Hochschule Karlsruhe
University of
Applied Sciences

Hochschule Karlsruhe- University of Applied Sciences

Machine learning in Python
Final Project

Teacher: Sarah Haq

Team members:
Shania Alessandra Martínez Anaya 81073
Jesús Gerardo Ortega Peimbert 81077
Samuel Russo Jaime 81175
Mauricio José González Aceves 81054

Submission date: January 31, 2022

Project Description:

The company is hoping that you can help them make sense of their audio data through exploratory data analysis, and build a robust machine learning model which serves two purposes:

1. Identifies the most important features for classifying music
2. Can be used to predict the genre of a new audio file

Methodology:

1. Exploratory data analysis

The first step in the project was to understand the data that we have. We realized that there was some information that was not necessary, for example the “filename” of the audio. Then after analyzing the data, we decided to plot some of the features that we believe can help us identify the classification of the music, these features are the tempo, the beats and the spectral bandwidth.

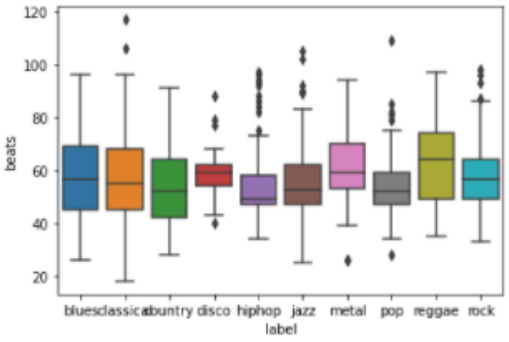

Figure 1.- Boxplot of the beats

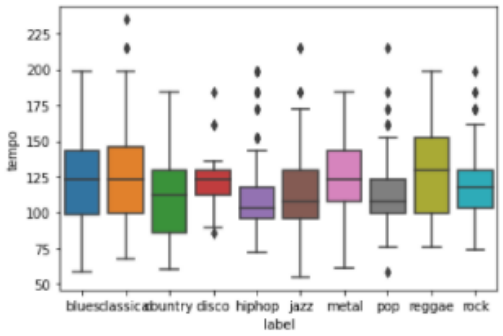

Figure 2.- Boxplot of the tempos

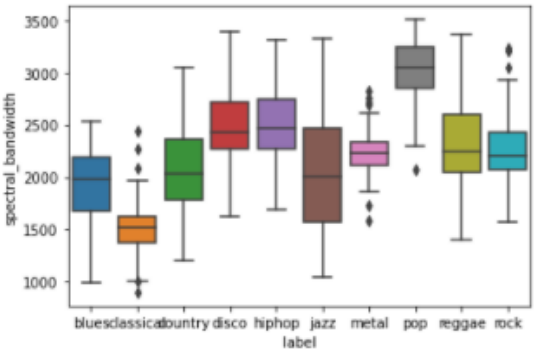

Figure 3.- Boxplot of the spectral bandwidth

After that, we used hot encoding for label, the feature defined as an object, since we want to have the same type of information for the management of the data.

Looking at the boxplots, we can see that there are similar patterns between the features of the different genres, and as a team we determined that further analysis would be necessary to correctly interpret the data.

2. Feature engineering and data cleaning

We proceed to check if there were duplicated data. As we do have, we drop all the duplicated features and check if in the remaining features there was correlated information. We found out that we had a lot of correlation so we deleted “beats”, “roll off”, “sepctral_centroid” and “mfcc2” since we don’t need them.
We checked if we had missing data and we didn't find any, but we decided to check the standard deviation in order to see if we could keep cleaning the data, we saw that there were 2 features with little std so we don’t need them either.
After the feature engineering we made, we split our data into a train and test set.

3. Build the machine learning model

As this is a classification problem, we decided to use 3 classifier models to compare and decide which one is the best for this information.
Models chosen:

a. Decision Tree Classifier
b. XGBoost Classifier
c. Random Forest Classifier

For each classification model we train the model with help of the sklearn and xgboost libraries. After the definition of the model, we did a comparison between the first five predicted values and the actual ones. In each model we calculated the evaluation metrics, meaning the accuracy, recall, precision and f1 score. Then to have a better

understanding we plot the confusion matrix, the learning curve and the feature importance of each classifier.

4. Evaluate the machine learning model

In table 1. we wrote the values of each metric and each model in order to compare them

Metric	Decision Tree	XGBoost	Random Forest
Accuracy	0.4	0.54	0.59
Recall	0.41	0.57	0.61
Precision	0.4	0.53	0.59
F1 score	0.4	0.54	0.59

Table 1. Evaluation metrics

We realized that the Decision Tree was not really accurate or precise, when we tried with the XGboost all the parameters improved, at least with the Random Forest the results were slightly better so we decided to choose that and keep working in this classifier.

5. Hyperparameter tuning

We realized that the evaluation matrix of the random forest was the highest of all tres models, so we tried to improve it even more. Using the , however the hyperparameter tuning didn’t help our model, due to the decrease in the evaluation metric after the second iteration.

Results:

1. Decision Tree

- Confusion matrix

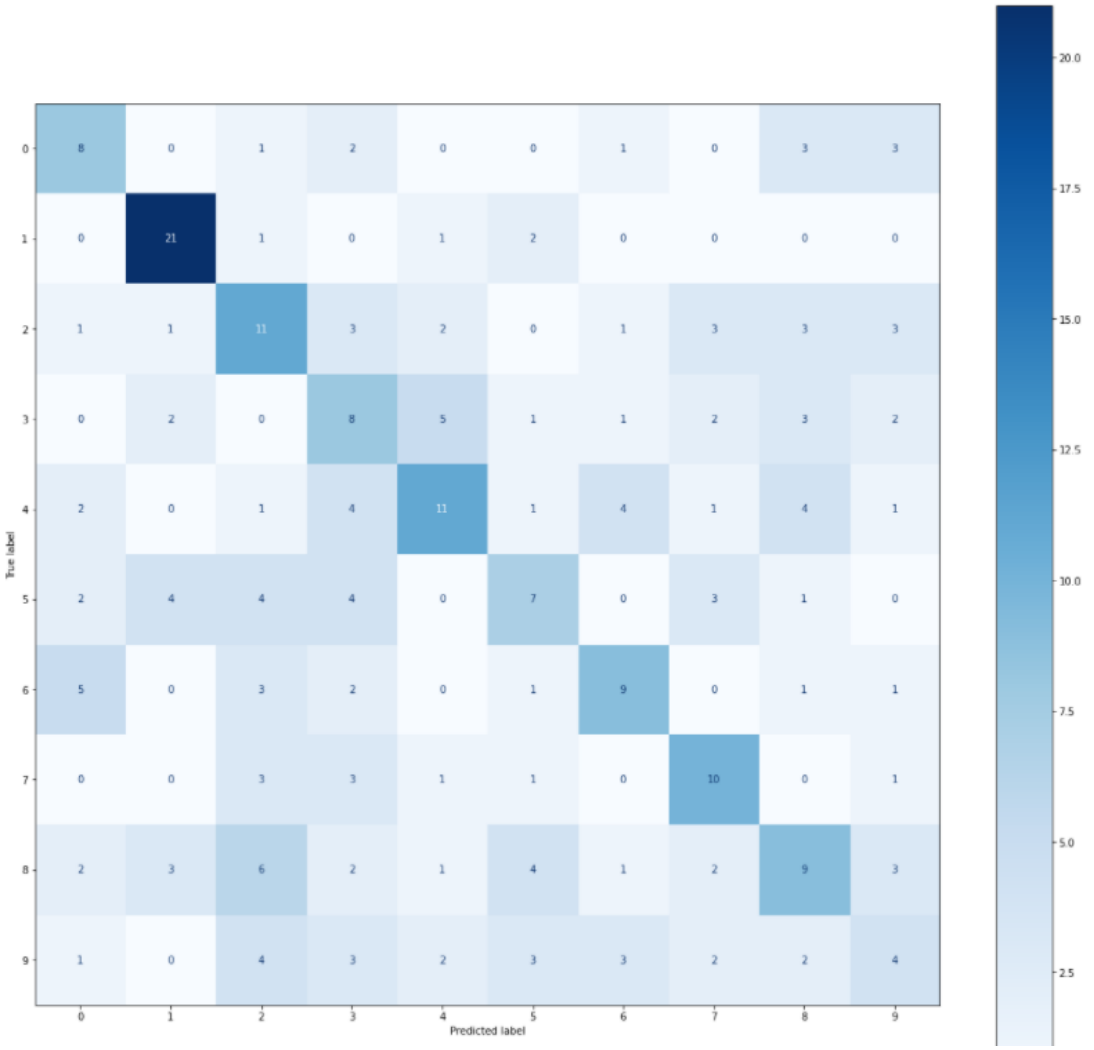

Figure 4.- Confusion matrix of decision tree

For the case of the Decision Tree we can notice that the Accuracy of the Model is far below the other two, resulting in a strong Precision for some numbers, but a low average Precision for the rest of the number. The times that the Model correctly predicted a number are mostly below a frequency of 10, which explains the low Accuracy of the Decision Tree Model.

- Learning curve

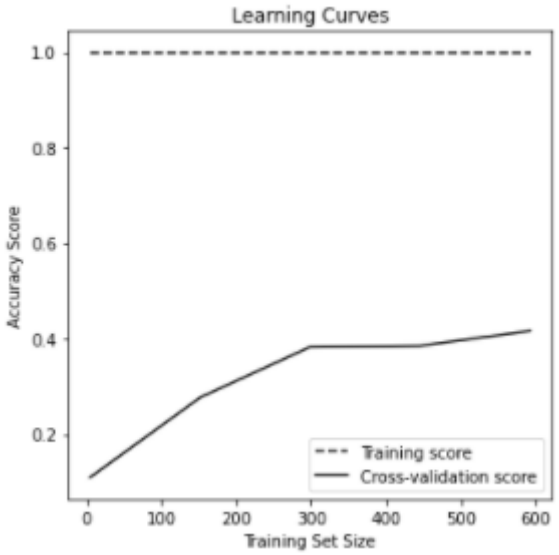

Figure 5.- Learning curve of decision tree

- Feature Importance

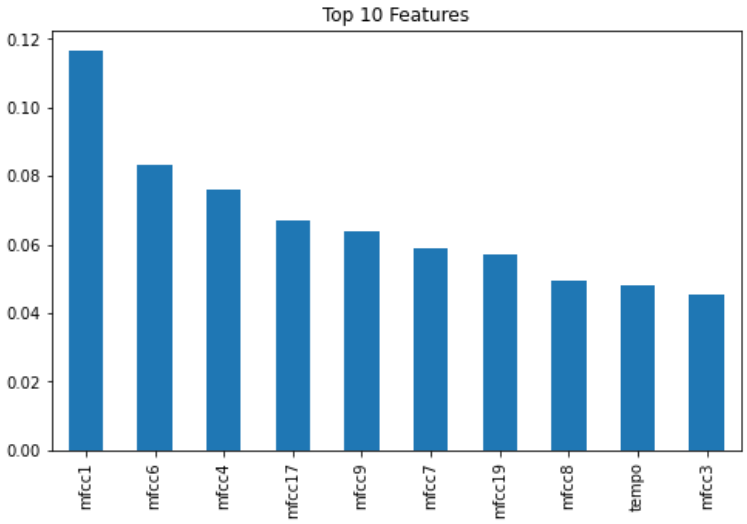

Figure 6.- Feature Importance of decision tree

2. XGBoost

- Confusion Matrix

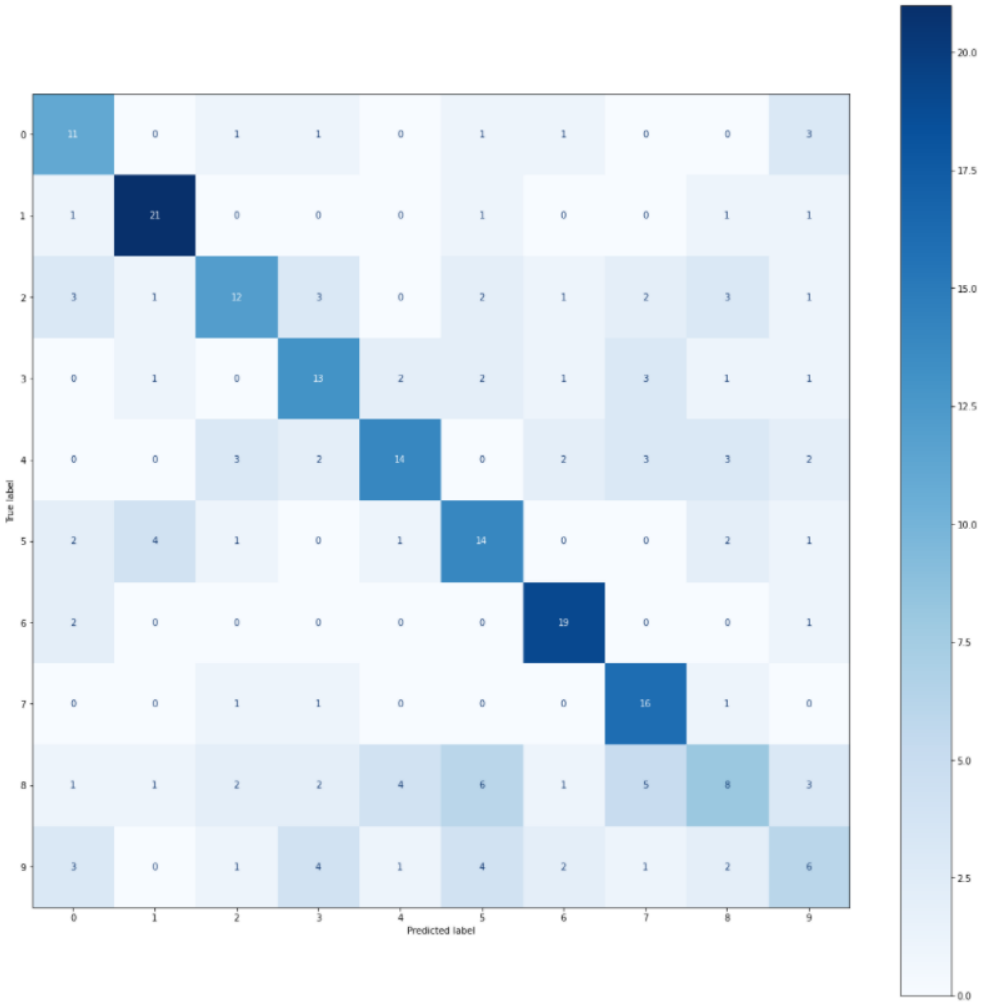

Figure 7.- Confusion matrix of XGBoost

To explain the Confusion Matrix of the XGBoost Model, we need to take into account two important elements: the Accuracy (0.54) and the Precision (0.53). With these two pieces of information we can understand the result of the Matrix better. The number of times that the Model correctly Predicted a value is far greater than the ones in the Decision Tree Model, with a bigger Accuracy reflected in most of the numbers being above 11. We can also see some spikes in numbers, which is not desirable, but well received.

- Learning curve

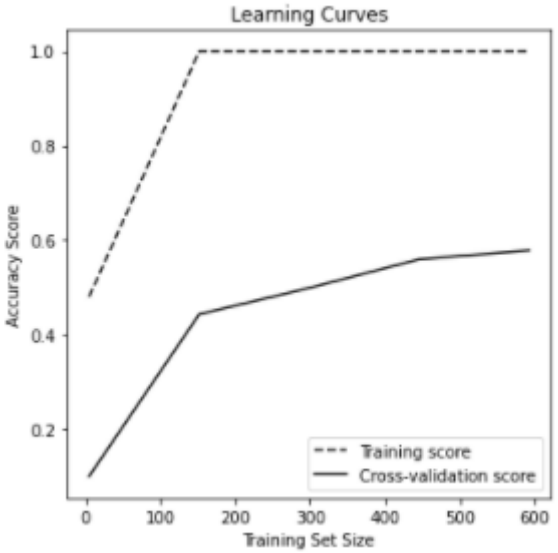

Figure 8.- Learning curve of XGBoost

- Feature Importance

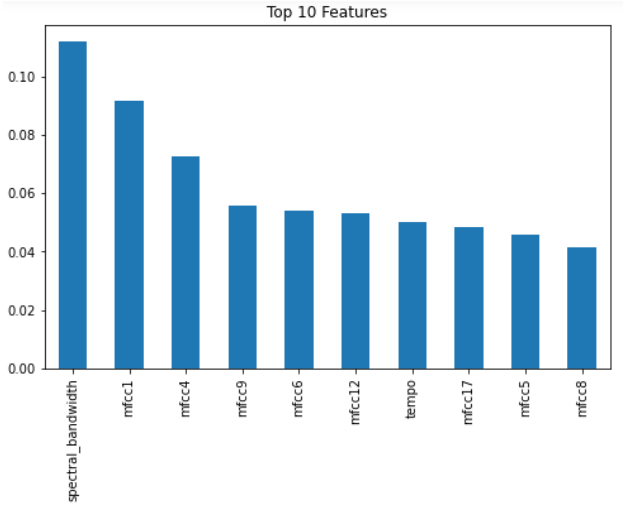

Figure 9.- Feature importance of XGBoost

3. Random Forest

- Confusion Matrix

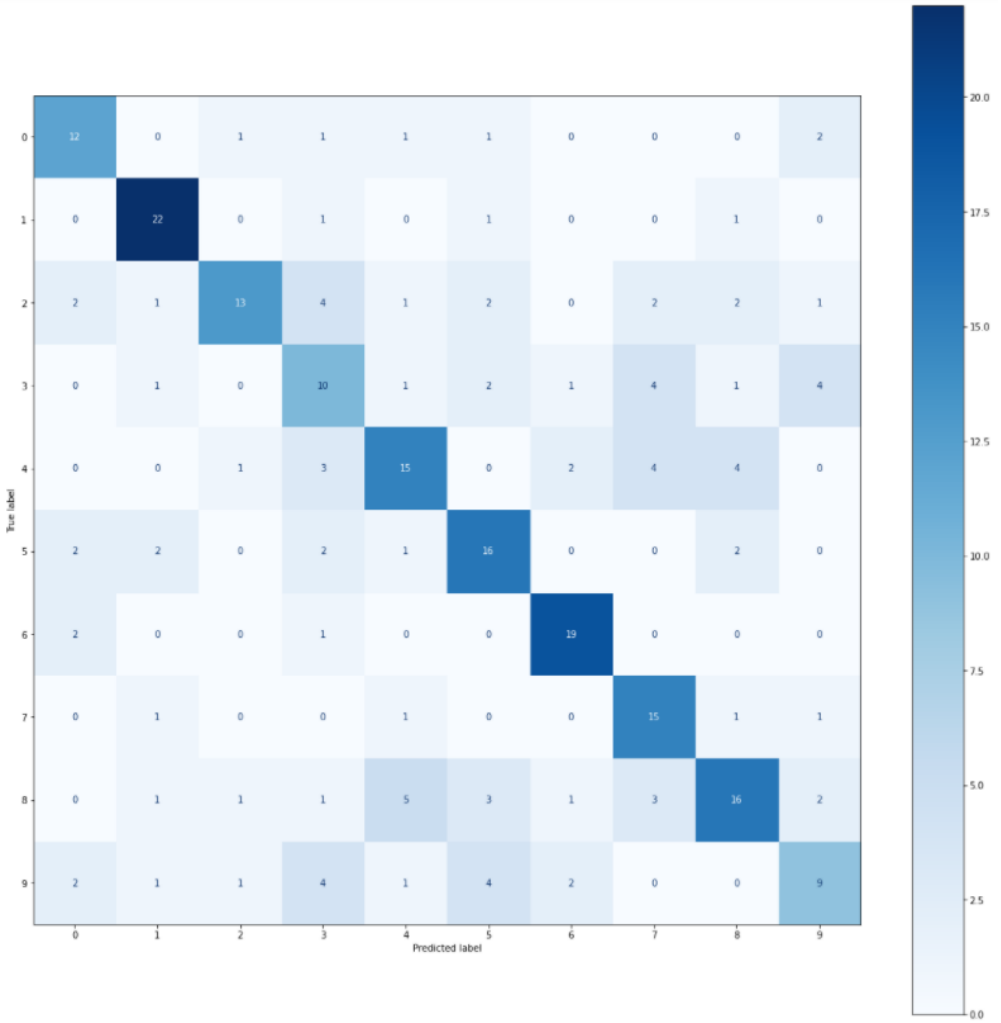

Figure 10.- Confusion matrix of random forest

Of all the Confusion Matrix we have seen, this is the best one. We can see that the times that the Random Forest Model predicts the number correctly more than 10 times in all of the cases, except one; this is a sign that the Model has an acceptable Accuracy. We can also notice that the Model predicts in a more average way, having a high number of correct predictions in all of the numbers, and not only in two numbers like the last Model.

- Learning curve

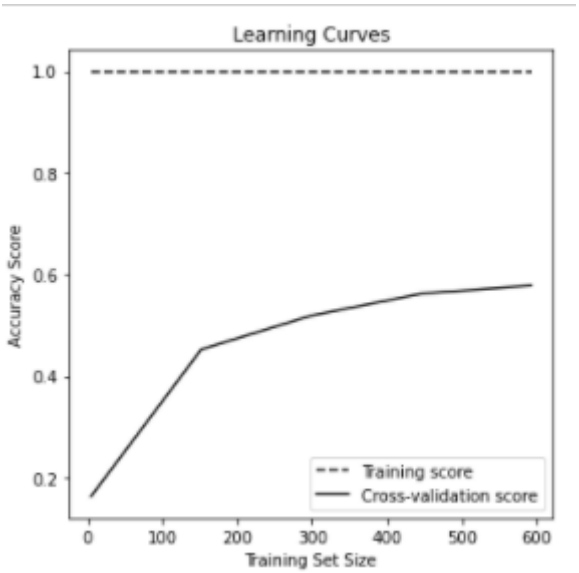

Figure 11.- Learning curve of random forest

- Feature importance

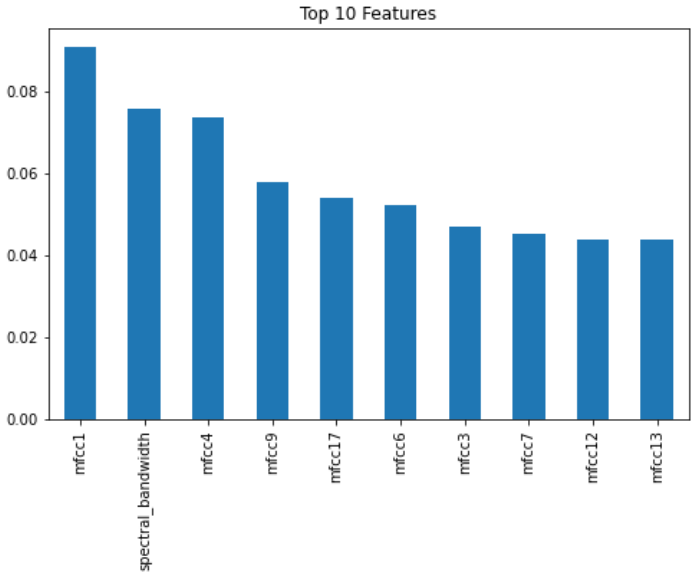

Figure 12.- Feature importance of random forest

	precision	recall	f1-score	support
blues	0.67	0.60	0.63	20
classical	0.88	0.76	0.81	29
country	0.46	0.76	0.58	17
disco	0.42	0.37	0.39	27
hiphop	0.52	0.58	0.55	26
jazz	0.64	0.55	0.59	29
metal	0.86	0.76	0.81	25
pop	0.79	0.54	0.64	28
reggae	0.48	0.59	0.53	27
rock	0.38	0.47	0.42	19
accuracy			0.60	247
macro avg	0.61	0.60	0.60	247
weighted avg	0.62	0.60	0.60	247

Table 2.- Evaluation of the target

4. Random Forest Second Iteration

- Confusion Matrix

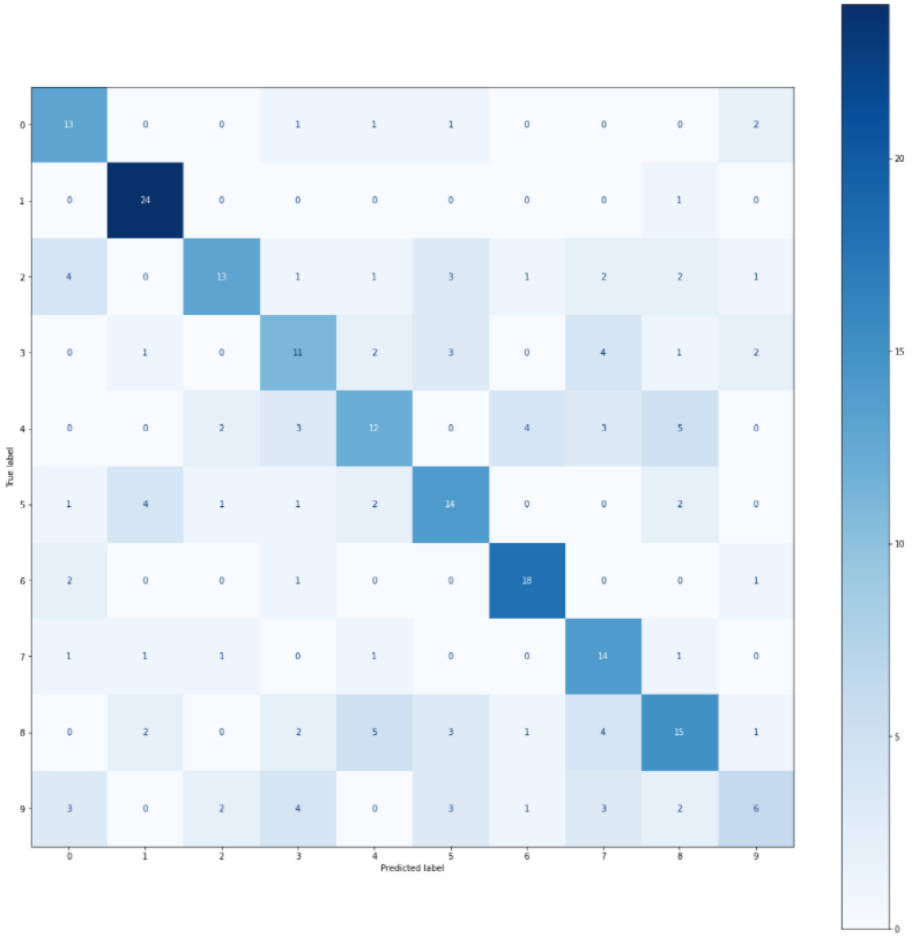

Figure 13.- Confusion matrix of random forest, second iteration

- Learning curve

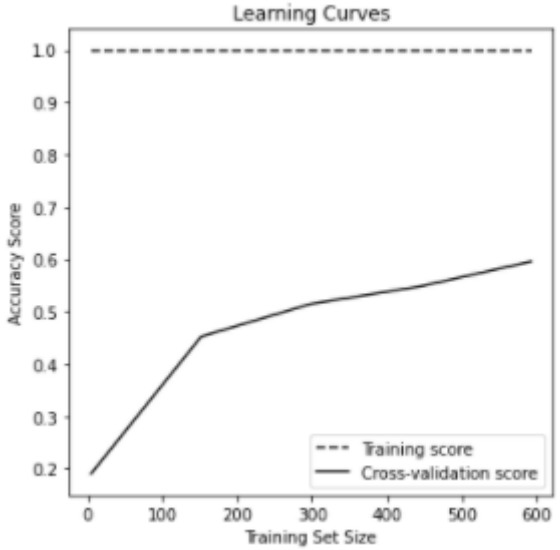

Figure 14.- Learning curve of random forest, second iteration

	precision	recall	f1-score	support
blues	0.72	0.54	0.62	24
classical	0.96	0.75	0.84	32
country	0.46	0.68	0.55	19
disco	0.46	0.46	0.46	24
hiphop	0.41	0.50	0.45	24
jazz	0.56	0.52	0.54	27
metal	0.82	0.72	0.77	25
pop	0.74	0.47	0.57	30
reggae	0.45	0.52	0.48	29
rock	0.25	0.46	0.32	13
accuracy			0.57	247
macro avg	0.58	0.56	0.56	247
weighted avg	0.62	0.57	0.58	247

Table 3.- Evaluation of the target

As seen in the learning curves, every model overfits to some degree, having high variance and low bias. This means that the models do not generalize well to the test data and have too high accuracy with the training data. Specifically for the random forest, it overfits as its training score has .4 less accuracy than the cross validation score, but we consider that its accuracy of .6 is good enough. A way to decrease the variance would be to have even more data. We also tested decreasing the features and scaling the data but this had no impact in its results.

Conclusion:

The genres that are best predicted by our model are classical and metal (both with f1-score of .81). This could mean that the songs from these genres are similar to each other, being easier to predict. The genres with the worst f1-scores are disco and rock, this means that those genres are hard to predict with our model.

The best model we found was the Random Forest Classifier, it had an accuracy and a f1-score of .59.
The most important features for this model are “mfcc1”, “spectral bandwidth” and “mfcc4”. We can notice that in the beginning of the project we expected spectral bandwidth to be an important feature to identify the music genre and at the end of the project we confirmed it.