Music genre classification using GTZAN dataset

Riccardo Bona
Department of Computer Science, University of Milan, Via Giovanni Celoria 18, 20133 Milano, Italy

***Abstract*—The project consists in the analysis of the performance of four different classification methods applied to music genre discrimination using the *GTZAN* dataset. The dataset contains 100 audio tracks for each one of the 10 represented music genres. From the audio tracks, 9 short-term audio features have been extracted and integrated with mean and standard deviation values. In order to analyze the complexity of the classification task, the feature space has been clustered using *K-means*. Classification has been performed using the following algorithms: *K-nearest neighbors (K-NN)*, *decision tree (DT)*, *multi-layer perceptron (MLP)* and *support vector machines (SVM)*. The performance of each classification algorithm has been evaluated using *repeated K-fold cross-validation* and by comparing accuracy, precision, recall and F1-score for both training and test sets. Results highlight a much higher score on the training part for all classifiers. *SVM* and *MLP* achieved the best performance on the test part with overall accuracy scores of 75% and 77% respectively, the score for *K-NN* is around 69% and *DT* performed the worst with a score of 53%.**

***Index Terms*—music information retrieval, audio analysis, machine learning, music genre classification**

I. Introduction

Trying to define general rules able to discriminate different musical genre could prove a difficult task. This is due to multiple factors like the number of genres that need to be differentiated. E.g. a piece of classical music can be easily distinguished from a modern music track based on intuitive aspects, however when taking into account a wider set of music genres the complexity of the task increases considerably, more so if some of the genres present similar characteristics. Another important factor to consider is that genre labels associated to music tracks derive almost exclusively from human judgement (e.g. two person could give a different label to the same listened track) and a single music track could have more than one genre associated.

A solution to this problem could be advantageous in a variety of applications such as automatic content tagging in online music streaming, sharing, promotion and distribution services or music recommendation systems based on user activity (e.g. preferences, most listened tracks, downloaded tracks/albums etc.).

This project is based on the *GTZAN* dataset, containing 100 tracks (30 seconds long) for each of the 10 represented genres: *blues*, *classical*, *country*, *disco*, *hip hop*, *jazz*, *metal*, *pop*, *reggae* and *rock*. From each track, the following audio features have been extracted: *root-mean-square energy*, *entropy of energy*, *zero-crossing rate*, *spectral flux*, *spectral centroid*, *spectral rolloff*, *mel-scaled spectrogram*, *chroma vector* and the first 13 *mel-frequency cepstral coefficients*. All audio features have been subsequently integrated with mean and standard deviation values. Different projects employing similar sets of features on the *GTZAN* dataset (e.g. on *Kaggle*) showed satisfactory results with accuracy scores ranging above 75-80% using deep-learning techniques.

This project focuses on music genre classification employing the following four learning algorithms: *K-nearest neighbors (K-NN)*, *decision tree (DT)*, *multi-layer perceptron (MLP)* and *support vector machines (SVM)*. The classification results are evaluated using *confusion matrices* and validated via *repeated K-fold cross-validation* employing a different permutation of the dataset at each iteration.

II. System overview

This section covers all the preliminary work of features extraction and analysis, prior to the actual classification.

A. Dataset and feature extraction

As previously stated, the project is based on the *GTZAN* dataset containing a total of 1000 audio tracks (*.wav*), each 30 seconds long. All tracks are mono and sampled at 22050 Hz.

From the audio tracks, 3 time-domain and 6 frequency-domain features have been extracted. For all features, short-term extraction has been performed with a window size of 2048 samples ($\simeq$ 93 milliseconds) and with a window step set to a quarter of the window size, i.e. 512 samples ($\simeq$ 23 milliseconds). The audio file number 54 of the 100 *jazz* tracks appears to be corrupted, thus it was removed from the dataset implying a slight imbalance for the *jazz* class. The obtained dataset presents 5 rows containing outliers. Since the removal of the aforementioned outliers didn't affect the final classification scores, the following sections refer to the dataset including all rows.

B. Complexity of the problem

In order to analyze the complexity of the classification problem, the feature space has been clustered using the *K-means* algorithm. Prior to performing clustering, the features have been scaled using *min-max normalization* and *principal component analysis (PCA)* has been applied to the dataset in order to obtain easy-to-plot two-dimensional datapoints. Plotting the non-clustered datapoints (Fig. 3) shows that most of the points aren't well separated in the space relatively to their associated genres, exception made for those corresponding to the labels: *classical*, *metal* and *pop*. Applying *K-means* with three clusters (value choice based on the results of the *elbow method* shown in Fig. 2) shows that genres like

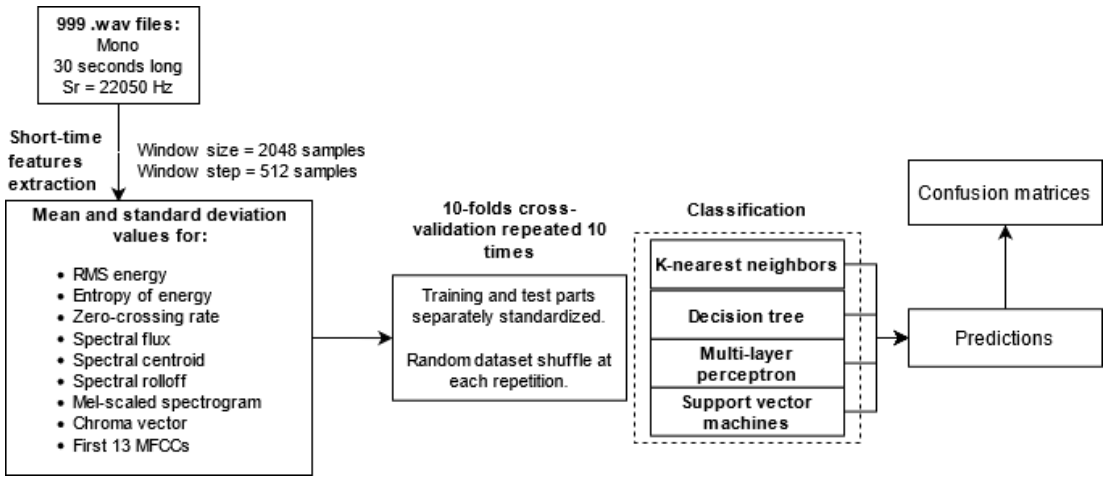

Fig. 1. Block diagram of the general workflow.

classical, *metal*, *pop*, and *reggae* are well represented in a single cluster, while the points relative to other genres like *rock*, *disco* or *blues* are divided among two or three clusters (Fig. 4). The difficulty of grouping the datapoints coherently with their class labels becomes even more evident by running *K-means* with a number of clusters equal to the number of genres in the dataset. The resulting clusters (Fig. 5) show that only for the classes *classical* and *metal*, a good portion of their associated datapoints are included in a single cluster. Thus, satisfactory classification are not to be expected for all of the 10 represented genres.

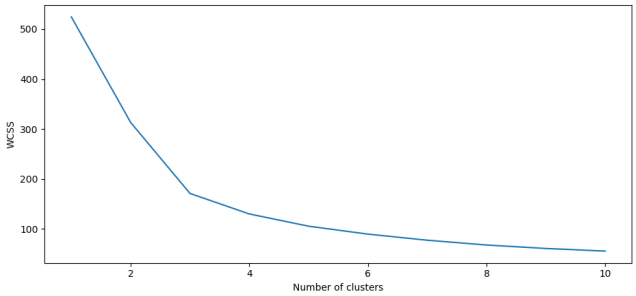

Fig. 2. *Within-cluster sum-of-squares* value for each number of clusters.

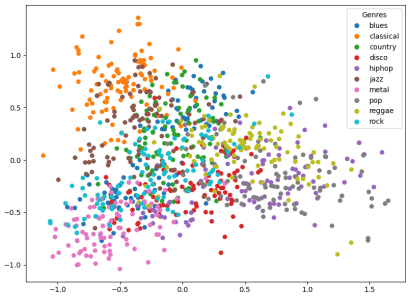

Fig. 3. Two-dimensional datapoints obtained from *PCA* plotted by the respective genre.

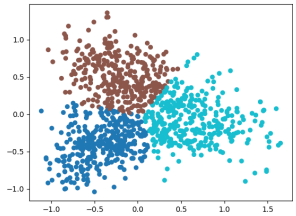

Fig. 4. *K-means* clustering with K=3. *Inertia score* = 171.04. *Silhouette score* = 0.44.

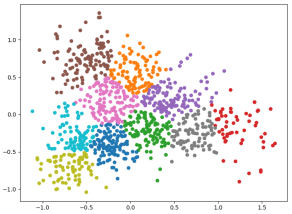

Fig. 5. *K-means* clustering with K=10. *Inertia score* = 55.32. *Silhouette score* = 0.35.

C. Validation method and classification implementation

The classification performance is validated using a custom implementation of *K-fold cross-validation* with K= 10, which is repeated 10 times in order to provide a more robust estimate for each classification algorithm. At the beginning of each repetition of the *10-fold cross-validation*, the dataset is randomly shuffled in order to avoid unwanted influences on the classification outcomes due to the fact that the audio tracks are ordered by genre. Since the *cross-validated* results change slightly with different shuffling instances, the final results have been obtained by averaging the results of multiple test runs.

The classification and validation phase is implemented as follows:

1) The dataset is randomly shuffled and subsequently partitioned into K = 10 equal and non-overlapping parts;
2) For each fold k = 1,...,K: k is retained as the test set, while the rest of the dataset minus the fold k is used as the training one;
   a) The training and test parts are standardized separately;
   b) The chosen classification algorithm is run on the training part and the resulting model is applied to the test set;
3) The resulting predicted labels are compared with the

actual ones for both training and test sets, calculating the relative confusion matrices.

This process is repeated 10 times for each classification algorithm and the values of the resulting confusion matrices are averaged.

III. Experimental set-up

This section covers the employed classification methods, hyper-parameters choice and analysis of the results. The values of all mentioned metrics are rounded down to the first two decimal digits.

A. K-nearest neighbors

The first tested method is *K-nearest neighbors*. In order to find a suitable value for the number of neighbors, the algorithm has been run following the steps described in subsection II-C, each time with a different value of K, ranging from 1 to 10. As the value of K changes, the results show almost no

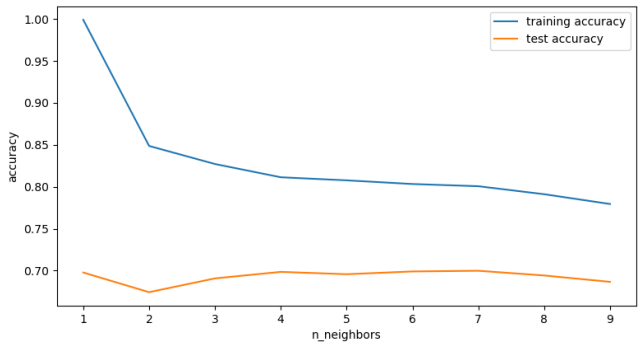

Fig. 6. Training and test accuracy scores for each tested value of K.

improvement in the test accuracy, while the training accuracy decreases as the number of neighbors increases. With the chosen value of $K = 4$, the *K-NN* algorithm achieved an overall training accuracy of 83% and test accuracy of 69%. In particular, by looking at table I, can be seen how the algorithm classifies well genres like *classical*, *metal*, and *pop*, while obtaining poor results in identifying genres like *disco* or *rock*. As theorized in the cluster analysis (subsection II-B), the *K-NN* algorithm, which is based on distance, classifies with satisfactory results only a few of the represented genres in the dataset. The algorithm was also tested by giving the datapoints weights equal to the inverse of their distance instead of being equally weighted. This approach, however, led to almost identical results.

B. Decision Tree

As a mean to determine the quality of each split in the implementation of the decision tree algorithm, the chosen measure for the information gain is *entropy* since *Gini* index yielded slightly worse results. The algorithm achieved an overall training accuracy of 99% and test accuracy of 53%. In order to try to mitigate this extreme case of *overfitting*, the algorithm has been tested with a limit on the maximum depth of the tree. The results (fig.7) show that limiting the

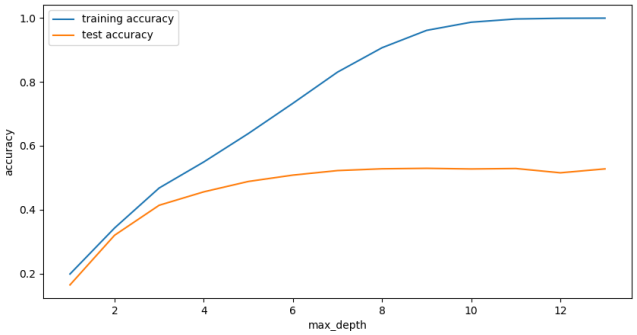

Fig. 7. Training and test accuracy scores for each tested maximum depth values.

maximum depth of the decision tree is not beneficial: with a maximum depth ≤ 5 the algorithm *underfits*, while at higher values it doesn't prevent the tree from *overfitting* and doesn't contribute in improving the test score.

C. Multi-layer perceptron

The *multi-layer perceptron* has been implemented with three hidden layers of sizes 128, 64 and 32, using the *relu* activation function. The chosen solver for weight optimization is *adam* (based on stochastic gradient descent). The *identity*, *logistic sigmoid* and *hyperbolic tan* (tanh) activation functions have also been tested but, since they achieved similar results, the default *relu*, activation function has been kept. *MLP* achieved an overall training accuracy of 99% and test accuracy of 75%. As in the previous cases, the most misclassified genres are *disco*, *reggae* and *rock*, while for the remaining genres the metric scores are above 70%. It's important to note that the convergence of the algorithm is rather slow; increasing the learning rate speeds up the model training at the cost of much lower accuracy scores. For the purposes of the project the learning rate has been left at the default value of 0.001.

D. Support vector machines

The last tested algorithm is *support vector machines*. The chosen kernel is *radial basis function*, which scored better than the other tested kernels (*linear*, *sigmoid* and *polynomial*). The chosen decision function shape is *One-vs-One (OVO)*

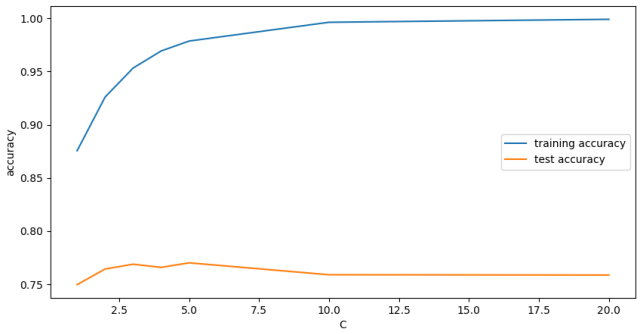

Fig. 8. Training and test accuracy scores for each tested value of the C regularization parameter (kernel: *radial basis function*, decision function shape: *OVO*).

TABLE I
REPEATED 10-FOLD CROSS-VALIDATION: TEST PERFORMANCE RESULTS

	K-NN			DT			MLP			SVM		
	Precision	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
Blues	0.76	0.81	0.78	0.49	0.47	0.48	0.77	0.78	0.77	0.78	0.84	0.81
Classical	0.87	0.93	0.90	0.79	0.75	0.77	0.91	0.89	0.90	0.91	0.92	0.92
Country	0.51	0.81	0.63	0.41	0.42	0.42	0.71	0.75	0.73	0.73	0.80	0.76
Disco	0.53	0.63	0.58	0.42	0.42	0.42	0.66	0.65	0.66	0.64	0.70	0.67
Hip-hop	0.68	0.55	0.61	0.47	0.50	0.48	0.72	0.70	0.71	0.74	0.69	0.71
Jazz	0.82	0.67	0.74	0.54	0.55	0.55	0.80	0.81	0.80	0.84	0.84	0.84
Metal	0.85	0.73	0.78	0.78	0.77	0.78	0.85	0.85	0.85	0.86	0.86	0.86
Pop	0.82	0.76	0.79	0.62	0.60	0.61	0.79	0.78	0.79	0.80	0.77	0.78
Reggae	0.66	0.57	0.61	0.50	0.49	0.50	0.65	0.68	0.66	0.72	0.66	0.69
Rock	0.55	0.43	0.48	0.28	0.29	0.29	0.62	0.61	0.61	0.65	0.60	0.63

*The support for the *jazz* results is 99 while for all others is 100.

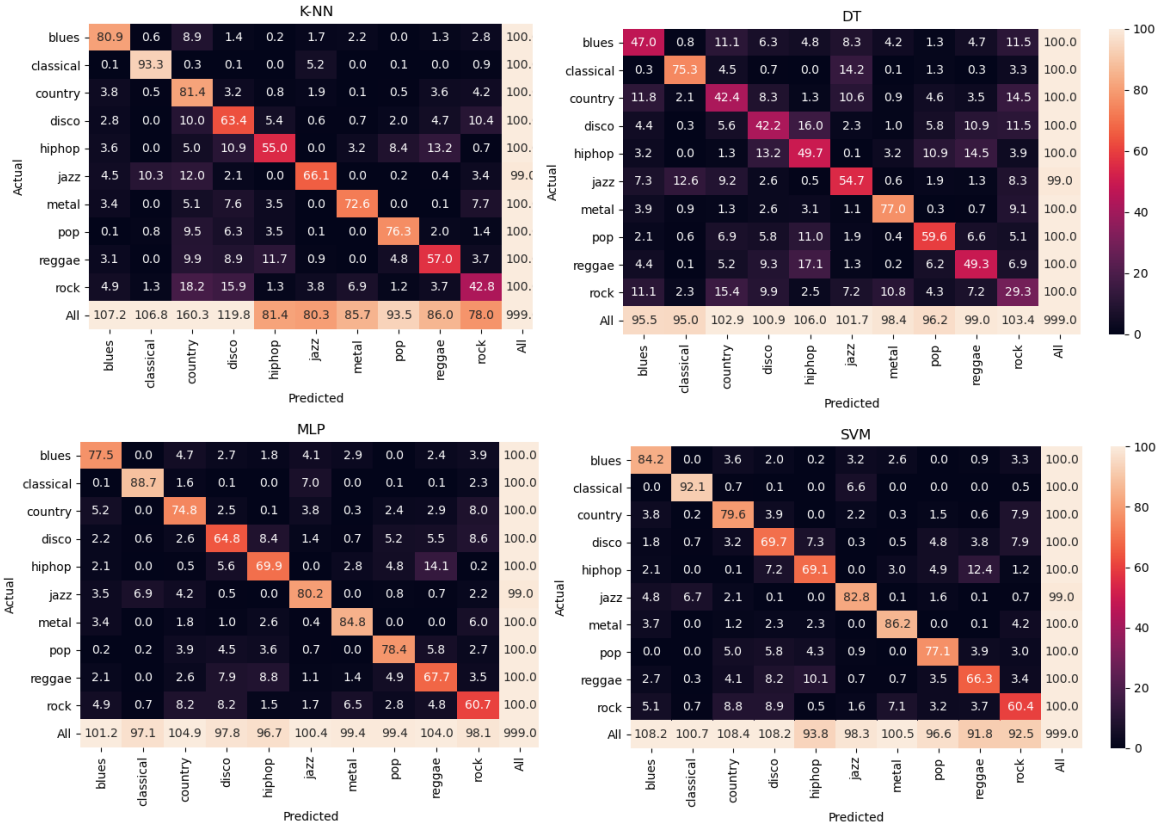

Fig. 9. Output confusion matrices (%) for each tested algorithm. All values are averaged across all repetitions of the respective *K-fold cross-validation* instances.

which performed almost identically to the *One-vs-Rest (OVR)* approach. The algorithm has been tested using different values of the C regularization parameter. As shown in fig. 8, as the value of C increases the test results worsen, while for values between 3 and 5 the test results are optimal even if by a slight amount. With a value of $C = 5$, *SVM* achieved an overall training accuracy of 98% and test accuracy of 77%. The *linear* kernel obtained a training accuracy of 96% and a test accuracy of 71%, the *sigmoid* kernel obtained a training accuracy of 59% and a test accuracy of 54% and the *polynomial* kernel achieved the best results with a polynomial order of 1, obtaining a training accuracy of 85% and a test accuracy of 74%. Compared all the previously tested methods, *SVM* obtained the best results, managing to better classify genres like *disco*, *reggae* and *rock*.

E. Confusion matrices confrontation

By looking at the resulting confusion matrices (fig. 9, the values are expressed in percentage), the hypothesis formulated in subsection II-B appears to be correct. *Classical* is the genre associated with the overall highest accuracy across all four methods, followed by *metal*. *K-nearest neighbors*, *multi-layer*

perceptron and *support vector machines* manage to achieve satisfactory accuracy results also for the genres *blues*, *country*, *jazz* and *pop*, while *decision tree* performed significantly worse for all genres. In general, the worst classified genres are *rock*, *reggae*, *disco* and *hip-hop*. This is due to the relatively high joint confusion percentage between *rock*, *country*, *disco* and *metal* and between *reggae*, *disco*, *hip-hop* and *pop*. If the *K-NN* confusion matrix is compared to the one of the other classifiers, it appears evident how the algorithm classified into the *country* and *disco* classes a number of datapoints significantly higher than the classes relative support. As a result, less tracks were labeled as *hip-hop*, *jazz*, *metal*, *reggae* and *rock*, resulting in worse precision scores for the two overly classified classes and worse recall scores for the misrepresented ones.

IV. Conclusions

SVM and *MLP* achieved satisfactory results, even if with a certain degree of confusion regarding some genres. It's however important to note that *SVM* obtained better results with significantly shorter model training times, in opposition to *MLP* which, among the four tested algorithms, resulted to be the most time inefficient. *K-NN* obtained lower accuracy scores, which was to be expected by looking at the clustering results, since the algorithm is simply based on the distance between datapoints. The *DT* algorithm scored the worst, achieving the lowest test accuracy, which could be due to the high number of extracted features, its inability to evaluate interactions between them and thus carrying a relatively low information if taken singularly. *DT* also showed a severe case of *overfitting* with a training accuracy of 99% compared to a test accuracy of 53% and limiting the maximum depth of the tree didn't improve the test score while affecting only the training one. *SVM* and *MLP* too obtained much higher training scores than the test ones even if significantly less distant. The *K-NN* algorithm seems to suffer less from *overfitting* issues since its training scores were overall the lowest and closer the test ones.

In conclusion, performing music genre classification based on time and frequency-domain audio features proved to be possible even if without excellent results. This could be due to the intrinsic difficulty of the task, which depends largely on the employed dataset, the extracted features and how the audio tracks have been labeled. By analyzing the results, it appears that most of the genres in the dataset are similar from a spectral and energetic point of view, which could also be influenced by how tracks, belonging to related music genres, are produced in analogous ways.

References

[1] Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. Journal of machine learning research, 3(Mar), 1157-1182.

[2] Ross S. M., Introduction to Probability and Statistics for Engineers and Scientists (fifth edition), 2014

[3] Tan P.-N., Steinbach M.,Kumar V., Introduction to Data Mining, (First Edition), Addison-Wesley Longman Publishing Co., Inc., 2005

[4] Shai S.-S. e Shai B.-D., Understanding Machine Learning: From Theory to Algorithms, Cambridge University Press, 2014.

[5] Mehryar M., Afshin R. e Ameet T., Foundations of Machine Learning (second edition), MIT Press, 2012.