About Music Genre Recognition

Course on “Audio Pattern Recognition”, University of Milan

Jean-Baptiste Lepidi

Abstract

Music genre recognition is a fundamental task in audio pattern recognition, with applications ranging from music recommendation systems to content-based music retrieval. This project aims to classify music files into different genres using machine learning techniques. The dataset used is the FMA dataset available on GitHub, and more precisely the "small" dataset that they provide. From this dataset, we will be extracting features like MFCCs, spectral centroid, or chroma vector, and train 2 different models (namely SVM and Random Forest) to compare their efficacy in identifying music genres.

1 Introduction

With the exponential growth of digital music repositories and streaming platforms, the ability to automatically classify music into different genres has become increasingly important. Music genre recognition plays a crucial role in various applications, including music recommendation systems, playlist generation, content-based music retrieval, and personalized music streaming services.

The task of automatically categorizing music files into predefined genres poses several challenges due to the subjective and multifaceted nature of musical genres. While humans can easily distinguish between genres based on auditory cues such as rhythm, melody, harmony, and timbre, developing computational models capable of replicating this capability requires sophisticated techniques from the field of audio pattern recognition.

This project does not aim to be cutting edge, nor exhaustive, but is more of an exploration of the realm of music genre recognition tasks. Our approach will be aimed at covering several important elements in these kinds of tasks, like what features and how many of them to extract, as well as comparing different machine learning models and their performances.

A lot of work has been done on the subject of music genre recognition, including but not limited to the work done by the researchers that created the FMA database (1), on which this project is based. Though the most impactful work in this field is mostly through the use of neural networks, we won't be covering them in this project.

2 Method and experimental setup

2.1 Dataset

As explained before, the dataset used for this project is the FMA dataset. This is a big archive of more than 100 000 music samples, free of rights, distributed in more than 150 genres. This dataset and the associated work can be found on GitHub. The researchers provide us with several sizes and qualities of dataset, ranging from 8000 30s extracts to 100 000 untrimmed extracts. In addition to the audio tracks, the researchers provide metadata about the tracks, like name, artists, and most importantly genre and subgenres of music associated to the track.

For the sake of simplicity we will be using the small dataset that is at our disposal. This dataset is composed of 8000 audio files of 30s, that are distributed between 8 genres. The dataset is also decomposed in 3 sets of training, validation and test data, and all the sets are more or less balanced between the classes.

Using a balanced dataset with a limited number of tracks will make our life easier in this process of exploration (less training time and supposedly simpler to classify the extracts), but 8000 tracks is still enough data that we have a good idea of the challenges of MGR (music genre recognition) tasks.

2.2 Method

Our exploration process will follow a simple scheme that will be repeated several times in order to gather significant data and draw conclusions about the task at hand.
The method is as follows :

- Extract relevant features from the data at our disposal
- Apply dimensionality reduction and clustering to visualize the data and get a feel of the quality of the data and the difficulty of the classification

task

- Train models : support vector machine (SVM) and random forest, to predict the genre of previously unseen tracks

2.3 Feature extraction

For the feature extraction stage, though the researchers of the FMA already provide extensive data, it was decided that it would be best to redo the extraction. The idea was to be able to compare the performances of 2 different feature extraction, one more succinct, and one more complete. The feature extraction is done using the *librosa* library in Python.

The first feature extraction process extracts only MFCCs and spectral centroid, that are two very interesting indices. MFCCs have been proven to yield very good performance on audio classification tasks, and spectral centroid are quite complementary as they capture more general information about the audio spectrum. Usually it is enough to extract the first 13 MFCC coefficients but 20 being the default for *librosa*'s MFCC extraction, that is what was used in the end.

The second one extracts not only MFCCs and spectral centroid, but also chroma vector, zero crossing rate and spectral contrast, rolloff, and bandwidth. This extraction is more complete, captures more elements about the spectrum, and therefore increases our chances of finding discriminant features to classify the tracks.

Note that all those features are not extracted at the track level, but at a frame level, yielding several thousands of feature vectors per track. Following the idea used in the FMA researchers' approach, we calculated statistical indices about those features in order to reduce the extraction to one feature vector per track. The calculated indices are : mean, min, max, median, kurtosis, skew and standard deviation. This choice can of course be discussed given that it gets rid of the temporal information previously extracted. This loss of information can be detrimental to our classification process but makes the training easier. We will need to keep this in mind for the rest of the paper.

2.4 PCA and clustering

We will make use of clustering to get an estimate of the quality of the data, which will tell us how "easy" the classification task will be.

To perform the clustering we will use dimensionality reduction through principal components analysis (PCA). First to reduce to 2 dimensions in order to be able to visualize the clusters. Then we will use a PCA prodcedure to get a feel of how much of the variance in the data can be explained by each attribute, so that we can choose the "right" number of dimensions to reduce to in order to have a fast clustering, but not lose too much information.

Because the second clustering will be on more than 2 dimensions, we will not be able to visualize the data and therefore we will need to use other means to evaluate the clustering, namely : silhouette score, homogeneity, compactness, V-measure, adjusted rand index (ARI) and normalized mutual information (NMI).

The clustering is done using the k-means algorithm which will work especially well because we know beforehand the number of clusters, namely 8, because we have 8 music genres in the small dataset.

2.5 Training the models

Our goal is to train 2 different models to try to predict the genre of unknown tracks. We will be training a support vector machine (SVM) and a random forest model, using 5-fold cross validation.

The first step is to divide our dataset in 2 parts namely training and testing data. As we use k-fold cross validation, we do not actually need to have a validation set.

The second step is to normalize the data, by subtracting the mean and dividing by the standard deviation. This is a crucial step for the SVM training as SVM is very sensitive to scale. If we do not scale the data, then the model will have very bad performances because it is based on a measure of distance that will mean nothing if the data is not normalized. This is not true for random forest though, as it is composed of decision trees that split the data based on a threshold which is not dependent of the scale.

Once the data has been scaled, we can start the actual training process. These two models have hyperparameters that we need to tune. For this we will be using a method called grid search, where we provide arrays of values to be tested for the hyperparameters, and the program will train one model for each combination of parameter, and keep the best performing one. For the SVM we will be using a rbf kernel because it fits our problem quite well as it can capture non linear relationships in the data, and we will be tuning 2 parameters : C, the regularization that tunes the tradeoff between high training accuracy and good test accuracy (basically avoids overfitting and underfitting), and gamma which is the parameter of the rbf kernel. For the random forest, we will be tuning the number of estimators, namely the number of trees used in the forest, as well as the maximum number of features considered for splitting at each decision node.

With the trained model, the only step left is to evaluate the performance on the test data, to see how well the model is able to generalize to unknown data.

3 Experimental results

We will first analyze the clustering and then the classification results. As previously said we will analyze and compare the results for 2 feature sets of different sizes.

3.1 PCA and Clustering

3.1.1 Smaller feature space

For this first feature space, using dimensonality reduction to 2 dimensions with PCA, and k-means clustering, we can visualize the following clusters :

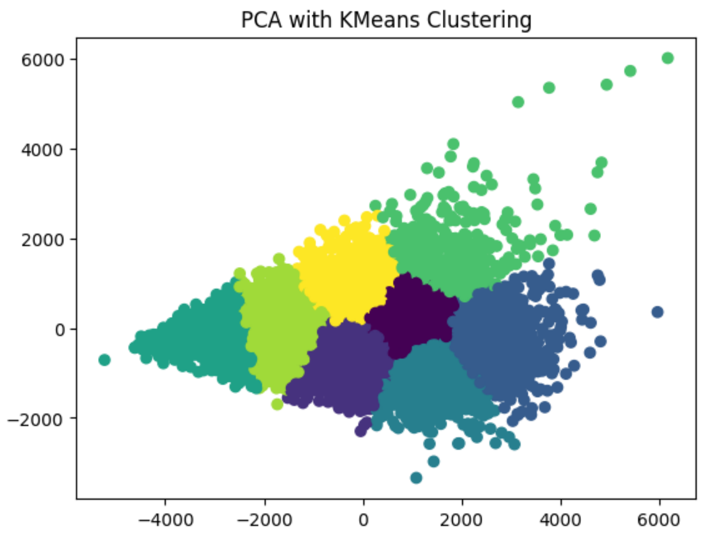

Figure 1: k-means clustering of the 2-dimension PCA for the smaller feature space

Visually, the clustering seems quite good, in the sense that the clusters are well defined. But, they are absolutely not disjoint which indicates that the classification could be hard.

PCA can help us understand how many features are "relevant" in the feature space, and how many features we actually need to explain most of the variance on the data :

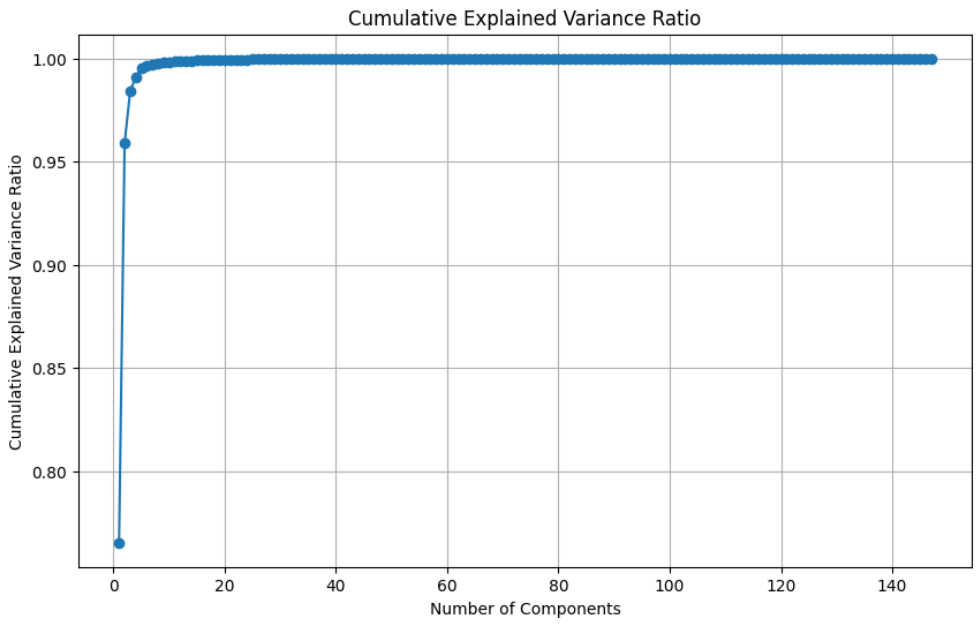

Figure 2: Cumulative explained variance

As we can see, more than 99% of the variance can be explained by 3 dimensions, so the "ideal" number of dimensiosn for PCA would be around 5. But it also means that our clustering with 2 dimensions is quite representative of the data.

The clustering evaluation for both number of dimensions is reported in the table below :

	2 dimensions	5 dimensions
Silhouette score	0.340	0.293
Homogeneity	0.102	0.104
Compactness	0.105	0.107
V-measure	0.104	0.106
ARI	0.060	0.062
NMI	0.104	0.106

Table 1: Evaluation of the clustering

The performances of the clustering for both PCA yield very similar results. Without going into too much detail about each evaluated index, the provided metrics indicate that while there is some structure in the clustering, it is not strong. The clusters are moderately separated according to the silhouette score, but they are not very homogeneous or complete. The

ARI and NMI scores suggest that the clustering does not agree strongly with the ground truth labels. These results indicate that the classifying task at hand will in fact not be easy and that it would not be surprising to get "low" accuracy on the trained models.

3.1.2 Larger feature space

The larger feature set yields quite similar results to the smaller one. The disposition of the clusters is similar, as well as the actual evaluations of the clustering. We can see though, that the "ideal" PCA would be around 10 dimensions, meaning that we have added meaningful features to our set.

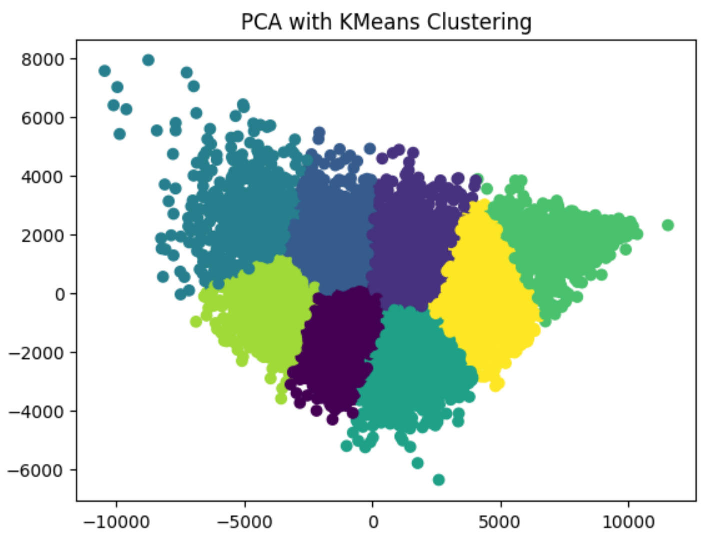

Figure 3: *k*-means clustering of the 2-dimension PCA for the smaller feature space

	2 dimensions	10 dimensions
Silhouette score	0.336	0.262
Homogeneity	0.106	0.109
Compactness	0.109	0.112
V-measure	0.108	0.111
ARI	0.063	0.065
NMI	0.108	0.111

Table 2: Evaluation of the clustering

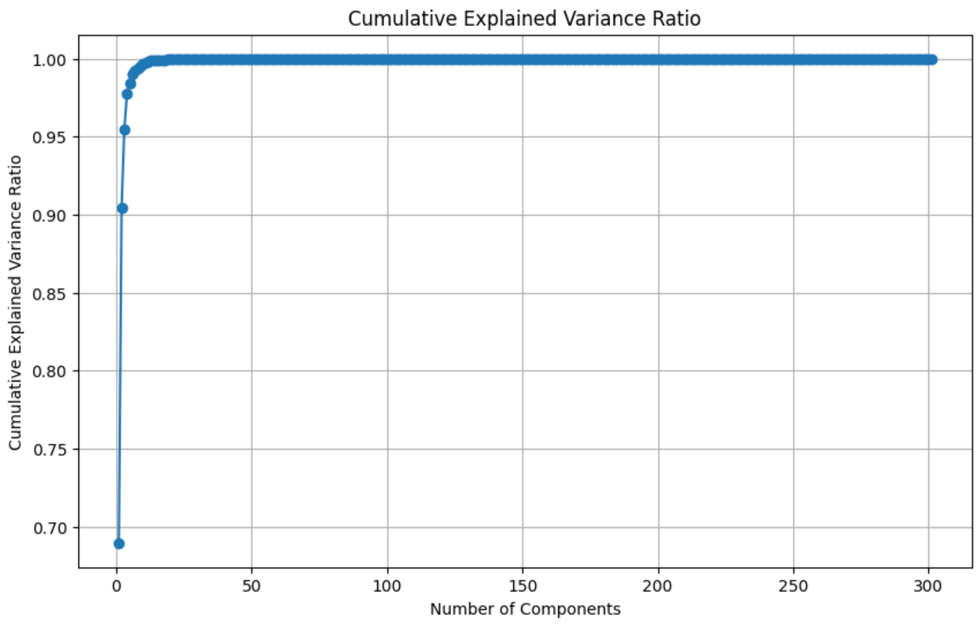

Figure 4: Cumulative explained variance

3.2 Classifiers

After training both classifiers, we get quite similar results in all cases. The first thing we can notice is that with the larger feature space, we get better accuracy, but that is not enormous, as we gain about 3 to 5% accuracy. The second thing we can notice is that the random forest yields a bit better results than SVM. Overall, the performance of the models is around 40 to 45% accuracy which is not bad considering what we just saw in the previous section (namely that the classification task at hand was quite difficult, given the clustering results), but is not amazing.

The performances of the models on both feature sets are reported in the table below :

	SVM	Random Forest
Smaller feature set	39%	41%
Larger feature set	43%	44%

Table 3: Accuracy of the classifiers

What we can conclude from this is that the MFCCs and spectral centroid are good features to use when classifying music genre : in fact, all the other extracted features in the larger feature set don't make that big a difference on the final performance of the model. We could also note that though 8000 samples is not a bad number, it might not be enough to get very good accuracy.

4 Conclusion

Overall we have seen that the MGR task, though very useful, is quite difficult and requires a huge amount of data and processing, even for not that great of a result. It is important to recall that we have been using a statistical representation of our data which does not always perfectly capture all the characteristics, especially time dependent features like rhythm.

One thing to note is that we could also have dove into the usage of neural networks which are probably the best models to use for this kind of difficult classification task. Other interesting experiments that could be conducted include the comparison between "classical" models like SVM and Random Forest, and deep learning approaches like CNNs or recurrent networks.

References

[1] Michaël Defferrard, Kirell Benzi, Pierre Vandergheynst and Xavier Bresson, *FMA: A Dataset For Music Analysis (v3)*, Sep 2017