Music Genre Classification using Machine Learning methods

Katerina Maxouri

Project 2020-21
Department of Electrical and Computer Engineering
University of Thessaly
amaxouri@inf.uth.gr

Abstract This project aims to compare machine learning methods based on their ability to automatically recognize a song's genre. Various classification algorithms, along with feature engineering techniques, were tested on a dataset containing audio tracks. Results indicate that Support Vector Machines and Linear Discriminant Analysis classifiers are the most suitable for this task. Furthermore, dimensionality reduction of features, using Linear Discriminant Analysis enhanced all algorithms' performances. Algorithms are evaluated on their accuracy, average precision, recall as well as f1-score of each class.

1 Introduction

Due to development of computer networks, more and more people tend to purchase and download digital music from the Internet or use music streaming services such as Spotify and Apple Music. Managing music databases may be challenging, based on the fact that a typical database usually contains millions of tracks. Therefore, it is required to organize the music content by categories like genre, artist and album. A study conducted by Perrot and Gjerdingen, showed that humans can achieve about 70% accuracy in recognizing music genre of a track among ten total genres [1]. In another study, humans were also tested in recognizing genre among six total genres, and they achieved an inter-participant agreement rate of 76% [2]. The results of both studies indicate that humans specify genre subjectively, meaning that genre classification by humans can not provide accurate results. Thus, automatic music genre classification has become a popular topic of research.

In this project, different supervised classification techniques were tested and compared for genre classification. Supervised machine learning uses labeled data sets in order to train algorithms to classify data or predict outcomes [3]. The process of classification music tracks can be divided in two processes. The first one is the analysis of the signals, usually that is extraction of signal features. such as spectral features. The second one is the multi-class classification based on the computational analysis of the features extracted. The purpose of this project is to implement various classification techniques and compare their results, so as to develop an effective and accurate model for music genre recognition.

2 Related Work

Extraction of feature vectors is fundamental for any automatic audio analysis system. Usually they are based on time-frequency representation of the signals. Mel-frequency cepstral coefficients (MFCC), a set of features that are extensively used in speech recognition [4], are also included in features for genre classification. MFCCs belong to spectral features, that contain information about bandwidth, spectral energy etc. [5].

Tzanetakis and Cook [6] use three feature sets for representing timbral texture, rhythm, and pitch content. Using the proposed feature sets, they achieved a classification of 61% accuracy for ten total musical genres. MFCCs were also used in [7], along with modelling of the temporal variation of features, resulting 82.79% classification accuracy. Temporal modelling can be performed by calculating short-time means and variances of each dimension of every frame's features. In [8] both timbral features, such as spectral information and rhythmic content features, such as beat and tempo, are considered, adding that inter-genre similarity is modelled so as to classify hard-to-classify samples, resulting 92.40% accuracy score. Grimaldi, Cunnigham and Kokaram applied in [9] wavelet transformation to obtain the representation of signal at different levels, followed by time and frequency features extraction from these levels and 78.5% accuracy was achieved. Panagakis *et al.* [10], extracted multi-scale spectro-temporal modulation features and propose a multi-linear approach. They tested on two different datasets obtaining 78.20% and 80.95% accuracy scores.

3 Data and Features extracted

For the classification task, the popular GTZAN genre collection dataset was used. The dataset consists of 1000 audio tracks each 30 seconds long. It contains 10 total genres, each represented by 100 tracks. The genres included are the following:

- blues
- classical
- country
- disco
- hip-hop
- jazz
- metal
- pop
- reggae
- rock

Both types of features were used in representing the music content, temporal and spectral features. Temporal are time domain features, such as energy of signal, zero-crossing rate e.t.c. Spectral features are frequency based, obtained by applying Fourier transform, like spectral centroid, spectral roll-off, rhythm and so on.

More specifically, librosa python library [11] was used for feature extraction. Librosa implements various spectral representations, that is the distributions of energy over a set of frequencies. Librosa provides functions for computing spectral features per frame,

so each feature's mean and standard deviation is calculated, adding that it provides a function which estimates the tempo (BPM).

The features extracted are the following:

- chromagram
- constant-Q chromagram
- chroma variant "Chroma Energy Normalized" (CENS)
- mel-scaled spectogram
- MFCCs (the 20 first)
- root mean square (RMS) from the audio samples
- spectral centroid
- spectral bandwidth
- spectral contrast
- spectral flatness
- roll-off frequency
- tonal centroid features (tonnetz)
- zero-crossing rate
- beats per minute (BPM), also known as tempo

4 Methods

The methods used can be divided in two categories. The feature selection and dimensionality reduction techniques, as well as the classifiers, that were considered.

4.1 Feature Engineering

Exclusion of highly correlated features A common practice in machine learning is to exclude highly correlated features. Correlation means a mutual relationship between two or more variables/features and the correlation coefficient specifies if large and small values of one variable are paired with large and small values of the other one, respectively.

$$CorrelationCoefficient = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

If the absolute value of the correlation coefficient is above 0.7, it means that features are highly correlated, thus one of them can be excluded. In this project, when exclusion was performed, the threshold chosen was 0.85. The correlation matrix of all features is shown in figure 1.

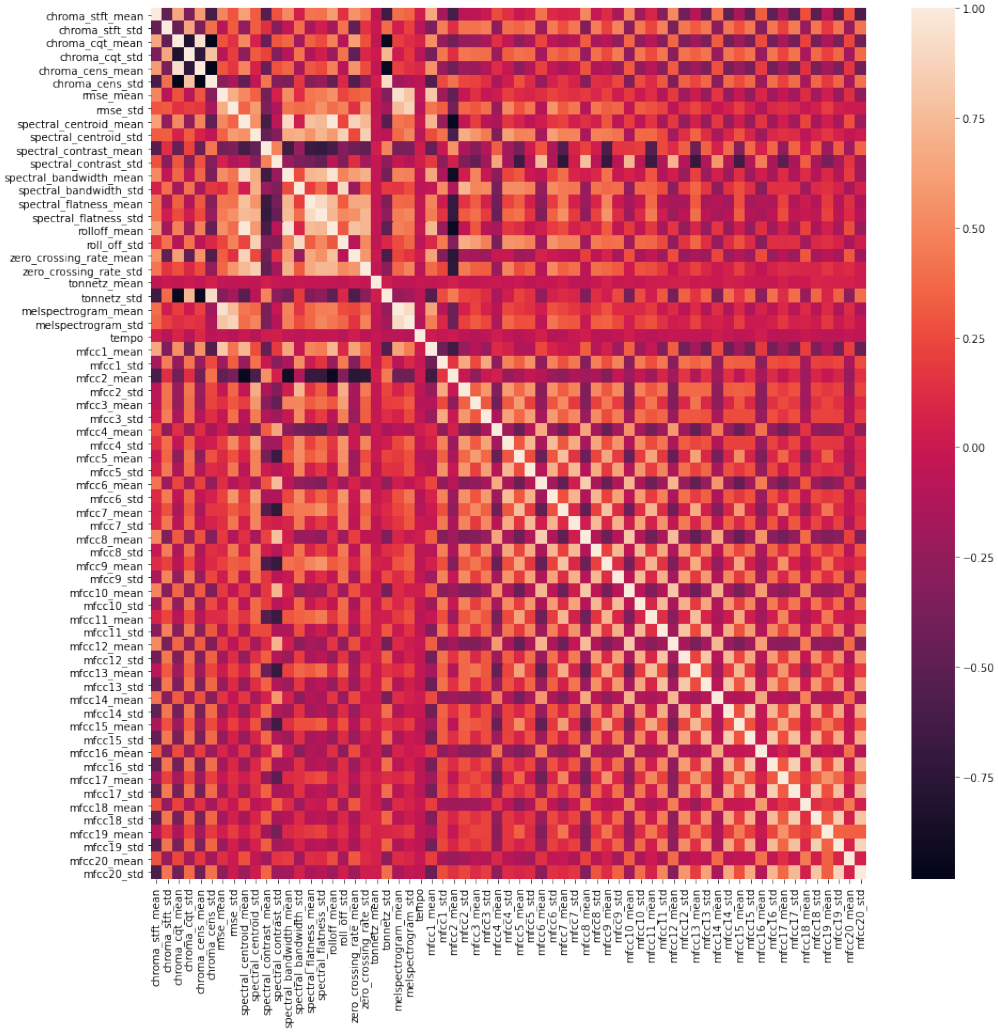

Fig. 1. The correlation matrix of all features extracted.

Removing features with low variance Features of low variance or variability do not provide much information to a machine learning model, on the grounds that their values are practically constant. Therefore, they can be removed. The variance threshold chosen was 1.

Univariate feature selection Univariate feature selection works by selecting those features with the best univariate statistical tests. The SelectKBest method,from Scikit-learn [12], was employed for transforming features. With this method were chosen 50, 40, 30, 20, and finally 10 best features, in order to test classification algorithms.

Recursive Feature Elimination (RFE) Provided an external estimator, RFE assigns weights to features. The purpose is to select features by considering recursively smaller and smaller sets of features. To begin with, the estimator is trained on the initial set and the importance of features is obtained. The least essential features are pruned from the current set and the procedure is repeated recursively on the pruned set. Once the chosen number of features is reached, the procedure stops. Sets of 50, 40, 30, 20 and 10 total features were considered in the project.

Feature selection using SelectFromModel SelectFromModel is a meta-transformer by Sklearn that can be used along with any estimator that provides importance of features. Provided a threshold or using built-in heuristics for finding a threshold, it keeps the important features and removes the rest.

Principal Component Analysis (PCA) Principal Component Analysis is based on projection methods and it is mostly used for representing multivariate data tables as smaller set of variables. PCA allows analysis of datasets that are high dimensional or may contain missing values, categorical data etc. The aim is to extract essential information from data and summarize them in a set of indices called principal components.

Linear Discriminant Analysis (LDA) The goal of Linear Discriminant Analysis is to find a new space to project data in order to maximize the classes separability. LDA can be used as a dimensionality reduction algorithm to transform a set of features.

4.2 Algorithms

Gaussian Naive Bayes The first algorithm considered is Gaussian Naive Bayes, belonging to Naive Bayes classifiers. Naive Bayes classifiers are based on Bayes' Theorem, which make the assumption that features are independent of each other. Bayes' Theorem calculates the probability of an event occurring, given the probability of another event which has occurred already and its formula is the following:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

When dealing with a dataset the theorem is applied in the following way:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)} \Rightarrow P(y|x_1, \ldots, x_n) = \frac{P(x_1|y)P(x_2|y)\cdots P(x_n|y)P(y)}{P(x_1)P(x_2)\cdots P(x_n)}$$

where y is the target class and $X = (x_1, x_2, \ldots, x_n)$ is a feature vector.

In Gaussian Naive Bayes, continuous values associated with each feature are assumed to follow a Normal distribution $N(\mu, \sigma^2)$. The likelihood of the features is assumed to be Gaussian, as a consequence conditional probability is given by:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Stochastic Gradient Descent (SGD) In Stochastic Gradient Descent, a single sample of the dataset is selected for each iteration, instead of the entire dataset. The sample is randomly selected and the gradient of its cost function is calculated. The aim of SGD classifier is to minimize the cost function that penalizes two or more classes.

The SGD algorithm is the following:

for i in range (m):

$\theta_j = \theta_j - \alpha(\hat{y}^i - y^i)x_j^i$

K-Nearest Neighbors (KNN) K-Nearest Neighbors algorithm makes the assumption that similar data exist in close proximity. To begin with, in KNN the value of K, that is the number of nearest neighbors, must be chosen. Then, the distance between the query-instance and all the training samples is calculated. The distances are sorted, adding that K nearest neighbors are determined. In KNN classifier, the predicted result is the mode of nearest neighbors' labels.

Decision Tree A decision tree is a tree structure, where every internal node represents a feature (attribute), the branch represents a decision rule, while each leaf represents the outcome. The root node partitions data on the basis of the attribute value. The best attribute to split data is selected using Attribute Selection Measures (ASM). The most popular selection measures are the following:

- Information Gain, which measures the impurity of the input set
- Gain Ratio, which prefers the attribute with many outcomes
- Gini Index, which considers a binary split for each attribute

Once the attribute is determined, it becomes a decision node and divides the dataset into subsets. The tree is built by repeating this process recursively for every child node.

Random Forest Random Forest is a method based on the "divide and conquer" approach. It is an ensemble method of decision trees generated on a randomly split dataset. The set of decision tree classifiers is also known as forest. Each tree is generated using an attribute selection indicator and depends on an independent random sample. For classification tasks, all outcomes from the decision trees are collected and the mode is the final result of the classifier.

Support Vector Machines (SVM) Support Vector Machines classifier is based on linear separation. SVM aims to find a hyper-plane that maximizes the separation of data points to their classes in a n-dimensional space. The points closest to the hyper-plane are called Support Vectors. Multi-class classification is basically converted into multiple binary classification tasks. Data points are mapped to high dimensional space, so as to gain mutual linear separation between every two classes.

Neural Networks Neural Networks (NN) are series of algorithms inspired by the structure and function of human brain. Basically, a NN architecture mimics the way a human brain operates in processing data, learning patterns and making decisions. In this project there were tested three different NN architectures.

1. The first architecture considered consists of 5 Dense layers. The activation function ReLu was used for the 4 first layer and softmax for the output layer. The model was compiled with adam optimizer and sparse categorical crossentropy and was trained for 100 epochs with a batch size of 64.
2. Furthermore, a combination of convolutional and recurrent NN was tested. The model consists of 4 total layers, one Convolutional (1D) followed by MaxPooling, one LSTM layer and finally a Dense output layer (with softmax activation function). The model was compiled with RMSprop optimizer and sparse categorical crossentropy as a loss function. It was trained for 300 epochs with a batch size of 64.
3. The last NN is Scikit-learn's [12]implementation of Multilayer Perceptron classifier.

XGBoost In this project, was also employed XGBoost, an optimized distributed gradient boosting library that provides efficiency and flexibility [13]. It implements machine learning algorithms and provides a parallel tree boosting that solves various problems accurately and fast.

Linear Discriminant Analysis (LDA) Linear Discriminant Analysis is not only used for dimensionality reduction, but also for multiclass classification tasks. The representation of LDA consists of statistical properties of data calculated for each class. For example, the mean and variance for single a input variable or the mean and variance calculated over the multivariate Gaussian for multiple variables. These properties are determined by data and used for predicting classes. LDA makes predictions by estimating the probability of a set of inputs belonging to each class.

For the implementation of the algorithms mentioned above, python libraries Scikit-learn [12], XGBoost [13] and Keras [14] were employed.

5 Results

The classification results of the algorithms are presented in the following tables 1 - 22. In K best features there are 2 accuracy scores, because SelectKBest algorithm was employed with two different score functions, the default *f_classif* and *mutual_info_classif*.

5.1 Naive Bayes

All accuracy scores for Naive Bayes classifier are displayed in table 1.

All features	0.49		
Without highly correlated features	0.46		
Without low variance features	0.49		
K Best features	$K = 50$	0.54	0.52
	$K = 40$	0.52	0.51
	$K = 30$	0.52	0.52
	$K = 20$	0.55	0.43
	$K = 10$	0.46	0.38
PCA	0.59		
LDA	0.89		

Table 1. Accuracy scores of Naive Bayes' performances.

Naive Bayes performed the best when data were transformed with LDA, achieving an accuracy of 89%. The precision, recall and f1-score of each class are in table 2 and the confusion matrices are in figures 2,3.

Genre	Precision	Recall	F1-score
Blues	1	0.90	0.95
Classical	1	1	1
Country	0.59	1	0.74
Disco	0.83	1	0.91
Hip Hop	1	0.80	0.89
Jazz	1	1	1
Metal	1	1	1
Pop	1	0.90	0.95
Reggae	1	1	1
Rock	0.60	0.30	0.40
Average	0.90	0.89	0.88

Table 2. The classification report of Naive Bayes combined with LDA.

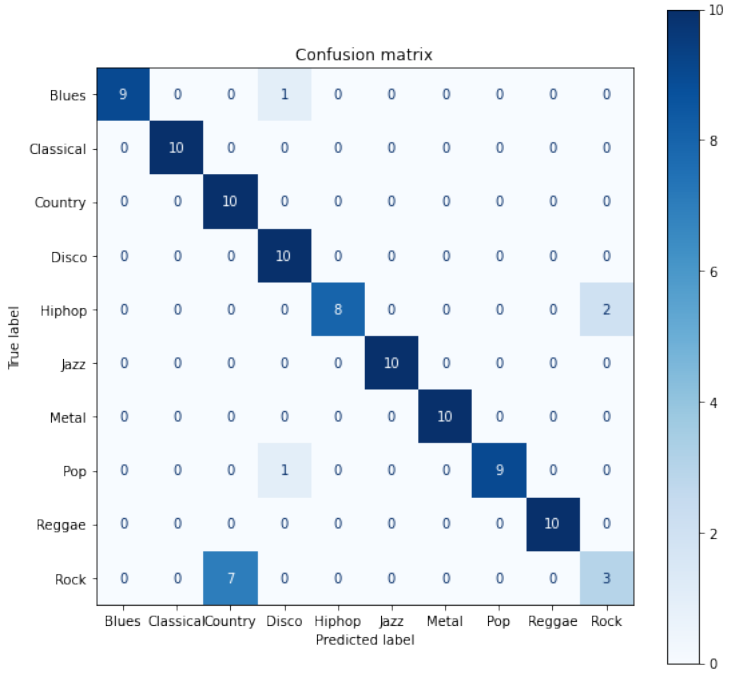

Fig. 2. The confusion matrix of Naive Bayes combined with LDA.

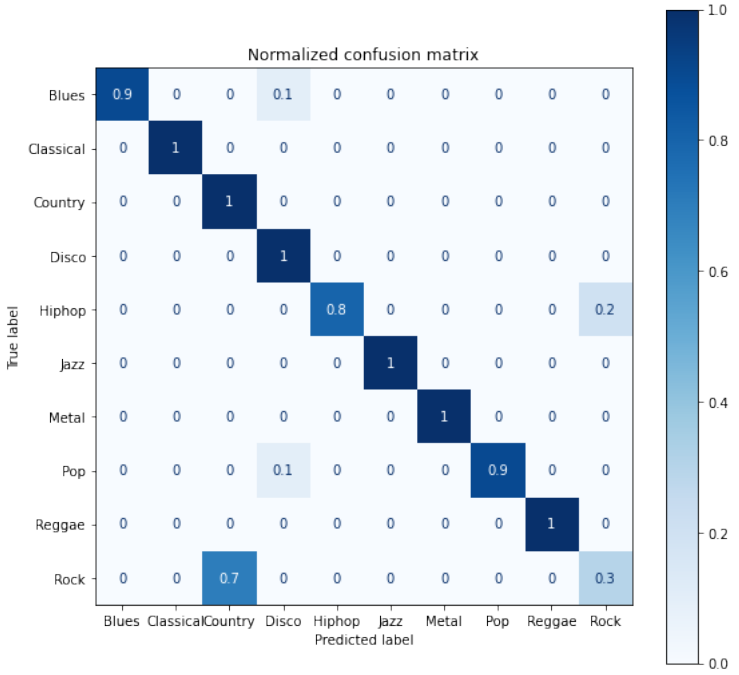

Fig. 3. The normalized confusion matrix of Naive Bayes combined with LDA

5.2 Stochastic Gradient Descent

Accuracy scores for SGD classifier are displayed in table 3.

All features	0.82		
Without highly correlated features	0.67		
Without low variance features	0.57		
RFE	$f = 50$	0.78	
	$f = 40$	0.77	
	$f = 30$	0.80	
	$f = 20$	0.75	
	$f = 10$	0.58	
K Best features	$K = 50$	0.74	0.78
	$K = 40$	0.77	0.77
	$K = 30$	0.70	0.72
	$K = 20$	0.70	0.64
	$K = 10$	0.48	0.51
Most important features	0.73		
PCA	0.6		
LDA	0.88		

Table 3. Accuracy scores of SGD's performances.

SGD achieved the best accuracy score along with LDA transformation, a score of 88%. The precision, recall and f1-score of each class are shown in table 4 and the confusion matrices are in figures 4 and 5.

Genre	Precision	Recall	F1-score
Blues	1	1	1
Classical	1	1	1
Country	0.53	0.80	0.64
Disco	0.83	1	0.91
Hip Hop	1	0.90	0.95
Jazz	1	1	1
Metal	0.91	1	0.95
Pop	0.83	1	0.95
Reggae	1	0.9	0.95
Rock	1	0.3	0.46
Average	0.91	0.88	0.87

Table 4. The classification report of SGD combined with LDA.

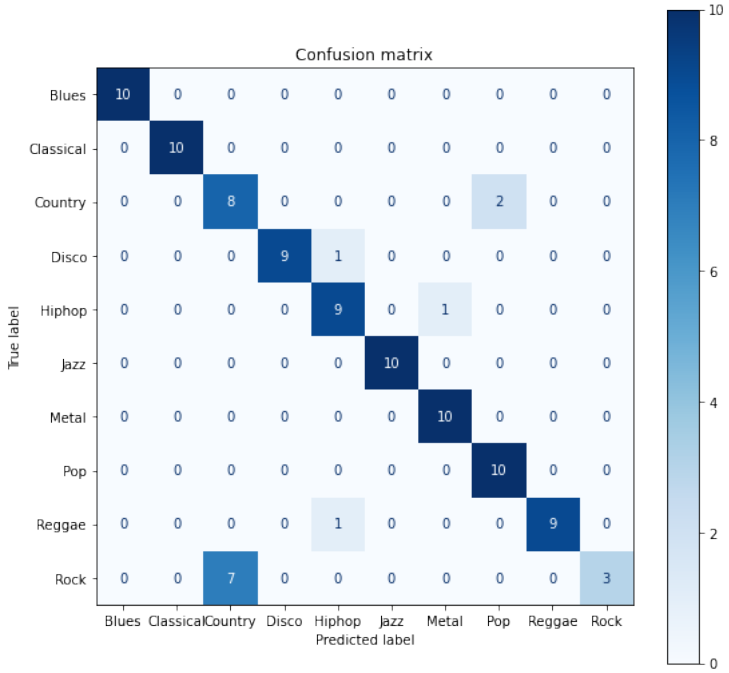

Fig. 4. The confusion matrix of SGD combined with LDA.

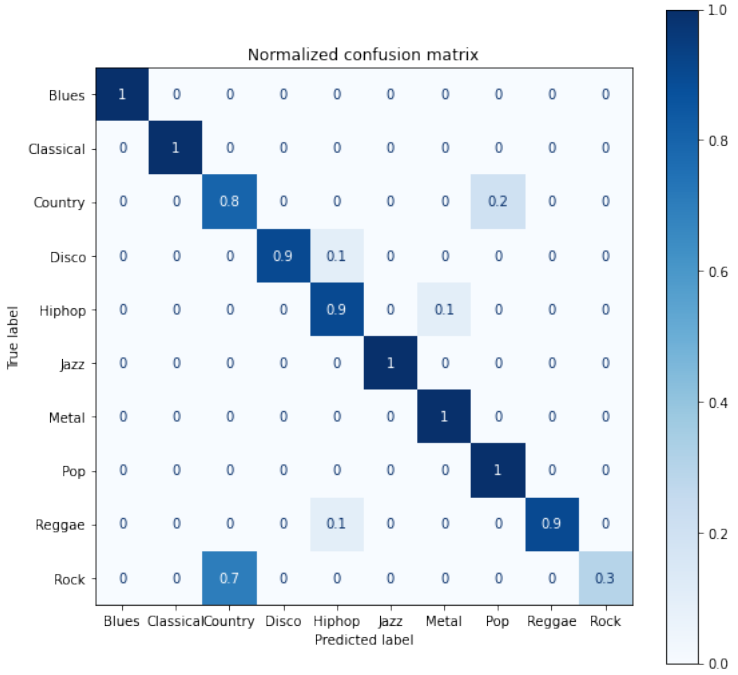

Fig. 5. The normalized matrix of SGD combined with LDA.

5.3 K-Nearest Neighbors

Accuracy scores of KNN classifier are displayed in table 5.

All features	0.69		
Without highly correlated features	0.63		
Without low variance features	0.62		
K Best features	$K = 50$	0.69	0.69
	$K = 40$	0.66	0.68
	$K = 30$	0.69	0.66
	$K = 20$	0.70	0.64
	$K = 10$	0.57	0.58
PCA	0.65		
LDA	0.91		

Table 5. Accuracy scores of KNN's performances.

KNN along with LDA transformation of features achieved the best accuracy score, 91%. Precision, recall and f1-score of each genre are shown in table 6 and the confusion matrices are in figures 6 and 7.

Genre	Precision	Recall	F1-score
Blues	1	0.90	0.95
Classical	1	1	1
Country	0.77	1	0.87
Disco	0.77	1	0.87
Hip Hop	0.83	1	0.91
Jazz	1	1	1
Metal	0.91	1	0.95
Pop	1	1	1
Reggae	1	0.9	0.95
Rock	1	0.3	0.46
Average	0.93	0.91	0.90

Table 6. The classification report of KNN combined with LDA.

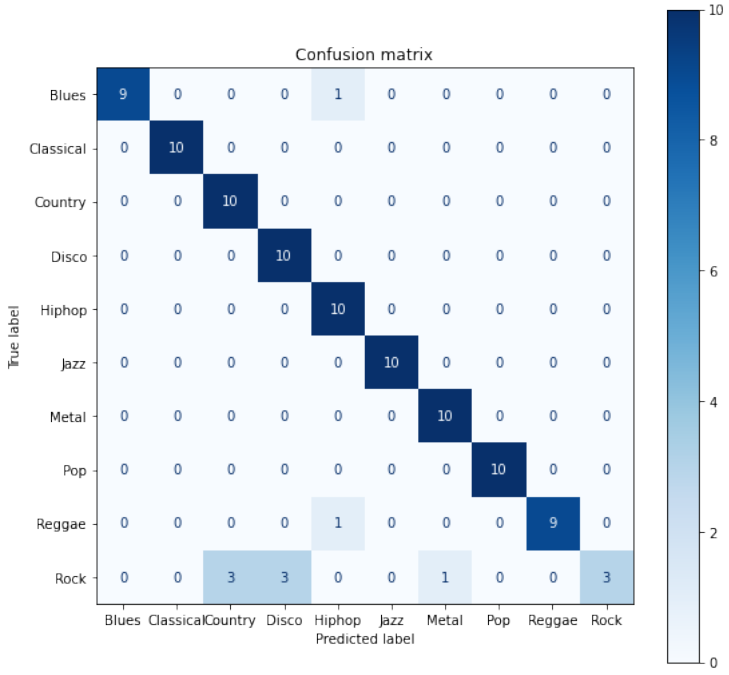

Fig. 6. The confusion matrix of KNN combined with LDA.

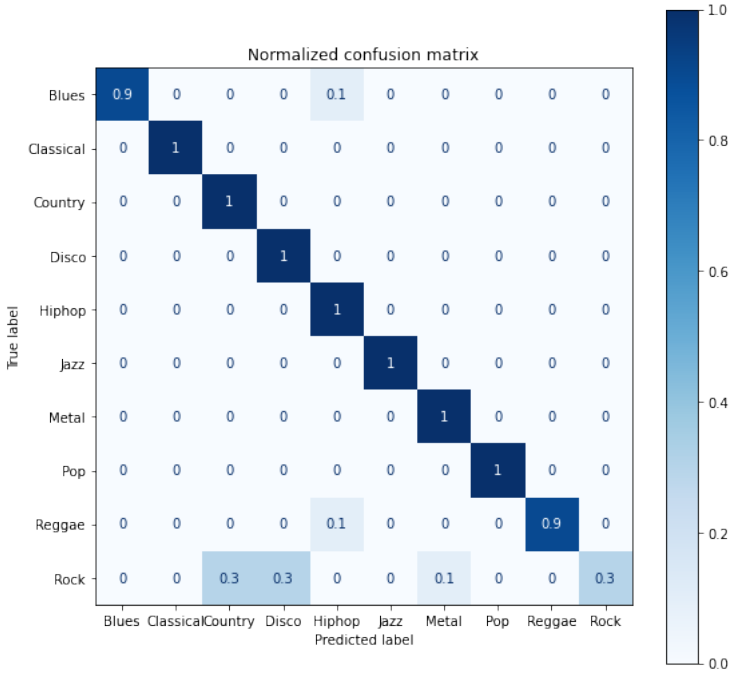

Fig. 7. The normalized confusion matrix of KNN combined with LDA.

5.4 Decision Tree

Accuracy scores for decision tree classifier are displayed in table 7.

All features	0.45		
Without highly correlated features	0.44		
Without low variance features	0.44		
RFE	$f = 50$	0.42	
	$f = 40$	0.44	
	$f = 30$	0.51	
	$f = 20$	0.39	
	$f = 10$	0.42	
K Best features	$K = 50$	0.54	0.55
	$K = 40$	0.56	0.47
	$K = 30$	0.48	0.49
	$K = 20$	0.51	0.40
	$K = 10$	0.46	0.43
Most important features	0.43		
PCA	0.36		
LDA	0.55		

Table 7. Accuracy scores of decision tree's performances.

Decision tree's best accuracy score was 55%. Both K best features (with k = 50 and mutual_info_classif) and LDA along with decision tree achieved the same score. However, based on average precision and average f1-score of classes, K best features performed slightly better. The precision, recall and f1-score of each genre are in table 8 and the confusion matrices are shown in figure 8 and 9.

Genre	Precision	Recall	F1-score
Blues	0.46	0.60	0.52
Classical	1	0.90	0.95
Country	0.40	0.40	0.40
Disco	0.33	0.30	0.32
Hip Hop	0.50	0.50	0.50
Jazz	0.62	0.50	0.56
Metal	0.78	0.70	0.74
Pop	0.71	0.50	0.59
Reggae	0.67	0.60	0.63
Rock	0.31	0.50	0.38
Average	0.58	0.55	0.56

Table 8. The classification report of decision tree combined with 50 best features.

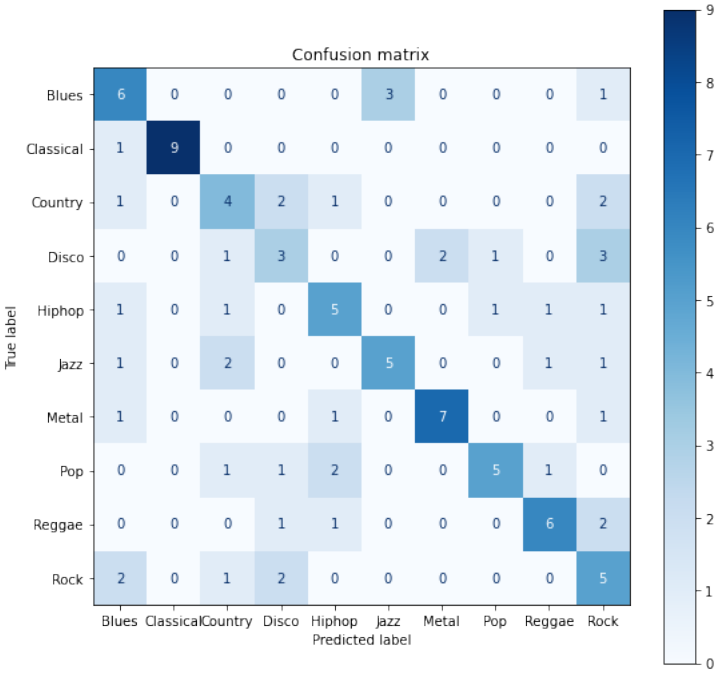

Fig. 8. The confusion matrix of decision tree with 50 best features.

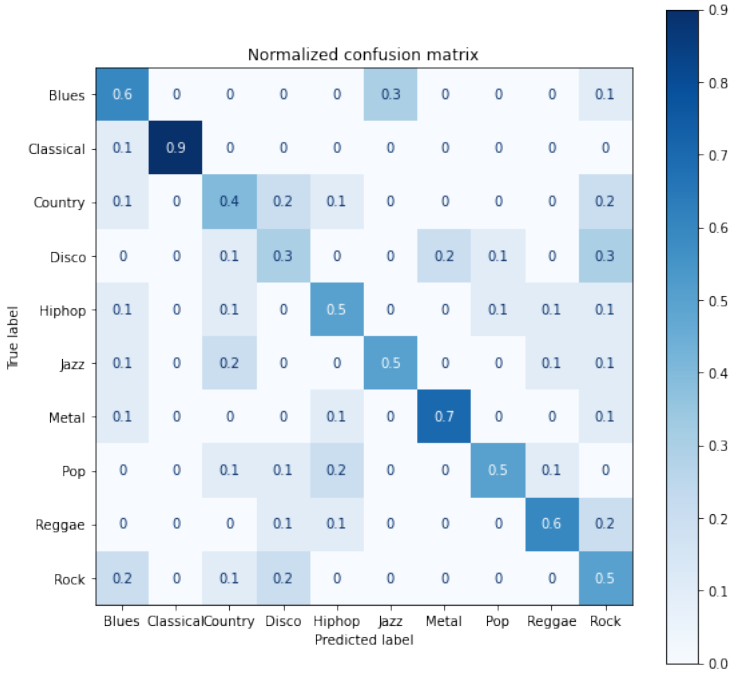

Fig. 9. The normalized confusion matrix of decision tree with 50 best features.

5.5 Random Forest

Random forest's accuracy scores are in table 9.

All features	0.70		
Without highly correlated features	0.65		
Without low variance features	0.67		
RFE	$f = 50$	0.70	
	$f = 40$	0.68	
	$f = 30$	0.70	
	$f = 20$	0.71	
	$f = 10$	0.61	
K Best features	$K = 50$	0.71	0.71
	$K = 40$	0.74	0.72
	$K = 30$	0.72	0.66
	$K = 20$	0.72	0.65
	$K = 10$	0.61	0.57
Most important features	0.66		
PCA	0.65		
LDA	0.90		

Table 9. Accuracy scores of random forest's performances.

Random forest performed the best with LDA transformation, achieving 90% accuracy. Precision, recall and f1-score of each genre are displayed in table 10 and the confusion matrices are in figures 10 and 11.

Genre	Precision	Recall	F1-score
Blues	0.91	1	0.95
Classical	1	1	1
Country	0.59	1	0.74
Disco	0.91	1	0.95
Hip Hop	1	0.90	0.95
Jazz	1	1	1
Metal	1	1	1
Pop	0.91	1	0.95
Reggae	1	1	1
Rock	1	0.10	0.18
Average	0.93	0.90	0.87

Table 10. The classification report of random forest combined with LDA.

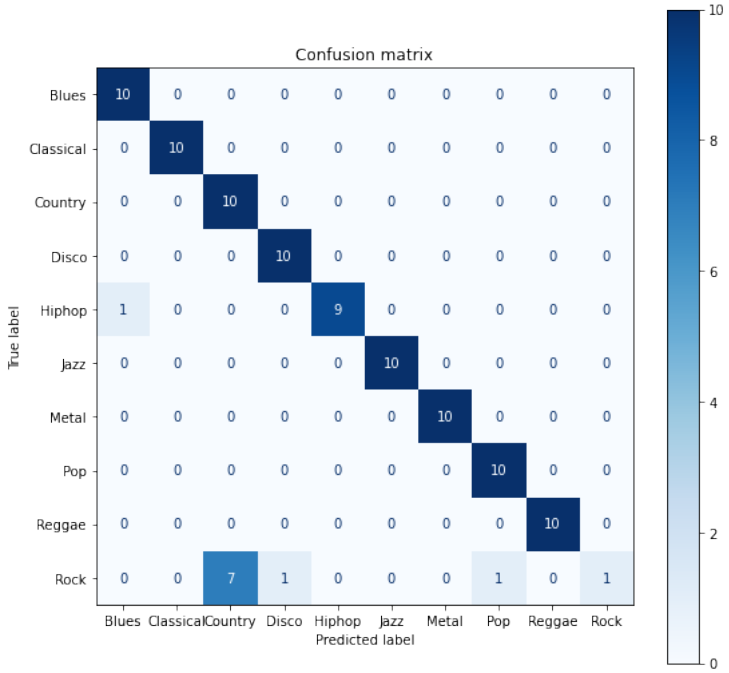

Fig. 10. Confusion matrix of random forest combined with LDA.

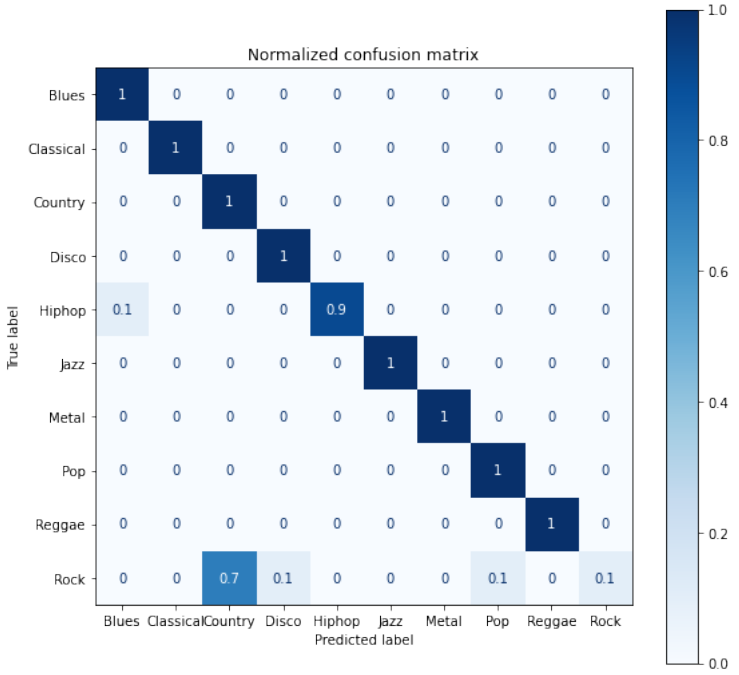

Fig. 11. The normalized confusion matrix of random forest combined with LDA.

5.6 Support Vector Machines

SVM's accuracy scores are displayed in table 11.

All features	0.73		
Without highly correlated features	0.64		
Without low variance features	0.63		
K Best features	$K = 50$	0.70	0.69
	$K = 40$	0.73	0.71
	$K = 30$	0.66	0.70
	$K = 20$	0.68	0.69
	$K = 10$	0.56	0.56
PCA	0.66		
LDA	0.94		

Table 11. Accuracy scores of SVM's performances.

SVM classifier with LDA transformation achieved an accuracy score of 94%. Precision, recall and f1-score of each genre are in table 12 and the confusion matrices in figures 12 and 13.

Genre	Precision	Recall	F1-score
Blues	1	1	1
Classical	1	1	1
Country	0.67	1	0.80
Disco	1	1	1
Hip Hop	1	0.90	0.95
Jazz	1	1	1
Metal	1	1	1
Pop	1	1	1
Reggae	1	1	1
Rock	0.83	0.50	0.62
Average	0.95	0.94	0.94

Table 12. The classification report of SVM combined with LDA.

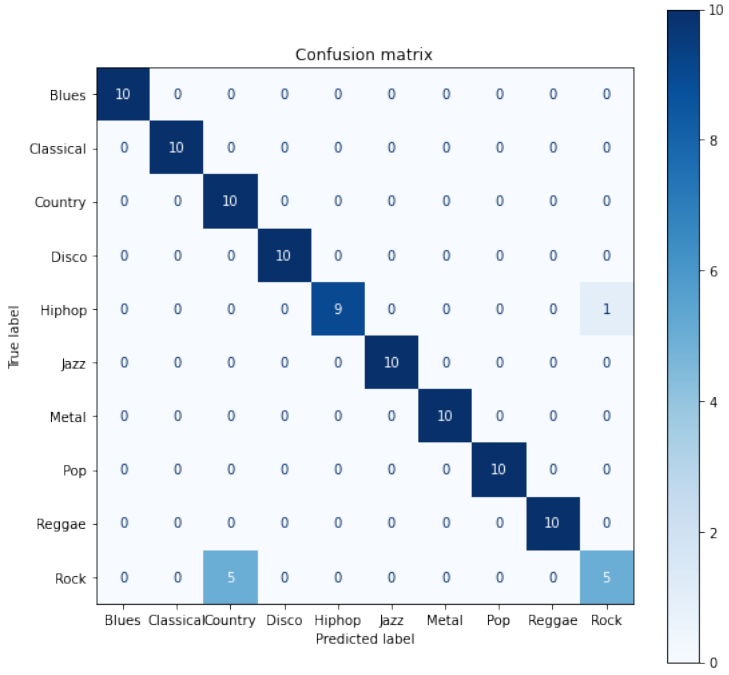

Fig. 12. The confusion matrix of SVM combined with LDA.

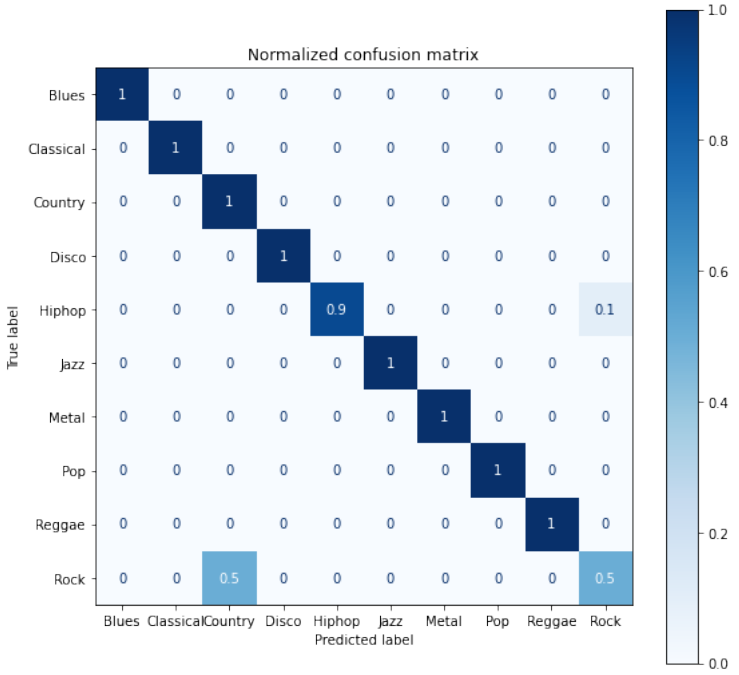

Fig. 13. The normalized confusion matrix of SVM combined with LDA.

5.7 First Neural Network

The accuracy scores of first NN are displayed in table 13.

All features	0.82	
K Best features	$K = 50$	0.85
	$K = 40$	0.85
	$K = 30$	0.82
	$K = 20$	0.76
	$K = 10$	0.71
PCA	0.77	
LDA	0.93	

Table 13. Accuracy scores of first NN's performances.

The best accuracy score, 94%, was obtained when data had been transformed with LDA. The classification report is displayed in table 14 and the corresponding confusion matrices are in figures 14 and 17.

Genre	Precision	Recall	F1-score
Blues	1	0.80	0.89
Classical	1	1	1
Country	0.90	0.90	0.90
Disco	1	1	1
Hip Hop	1	1	1
Jazz	0.83	1	0.91
Metal	0.91	1	0.95
Pop	0.90	0.90	0.90
Reggae	1	1	1
Rock	0.78	0.70	0.74
Average	0.93	0.93	0.93

Table 14. The classification report of first NN combined with LDA.

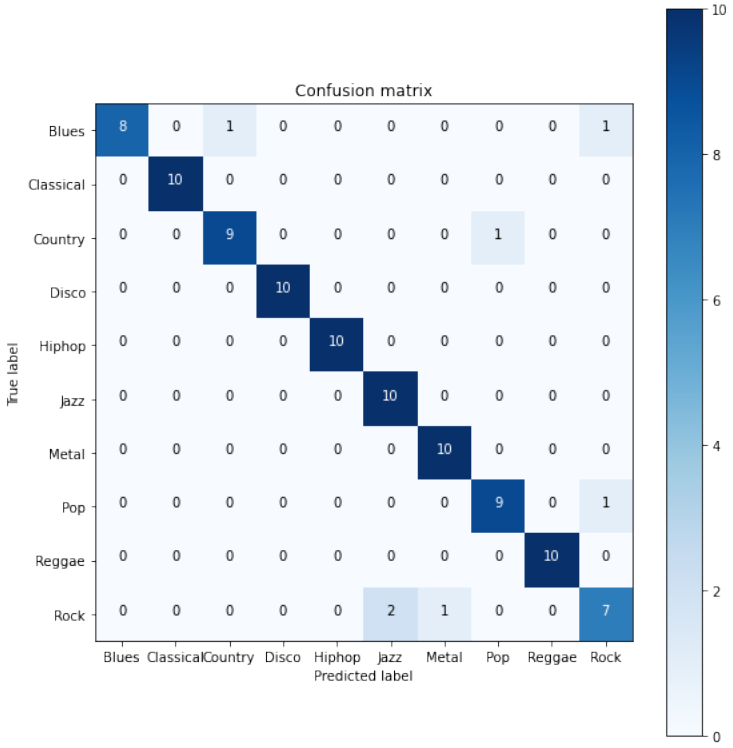

Fig. 14. The confusion matrix of the first NN combined with LDA.

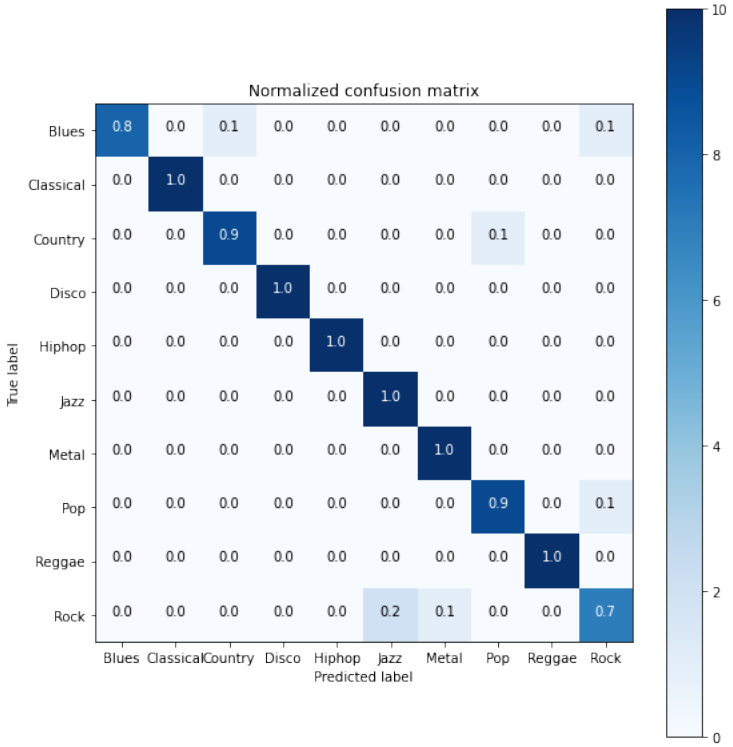

Fig. 15. The normalized confusion matrix of the first NN combined with LDA.

5.8 Second Neural Network

The accuracy scores obtained by the second NN architecture, the combination of convolutional and recurrent NN, are listed in the table 15.

All features	0.50	
Without highly correlated features	0.47	
Without low variance features	0.49	
K Best features	$K = 50$	0.63
	$K = 40$	0.65
	$K = 30$	0.62
	$K = 20$	0.58
	$K = 10$	0.52
PCA	0.56	
LDA	0.81	

Table 15. Accuracy scores of second NN's performances.

The model performs the best when features have been transformed with LDA, achieving an accuracy of 81%. The classification report is displayed in table 16 and the confusion matrices in figures 16 and 17.

Genre	Precision	Recall	F1-score
Blues	0.83	1	0.91
Classical	1	1	1
Country	0.56	1	0.71
Disco	1	0.60	0.75
Hip Hop	0.82	0.90	0.86
Jazz	1	1	1
Metal	0.75	0.90	0.82
Pop	1	0.70	0.82
Reggae	1	0.70	0.82
Rock	0.43	0.30	0.35
Average	0.84	0.81	0.80

Table 16. The classification report of second NN combined with LDA.

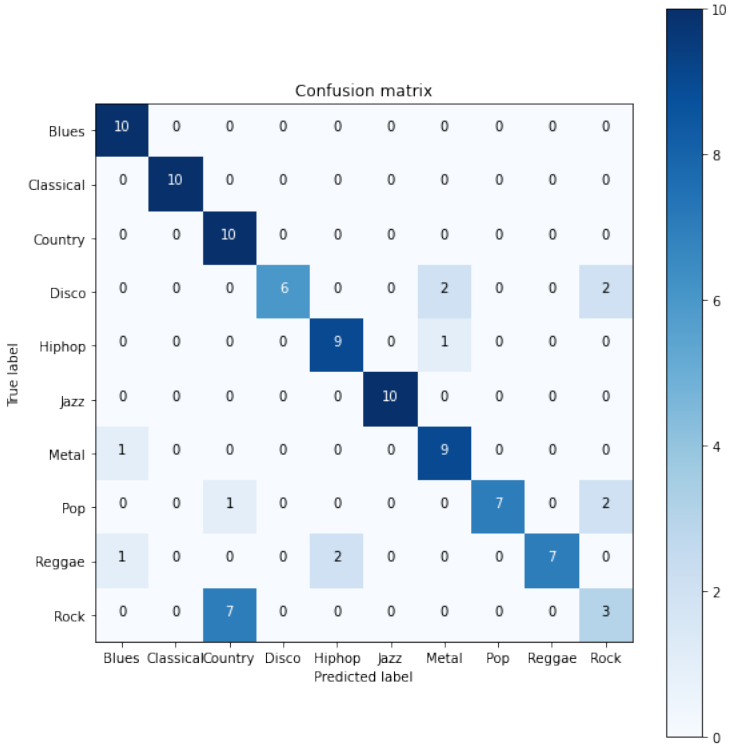

Fig. 16. The confusion matrix of second NN combined with LDA.

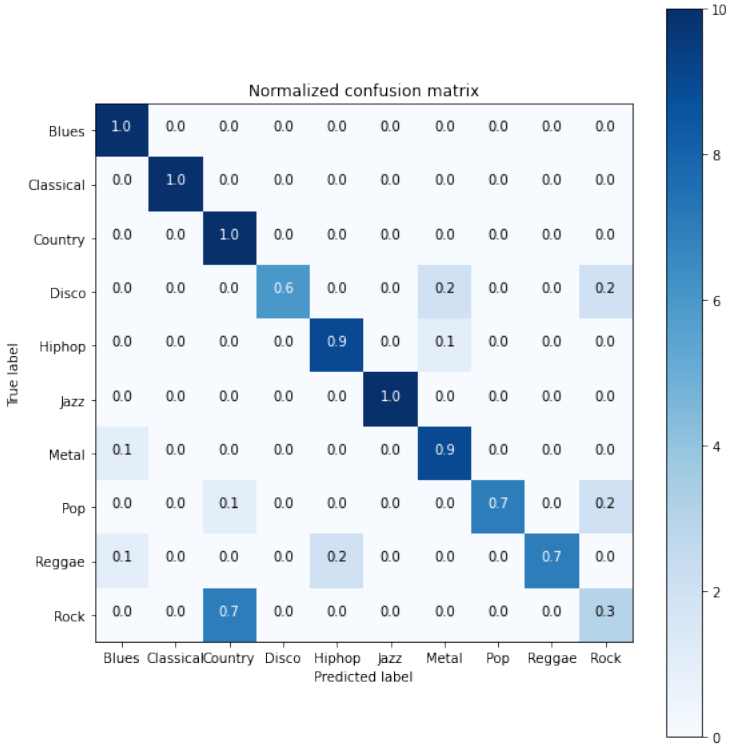

Fig. 17. The normalized confusion matrix of second NN combined with LDA.

5.9 Multilayer Perceptron

The accuracy scores of Multilayer Perceptron (MLP) classifier are listed in table 17.

All features	0.75		
Without highly correlated features	0.72		
Without low variance features	0.71		
K Best features	$K = 50$	0.80	0.75
	$K = 40$	0.81	0.76
	$K = 30$	0.76	0.74
	$K = 20$	0.71	0.73
	$K = 10$	0.55	0.61
PCA	0.62		
LDA	0.85		

Table 17. Accuracy scores of MLP's performances.

MLP achieves the best accuracy score, 85%, when combined with LDA transformation. Precision, recall and f1-score for each genre are listed in table 18. The corresponding confusion matrices are in figures 18 and 19.

Genre	Precision	Recall	F1-score
Blues	1	1	1
Classical	1	1	1
Country	0.53	0.90	0.67
Disco	0.78	0.70	0.74
Hip Hop	1	0.90	0.95
Jazz	1	1	1
Metal	0.83	1	0.91
Pop	1	0.80	0.89
Reggae	0.91	1	0.95
Rock	0.50	0.20	0.29
Average	0.85	0.85	0.84

Table 18. The classification report of MLP combined with LDA.

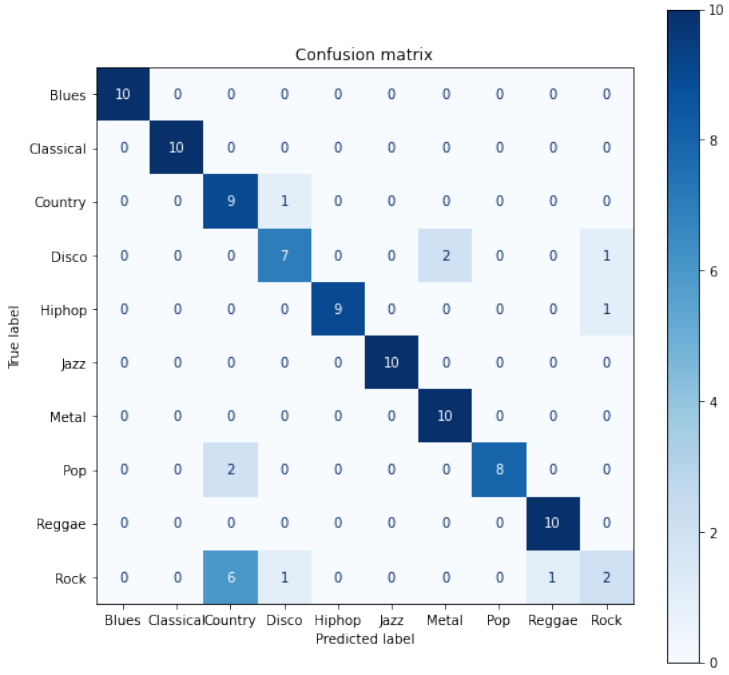

Fig. 18. The confusion matrix of MLP combined with LDA.

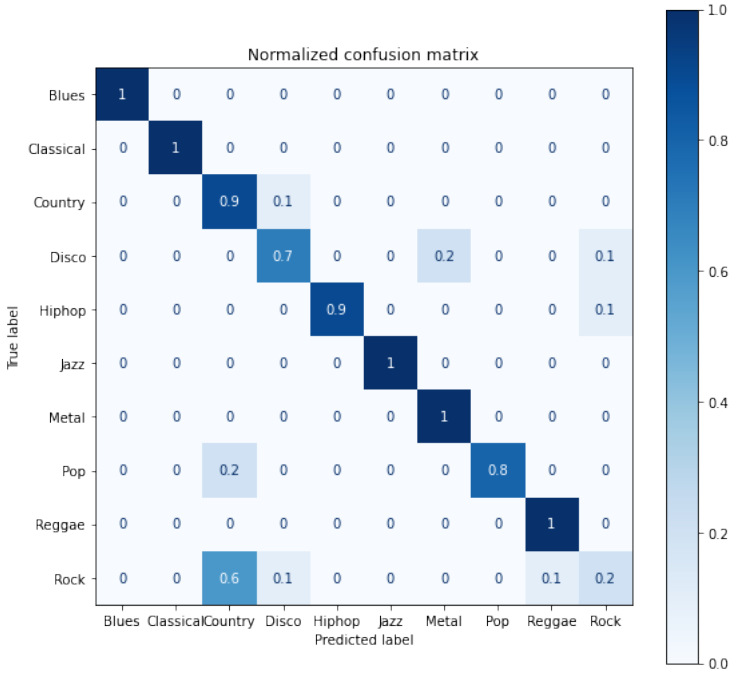

Fig. 19. The normalized confusion matrix of MLP combined with LDA.

5.10 XGBoost

The accuracy scores of XGBoost classifier are displayed in table 19.

All features	0.69		
Without highly correlated features	0.70		
Without low variance features	0.61		
RFE	$f = 50$	0.71	
	$f = 40$	0.67	
	$f = 30$	0.66	
	$f = 20$	0.64	
	$f = 10$	0.54	
K Best features	$K = 50$	0.64	0.64
	$K = 40$	0.67	0.62
	$K = 30$	0.62	0.65
	$K = 20$	0.70	0.58
	$K = 10$	0.58	0.58
Most important features	0.69		
PCA	0.67		
LDA	0.91		

Table 19. Accuracy scores of XGBoost's performances.

XGBoost achieved the best accuracy, a score of 91%, when features had been transformed with LDA. The classification report is in table 20 and the corresponding matrices are in figures 20 and 21.

Genre	Precision	Recall	F1-score
Blues	1	1	1
Classical	1	1	1
Country	0.67	1	0.80
Disco	0.91	1	0.95
Hip Hop	0.90	0.90	0.90
Jazz	1	1	1
Metal	0.90	0.90	0.90
Pop	0.91	1	0.95
Reggae	1	1	1
Rock	1	0.30	0.46
Average	0.93	0.91	0.90

Table 20. The classification report of XGBoost classifier combined with LDA.

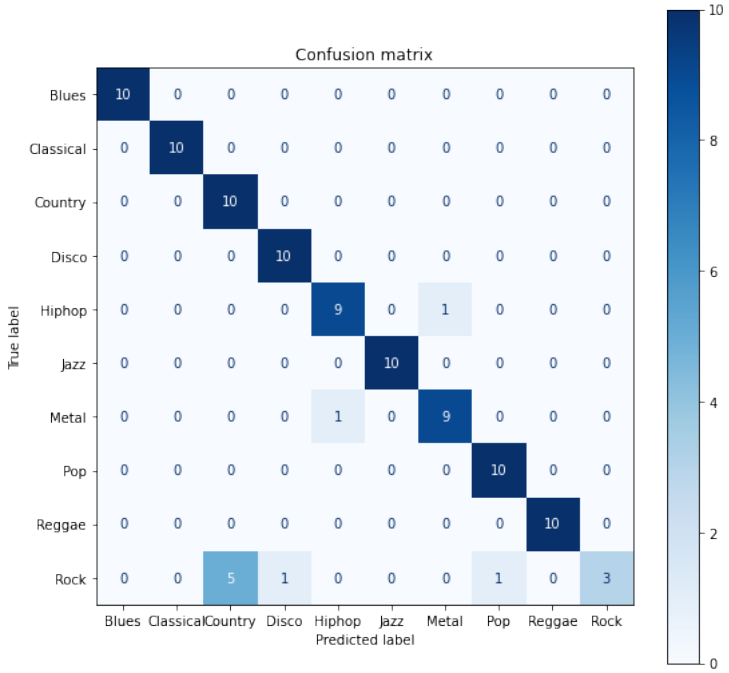

Fig. 20. The confusion matrix of XGBoost combined with LDA.

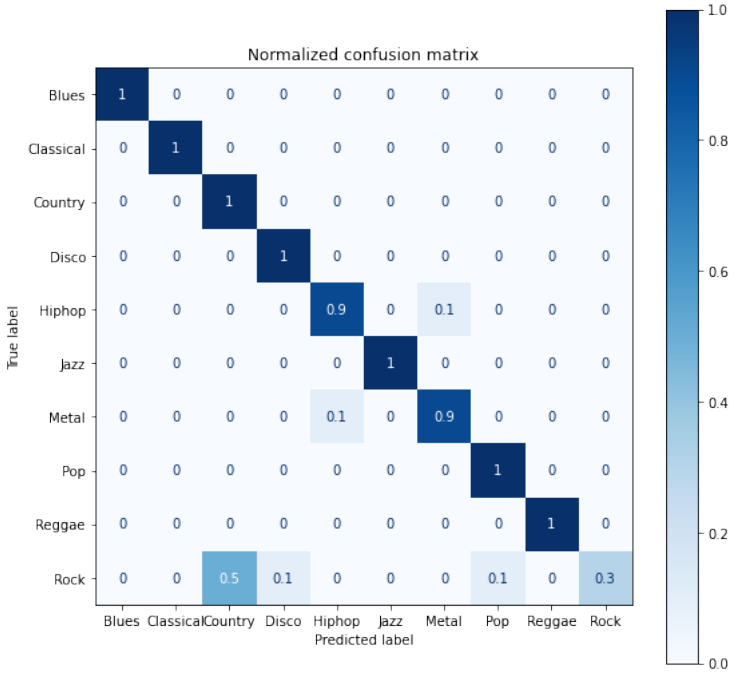

Fig. 21. The normalized confusion matrix of XGBoost combined with LDA.

5.11 Linear Discriminant Analysis

LDA classifier's accuracy scores are listed in table 21.

All features	0.77		
Without highly correlated features	0.65		
Without low variance features	0.58		
RFE	$f = 50$	0.78	
	$f = 40$	0.75	
	$f = 30$	0.65	
	$f = 20$	0.62	
	$f = 10$	0.49	
K Best features	$K = 50$	0.78	0.82
	$K = 40$	0.76	0.74
	$K = 30$	0.76	0.72
	$K = 20$	0.71	0.67
	$K = 10$	0.51	0.47
Most important features	0.61		
PCA	0.60		
LDA	0.94		

Table 21. Accuracy scores of LDA's performances.

LDA classifier performed the best when used along with LDA transformation of features, achieving a score of 94%. Precision, recall and f1-score for each class are shown in table 22 and the confusion matrices are in figures 22 and 23.

Genre	Precision	Recall	F1-score
Blues	1	1	1
Classical	1	1	1
Country	0.77	1	0.87
Disco	1	0.90	0.95
Hip Hop	0.82	0.90	0.86
Jazz	1	1	1
Metal	1	1	1
Pop	1	1	1
Reggae	1	0.90	0.95
Rock	0.88	0.70	0.78
Average	0.95	0.94	0.94

Table 22. The classification report of LDA classifier combined with LDA.

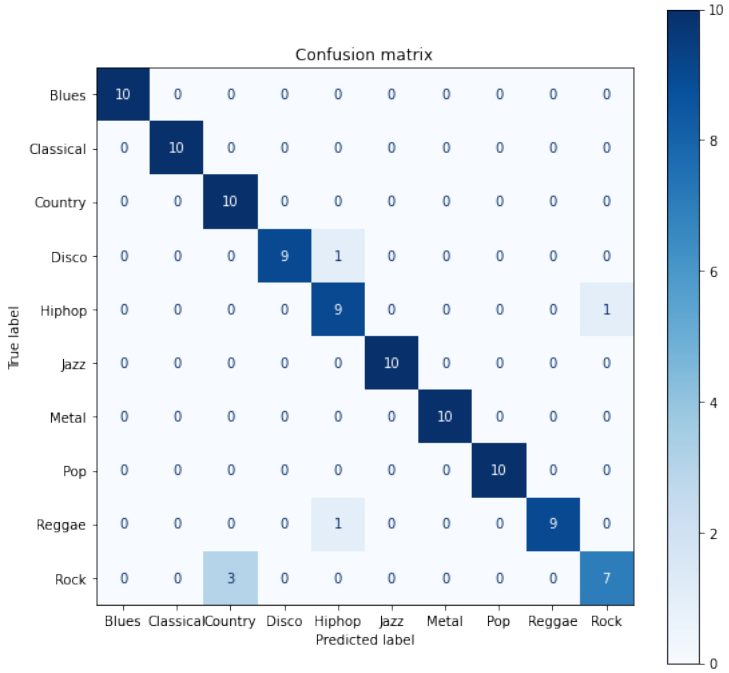

Fig. 22. The confusion matrix of LDA classifier combined with LDA transformation.

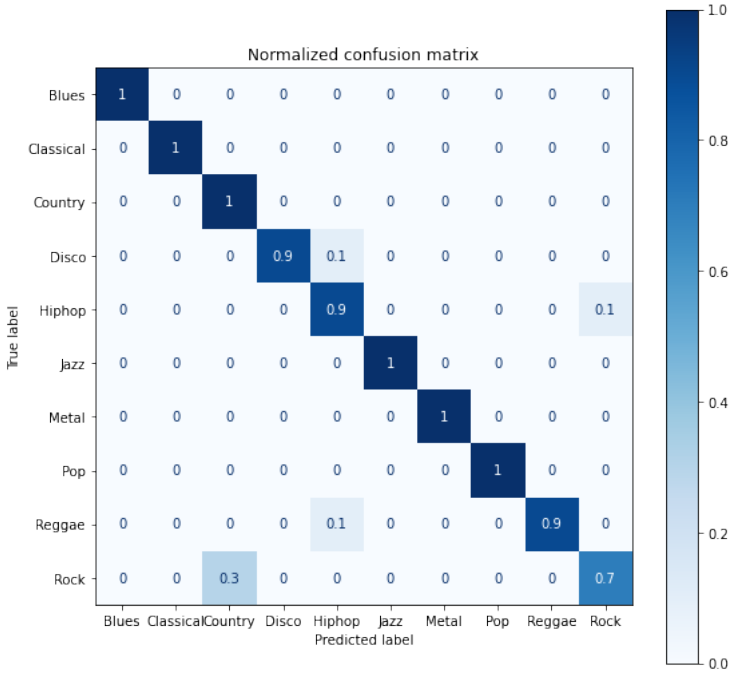

Fig. 23. The normalized confusion matrix of LDA classifier combined with LDA transformation.

6 Conclusion

The results of the best performances are collected in table 23. For each algorithm, accuracy, average precision, recall and f1-score are displayed.

Algorithm	Accuracy	Precision	Recall	F1-score
Naive Bayes	0.89	0.90	0.89	0.88
SGD	0.88	0.91	0.88	0.87
KNN	0.91	0.93	0.91	0.90
Decision Tree	0.55	0.58	0.55	0.56
Random Forest	0.90	0.93	0.90	0.87
SVM	0.94	0.95	0.94	0.94
NN #1	0.93	0.93	0.93	0.93
NN #2	0.81	0.84	0.81	0.80
MLP	0.85	0.85	0.85	0.84
XGBoost	0.91	0.93	0.91	0.90
LDA	0.94	0.95	0.94	0.94

Table 23. All results.

In general, all algorithms performed well and provided accurate predictions, except for decision tree. Based on the results obtained, transforming features with Linear Discriminant Analysis seemed to enhance algorithms' performances and improve significantly the accuracy scores. Rock genre was the most challenging to recognize for all algorithms.

SVM and LDA seem to be the most suitable for genre classification. They both achieved an accuracy score of 94%, an average precision of 95%, average recall as well as average f1-score of 94%. LDA classifier was better at recognizing rock genre rather than SVM.

References

1. Robert Gjerdingen and David Perrott. Scanning the dial: The rapid recognition of music genres. *Journal of New Music Research*, 37:93–100, 06 2008.
2. S. Lippens, J. P. Martens, and T. De Mulder. A comparison of human and automatic musical genre classification. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 4, pages iv–iv, 2004.
3. IBM Cloud Education. What is supervised learning? https://www.ibm.com/cloud/learn/supervised-learning.
4. C. K. On, P. M. Pandiyan, S. Yaacob, and A. Saudi. Mel-frequency cepstral coefficient analysis in speech recognition. In *2006 International Conference on Computing Informatics*, pages 1–5, 2006.
5. Shashidhar G. Koolagudi, Deepika Rastogi, and K. Sreenivasa Rao. Identification of language using mel-frequency cepstral coefficients (mfcc). *Procedia Engineering*, 38:3391 – 3398, 2012. INTERNATIONAL CONFERENCE ON MODELLING OPTIMIZATION AND COMPUTING.
6. George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10:293 – 302, 08 2002.
7. Kris West and Stephen Cox. Features and classifiers for the automatic classification of musical audio signals. 01 2004.
8. Engin Erzin. Automatic classification of musical genres using inter-genre similarity. *Signal Processing Letters, IEEE*, 14:521 – 524, 09 2007.
9. Marco Grimaldi, Padraig Cunningham, and Anil Kokaram. A wavelet packet representation of audio signals for music genre classification using different ensemble and feature selection techniques. pages 102–108, 01 2003.
10. Yannis Panagakis, Emmanouil Benetos, and C. Kotropoulos. Music genre classification: A multilinear approach. pages 583–588, 01 2008.
11. Brian McFee, Vincent Lostanlen, Alexandros Metsai, Matt McVicar, Stefan Balke, Carl Thomé, Colin Raffel, Frank Zalkow, Ayoub Malek, Dana, Kyungyun Lee, Oriol Nieto, Jack Mason, Dan Ellis, Eric Battenberg, Scott Seyfarth, Ryuichi Yamamoto, Keunwoo Choi, viktorandreevichmorozov, Josh Moore, Rachel Bittner, Shunsuke Hidaka, Ziyao Wei, nullmightybofo, Darío Hereñú, Fabian-Robert Stöter, Pius Friesch, Adam Weiss, Matt Vollrath, and Taewoon Kim. librosa/librosa: 0.8.0. https://doi.org/10.5281/zenodo.3955228, July 2020.
12. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

13. Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM.
14. François Chollet et al. Keras. https://github.com/fchollet/keras, 2015.