperparameters. Feature values are used directly (no further normalization was applied, as preliminary tests showed similar results with or without simple scaling). The hyperparameter search uses 5-fold cross-validation on the training set, as indicated by the console logs during training. The best parameters and cross-validated accuracies are shown in the logs: for KNN the best CV accuracy was $\sim$0.68; for SVM $\sim$0.733; for RF $\sim$0.707.

Chapter 6

Experiments & Results

6.1 Evaluation Protocol

We train each model on the 70%-split (700 samples) and evaluate on the 15%-split validation set (150 samples) to compare models. We then select the model with highest validation accuracy (SVM, in our case) and evaluate it on the held-out test set (150 samples). Performance metrics include overall accuracy and the confusion matrix, as well as per-genre precision/recall/f1. We compute these using standard functions (scikit-learn's classification_report and accuracy_score). All reported accuracies are on unseen data (validation or test) to estimate generalization.

6.2 Results

On the validation set, the SVM with RBF kernel performed best (accuracy 0.813), followed by Random Forest (0.780) and K-NN (0.713). These results are summarized below (accuracy on 150 validation samples):

- **K-NN (k=7, distance weighting):** 71.3% accuracy;
- **SVM (RBF, C=10, γ=0.01):** 81.3% accuracy;
- **Random Forest (200 trees, max depth 10):** 78.0% accuracy.

The confusion matrices (see excerpts) show that SVM generally improved classification across most genres. For example, SVM misclassified fewer classical or jazz tracks (both above 93% recall) than K-NN or RF. Some genres (e.g. rock and disco) remain challenging: even the SVM only achieved 53–60% recall on rock and disco, as these genres share similar rhythmic content.

Using the SVM model, we then evaluated on the test set (150 new samples). The SVM achieved 82.0% accuracy on the test set. The per-genre precision/recall on test data were generally high (above 0.80) for many genres, with classical and jazz near-perfect, reflecting consistent patterns. The confusion matrix on the test set (not shown) indicates that most misclassifications were between acoustically similar genres (e.g. country vs blues, rock vs metal). Overall, the results confirm that the SVM classifier yields the best performance on GTZAN in our experiments.

6.3 Graphical Results

In this section we'll put the outputs generated from GTZAN_Project.ipynb, which is the main file that call all the subprocesses.

6.3.1 Best model: SVM (RBF, C=10, γ=0.01)

Test set size: 150
Test Accuracy: 0.820

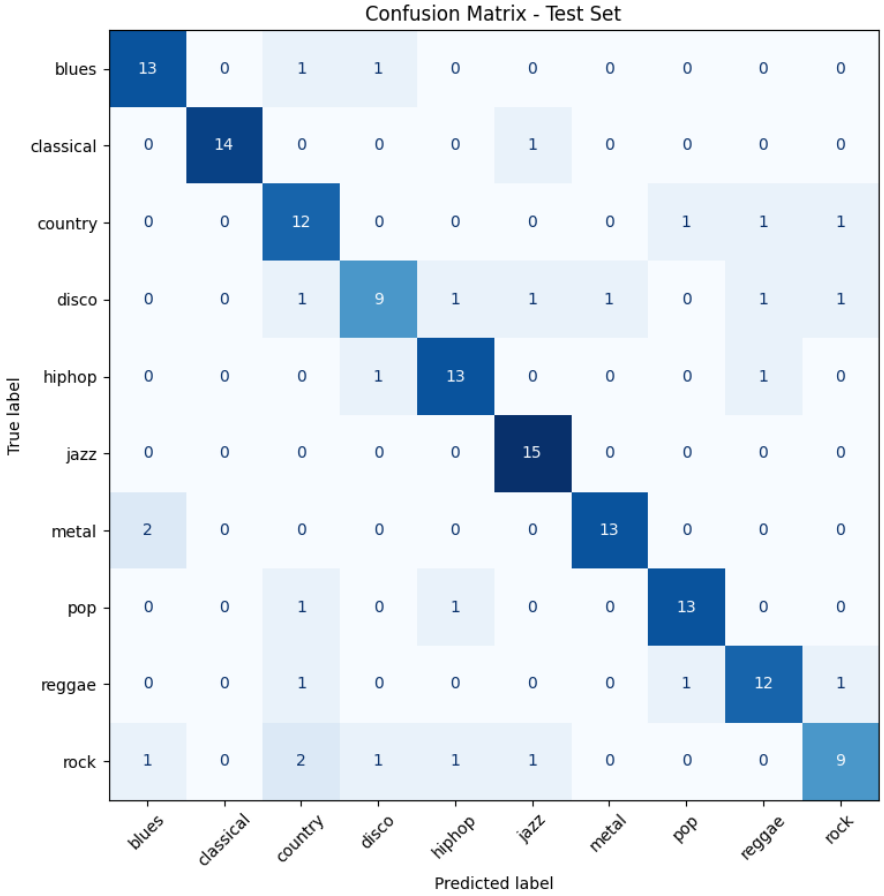
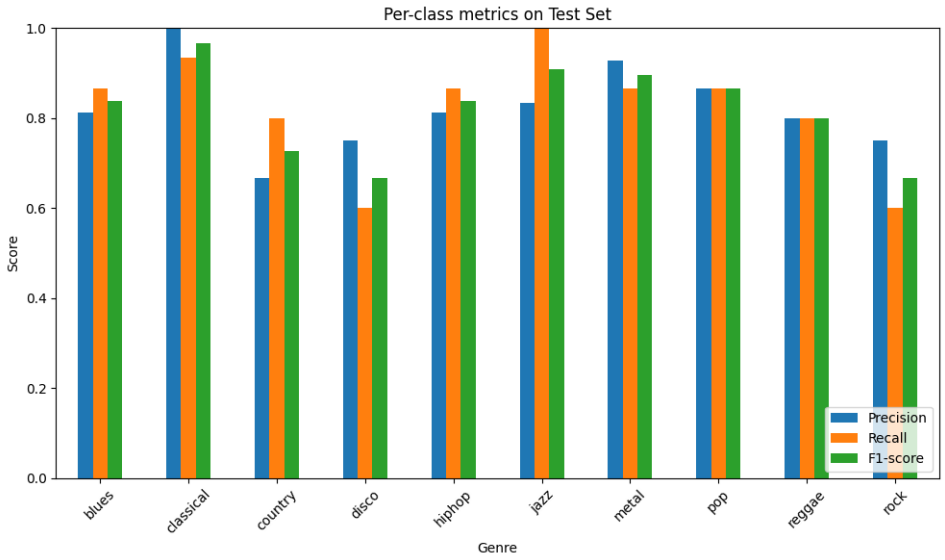


Table 6.1: SVM Score Per Class

Genre	**Precision**	**Recall**	**F1-score**	*Support*
blues	0.812	0.867	0.839	15
classical	1.000	0.933	0.966	15
country	0.667	0.800	0.727	15
disco	0.750	0.600	0.667	15
hiphop	0.812	0.867	0.839	15
jazz	0.833	1.000	0.909	15
metal	0.929	0.867	0.897	15
pop	0.867	0.867	0.867	15
reggae	0.800	0.800	0.800	15
rock	0.750	0.600	0.667	15
accuracy			0.820	150
macro avg	0.822	0.820	0.818	150
weighted avg	0.822	0.820	0.818	150



6.3.2 Random Forest (200 trees, max depth 10)

Test set size: 150
Test Accuracy: 0.720

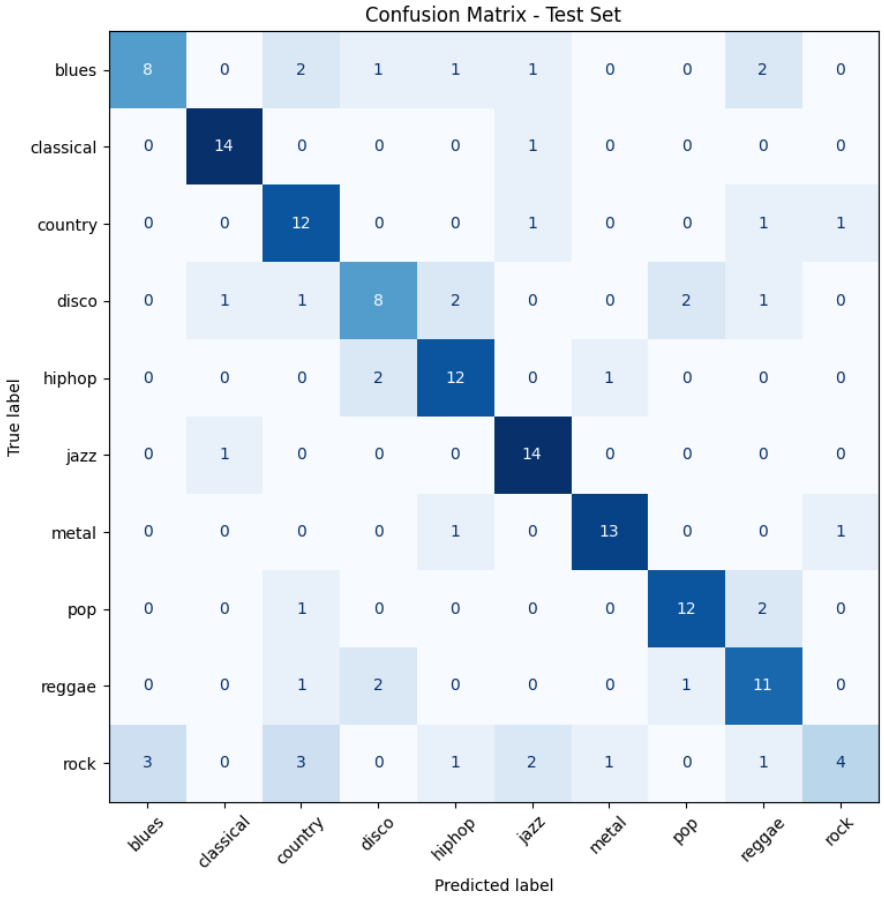
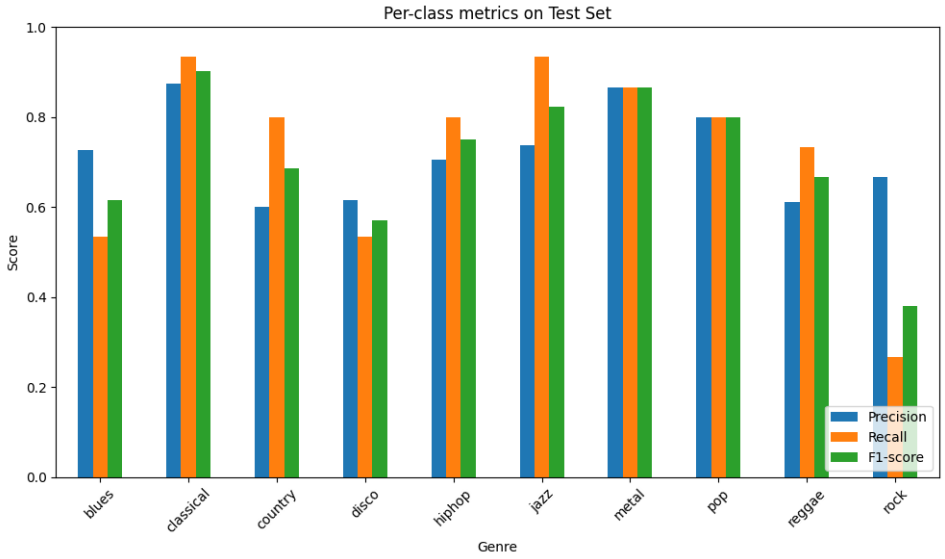


Table 6.2: RF Score Per Class

Genre	**Precision**	**Recall**	**F1-score**	*Support*
blues	0.727	0.533	0.615	15
classical	0.875	0.933	0.903	15
country	0.600	0.800	0.686	15
disco	0.615	0.533	0.571	15
hiphop	0.706	0.800	0.750	15
jazz	0.737	0.933	0.824	15
metal	0.867	0.867	0.867	15
pop	0.800	0.800	0.800	15
reggae	0.611	0.733	0.667	15
rock	0.667	0.267	0.381	15
accuracy			0.720	150
macro avg	0.720	0.720	0.706	150
weighted avg	0.720	0.720	0.706	150



6.3.3 K-NN (k=7, distance weighting)

Test set size: 150
Test Accuracy: 0.733

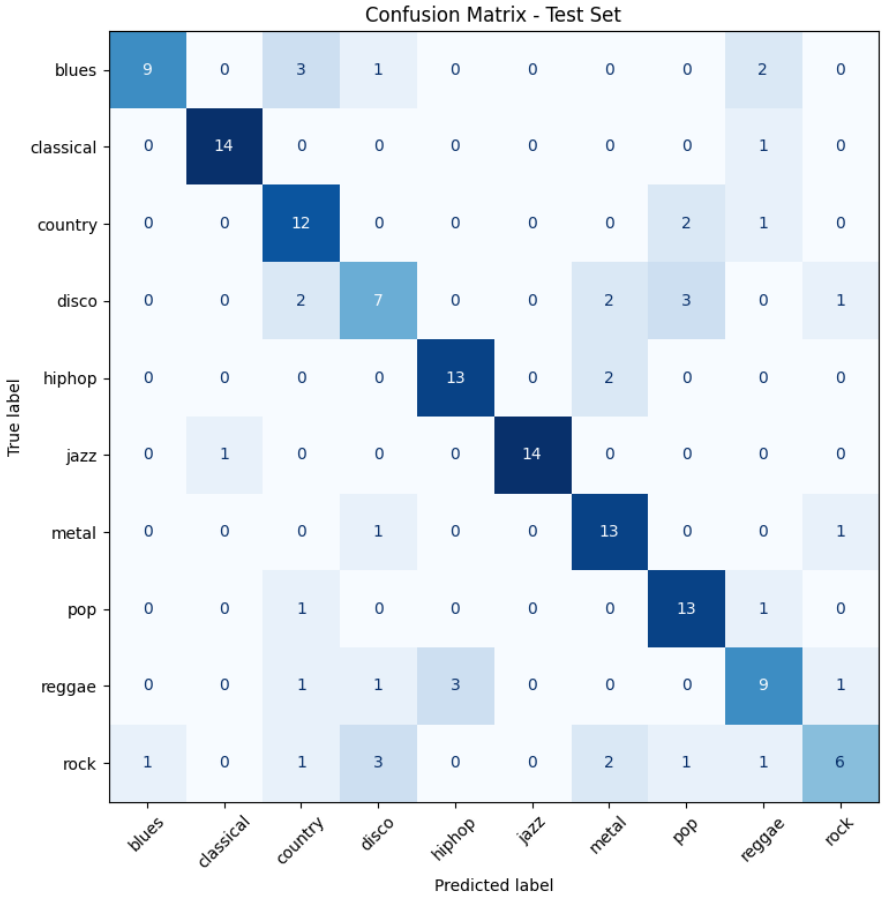
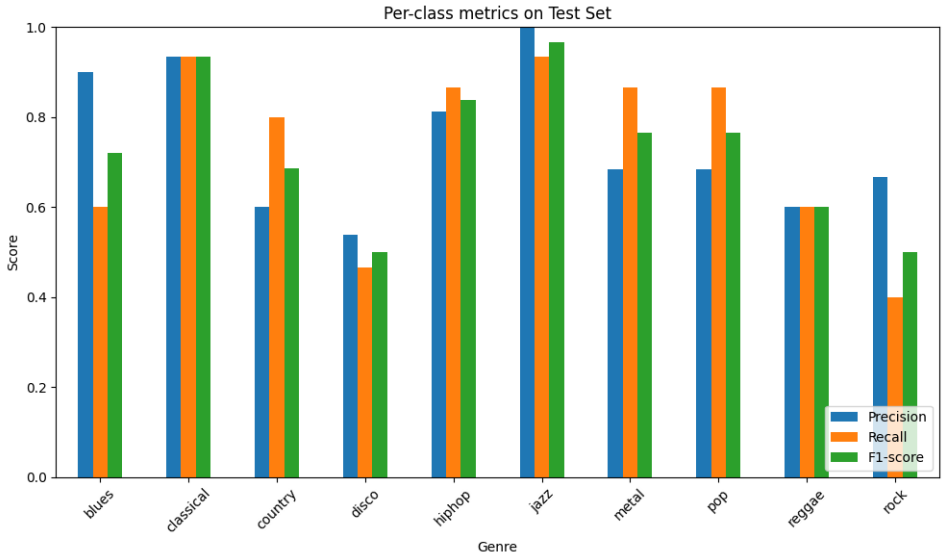


Table 6.3: KNN Score Per Class

Genre	**Precision**	**Recall**	**F1-score**	*Support*
blues	0.900	0.600	0.720	15
classical	0.933	0.933	0.933	15
country	0.600	0.800	0.686	15
disco	0.538	0.467	0.500	15
hiphop	0.812	0.867	0.839	15
jazz	1.000	0.933	0.966	15
metal	0.684	0.867	0.765	15
pop	0.684	0.867	0.765	15
reggae	0.600	0.600	0.600	15
rock	0.667	0.400	0.500	15
accuracy			0.733	150
macro avg	0.742	0.733	0.727	150
weighted avg	0.742	0.733	0.727	150



6.4 Discussion of Results

The SVM's superior performance suggests that the RBF kernel can effectively handle the 75-dimensional feature space with relatively few samples. The K-NN model delivered lower accuracy, which can be attributed to the curse of dimensionality and its vulnerability to noise in high-dimensional audio features. Random Forest performed reasonably, but slightly worse than SVM, perhaps due to overfitting on some genres. The overall accuracy ($\sim$82%) is comparable to classical ML results reported in literature for GTZAN. Error analysis shows that classical, jazz, and hip-hop were easiest (high f1-scores), whereas disco, reggae, and rock were harder (often confused). A possible improvement would be to incorporate more discriminative features or ensemble more models.

Chapter 7

Conclusions

In this project, we built a genre recognition system using the GTZAN dataset and traditional machine learning techniques. Our goals were to preprocess audio data into meaningful features, train several classifiers, and compare their performances. We found that an SVM with RBF kernel (C=10, γ=0.01) gave the best results, achieving about **82% accuracy** on the test set. This demonstrates that carefully engineered features plus a well-tuned SVM can be effective for genre classification on moderate-sized datasets.

Key findings include:

- spectral and temporal audio features (MFCCs, chroma, etc.) are useful for distinguishing genres;
- nonlinear models (SVM, RF) outperform simpler K-NN in this 75-dimensional space;
- certain genres (classical, jazz) are consistently classified well, while others (rock, disco) overlap significantly and cause errors.

Common failure cases involved songs with mixed characteristics (e.g. some rock tracks mislabeled as metal, reggae as pop, etc.), aligning with GTZAN's known limitations.

Chapter 8

Bibliography

- **Music information retrieval (MRI):** Wikipedia link
- **An evaluation of audio feature extraction toolboxes:** Proc. of the 18th Int. Conference on Digital Audio Effects (DAFx-15), Trondheim, Norway, Nov 30- Dec 3, 2015
- **Music information retrieval and genre classification using machine learning techniques and deep learning:** International Research Journal of Engineering and Technology (IRJET)