Music Genre Classification

Project Report

GitHub Repository

Dheeraj Sai Siramsetty

2024

Abstract

This report describes our experience with building a classifier for music genre classification. A data set containing music clips of 10 different genres is provided. Various models have been implemented to classify genres such as neural netowrks, SVC, Ensemble learnimg methods etc.

Contents

1 Introduction

1.1 About

Music experts have been trying for a long time to understand sound and what differentiates one song from another. What makes a sound different from another. How different frequencies combine together to form such pleasant sounds. This is our trial at trying to understand the essence of music and differentiate various sounds.

1.2 Dataset Description

The dataset contains music from 10 different genre and has 100 audio sample in each genre. It also includes 2 csv files with various parameters with length of audio being 3 seconds in one and 30 seconds in another. The dataset also contains the visual images of all the audios which was possible with the help of the mel-coefficients.

2 Dataset Processing

2.1 Pre-Processing

The dataset has been loaded onto the colab file, each song has been trimmed to a length of 30 seconds in order to ensure uniformity throughout the data. Then the 30 seconds audio file is further cut into 3 seconds which in turn increased the dataset size drastically by almost 10 times ensuring there is enough left to extract proper information from.

2.2 Feature Extraction

In order to ensure that we were capturing the most information from each and every song we performed 2 different types of feature extraction:

2.2.1 Standard Feature Extraction:

In this part of our data extraction, we extracted information that is traditionally extracted from sounds.

- Spectral Rolloff
- Spectral Bandwidth
- Zero Crossing Rate
- Chroma STFT
- Spectral Rolloff
- Mel Gibson Coefficients

- Tempo
- PLP
- Harmony
- RMS

From each of these, we calculated the mean and variance in order to gain the maximum information and thus preventing information loss and also ensuring the uniformity of data thus enabling ease of calculations while testing out models.

2.2.2 Power Spectral Analysis

We know that each and every audio signal is a combination of various frequencies. These frequencies come together and combine with each other in periodic and random ways in order make a sound. Taking this into account we decided to make use of such patterns in order to classify the data. In order to perform the extraction of such data, we were required to perform fourier transform, but since songs are a combination of signals which change with respect to time, we ue short time fourier transfor, or stft in short which enables us to monitor the change in frequencies over a duration of time. After obtaining the said transformation, we obtained 519 fourier transformations over each audio clip. We now have to derive the power from each of these fourier transformations and then with the help of a library called FOOOF we separate and extract the periodic and aperiodic elements of the audio. The maths behind operations performed can be found as below:

Similar to the previous data extra, we calculated the mean and average of each of the 519 components and added it to our dataframe. This ensures that the core information is retained while removing excess noise.

2.3 Dataset Visualisation

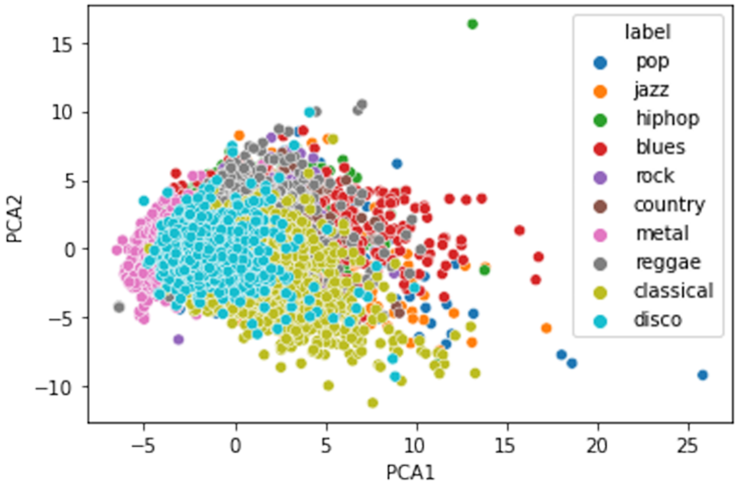

Figure 1: Visualization of PCA of 2 components

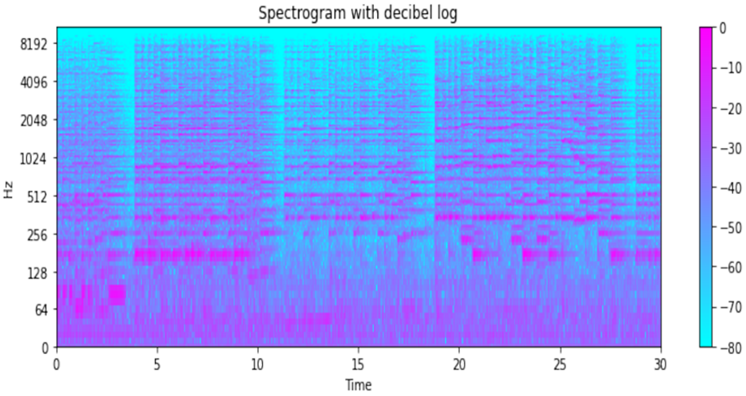

Figure 2: Spectrogram of a song from the dataset

3 Machine Learning Models

After performing feature extraction we needed a model sophisticated enough to processing the information well. We had to try multiple models like Random Forest Classifier, Logistic Regressor, NN etc as each of these models have different weak and strong suites and depending on the dataset give out various results. The models we tried are as below:

3.1 Basic Models:

3.1.1 Random Forest Classifier

Random Forest Classifiers use boosting ensemble methods to train upon various decision trees and produce ag- gregated results. It is one of the most used machine learning algorithms.

3.1.2 XGBoost

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible, and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting that solves many data science problems in a fast and accurate way.

3.1.3 Light GBM

Light Gradient Boost is is a gradient boosting framework. LightGBM algorithm grows vertically meaning it grows leaf-wise and other algorithms grow level-wise. LightGBM chooses the leaf with large loss to grow. It can lower down more loss than a level wise algorithm when growing the same leaf.

3.1.4 Support Vector Machine

In SVM , data points are plotted into n-dimensional graphs which are then classified by drawing hyperplanes.

3.1.5 Neural Networks

Traditionally, NN models are used in Image processing and recognition. We thought that the complexity of this dataset could be easily captured by such complicated networks and hence, we tried various architectures. While training the models on our dataset, we also wanted to understand how changing the architecture of the NN model affect the results:

3.1.6 Model 1

In Model 1, we begin the dense layer with a size of 256 and end with softmax layer and in between we add dropout layers with 0.3 blocked.

3.1.7 Model 2

In Model 2, we begin the dense layer with a size of 512 and end with softmax layer and in between we add dropout layers with 0.3 blocked. We wanted to observe how the increasing the parameters affects the NN network and we can see thar the model begins to train and learn better.

3.1.8 Model 3

In Model 3, we begin the dense layer with a size of 512 and end with softmax layer and without droupout layers. We can see that without dropout layer, the model overfits very fast and thus giving a bad validation loss.

3.1.9 Model 4

In Model 4, we begin the dense layer with a size of 1024 and end with softmax layer and in between we add dropout layers with 0.3 blocked. We wanted to observe how the increasing the parameters affects the NN network and we can see that the model begins to train and learn better and gives better result but if allowed to run for longer epochs tends to overfit drastically.

3.2 Custom Models

Despite such good results on the various models, we were not satisfied with the result. And hence we though that in order to properly evaluate such a complex dataset, we had to use more mathematically sophisticated models and also, we wanted to use Ensemble learning as it captures the best of every model it is comprised of. Hence, we designed a custom models.

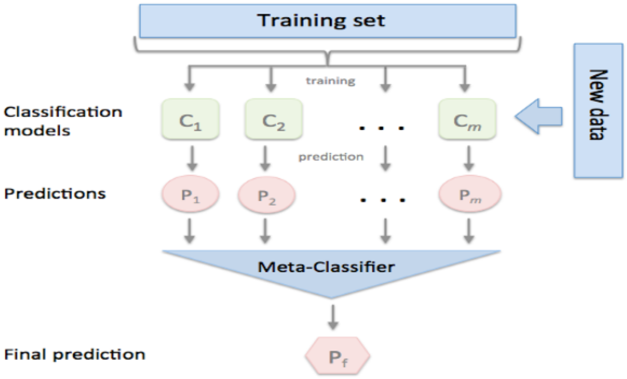

Figure 3: Meta Classifier

3.2.1 Custom Model 1

This is a stacking based model. In stacking , various models are trained on the dataset and then the output, without any processing, is passed through a meta classifier which decides which output is correct and then gives output.

3.2.2 Custom Model 2

After looking at the confusion matrix of the basic models we observed that few classifiers are extremely good at classifying few classes but performs horribly in other classes. In standard stacking the data is passed as it is and all of the classifiers are given equal weight age but in our model In custom model 2, before the predicted probability from the models is passed to the meta classifier, using the principle of weighted averages we tinker with the data. We do this by multiplying the validation accuracy with the given model.

4 Classifier Comparision

4.1 Basic Models

Model	Accuracy on Extracted Data	Accuracy on Given CSV File
Random Forest	75.35	85.48
XGBoost	72.32	77.97
LightGBM	71.97	78.32
Support Vector Machine	77.37	85.58
Model 1	81.83	91.07
Model 2	83.18	-
Model 3	83.23	-
Model 4	85.34	92.38

Table 1: Performance of Basic Models

4.2 Custom Models

Model	Random Forest	XGBoost	LightGBM	SV Classifier	Custom Model 1	Custom Model 2
Accuracy	75.35	72.32	71.97	77.37	75.02	78.97

Table 2: Comparison of Custom Models with Basic Models

5 Conclusion

Various custom and inbuilt classifier have been trained and tuned. The best performance is shown by neural network models with highest accuracy score of 85% and where as ensemble custom methods performed better than their constituent models. Observing the accuracy rates, we can say that Music like most thing in nature has pattern and although a little elusive, the pattern can be captured used to separate music by machine learning.

References

1. Y.-H. Cheng, P.-C. Chang and C.-N. Kuo, ”Convolutional Neural Networks Approach for Music Genre Classi- fication,” 2020 International Symposium on Computer, Consumer and Control (IS3C), 2020, pp. 399-403, doi: 10.1109/IS3C50286.2020.00109.

2. https://github.com/mlachmish/MusicGenreClassification

3. https://librosa.org/doc/main/index.html

4. https://fooof-tools.github.io/fooof/