NAML Project Report - Group 14

Music Genre Classification using
k-Nearest Neighbours
Nearest Centroid
Multiclass SVM
-

by Silvia Marino (codice persona) and Francesco Panebianco (10632465)
DEIB - Politecnico di Milano

February 2022

Contents

1 Introduction **2**
Scope 2
The Dataset 2
Feature Extraction 4
Dataset Visualization 5

2 Classifiers **7**
k-Nearest Neighbours Classifier 7
Nearest Centroid Classifier 7
Multiclass SVM Classifier 7

1 Introduction

Scope

The scope of this project is to create a music genre classifier using the machine learning algorithms *k-Nearest Neighbours* and *Multiclass SVM*, which were assigned to our group. The project is part of the evaluation of the “Numerical Analysis for Machine Learning” course, which is part of the first semester of the first year of Master’s Degree in Computer Science and Engineering at Politecnico di Milano. Given the similarities between *k-Nearest Neighbours* and *Nearest Centroid*, we chose to implement the latter as well, comparing its performance to the former, even though it is outside the specification of the project.

The Dataset

The dataset assigned to our project is the notorious *GTZAN Genre Collection*[3], which contains 100 different extracts from 10 different music genres, provided in .wav (Waveform Audio File Format). The genres considered are:

- Blues
- Classical
- Country
- Disco
- Hip Hop
- Jazz
- Metal
- Pop
- Reggae
- Rock

As it can be seen, genres that share similarities are included in the dataset (e.g. Blues and Jazz), but also dramatically different types of music such as Rock and Classical, which we expect the algorithms to classify with higher precision.

The *.wav* file format used by the dataset is the most common uncompressed audio file format in Microsoft Windows systems for. It was developed by IBM and Microsoft, for storing an audio bitstream on PCs[1].

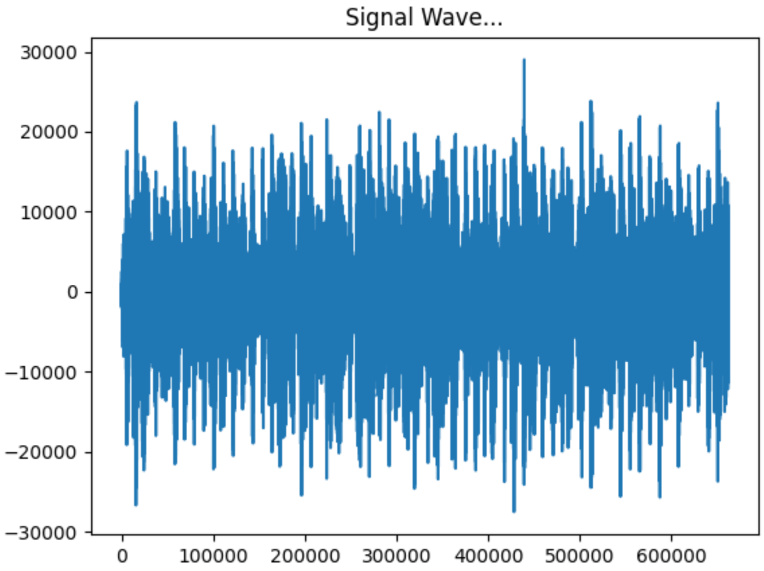

Figure 1: Plotted waveform of blues.00000.wav

This kind of files are managed in Python by using the *wave* module, which provides convenient functions to work with the WAV sound format. In our case, the audio samples are provided as 22050Hz Mono 16-bit 30 second tracks[3], which are easily transformed into a numpy array of *int16*.

The picture above is the result of the following code:

```
import wave
import matplotlib.pyplot as plt
import numpy as np

test_file = wave.open('genres/blues/blues.00000.wav', 'rb')
# Extract Raw Audio from Wav File
signal = test_file.readframes(-1)
signal = np.frombuffer(signal, dtype='int16')
test_file.close()

plt.figure(1)
plt.title("Signal Wave...")
plt.plot(signal)
plt.show()
```

Feature Extraction

For what concerns the features required to classify the genres, our first choice was the triplet *Zero Crossing Rate*, *Average Energy* and *Silent Ratio* as defined in one of the provided papers[2]. The extracted features were collected and exported in a csv file to avoid having to extract the feature multiple times, as the complete process took approximately 45 minutes on *Google Colab.*

Zero Crossing Rate

It indicates the frequency of signal amplitude sign change, which is in some way related to the average signal frequency. In practice, it is a key feature to classify percussive sounds as it's often correlated with the beat. ZCR records how many waves have passed for a certain time, by giving a positive amplitude a positive value (1) and a negative amplitude a negative value (-1). The implemented formula is as follows:

$$\mathrm{ZCR} = \frac{\sum_{n=1}^{N} \left| \mathrm{sgn}\ x(n) - \mathrm{sgn}\ x(n-1) \right|}{2N} \tag{1}$$

where sgn x is the *sign function.*

Average Energy

It indicates the loudness of the audio signal as a whole, being the average of the square amplitude of the audio signal.

$$\mathrm{E}_{\mathrm{avg}} = \frac{\sum_{n=0}^{N-1} x(n)^2}{N} \tag{2}$$

In practice, it corresponds to the loudness of the music within the time slice where it is measured.

Silent Ratio

It indicates the proportion of the sound piece that is considered to be *silent.* Silence is defined as a period within which the absolute value of amplitude is below a certain threshold. If we indicate

$$SR = \frac{\sum_{x(n)<\mathrm{thr}} 1}{N} \tag{3}$$

In our implementation, silence threshold is derived from the average energy as follows...

$$\mathrm{thr} = 0.8\sqrt{E_{\mathrm{avg}}} \tag{4}$$

Dataset Visualization

After exporting the feature dataset, we visualized it using built-in functions from *pandas* and *seaborn*. Full dataset visualization is available in a the Jupiter Notebook called *NAML_Project_Data_Visualization.ipynb*.

```
                ZCR  AVERAGE_ENERGY  SILENT_RATIO
count   1000.000000    1.000000e+03   1000.000000
mean       0.103768    2.631159e+07      0.827613
std        0.041886    2.396840e+07      0.024517
min        0.021714    6.178678e+04      0.767521
25%        0.070328    9.781419e+06      0.810039
50%        0.099618    1.834412e+07      0.824969
75%        0.132136    3.768007e+07      0.842175
max        0.275001    1.777716e+08      0.926341
```

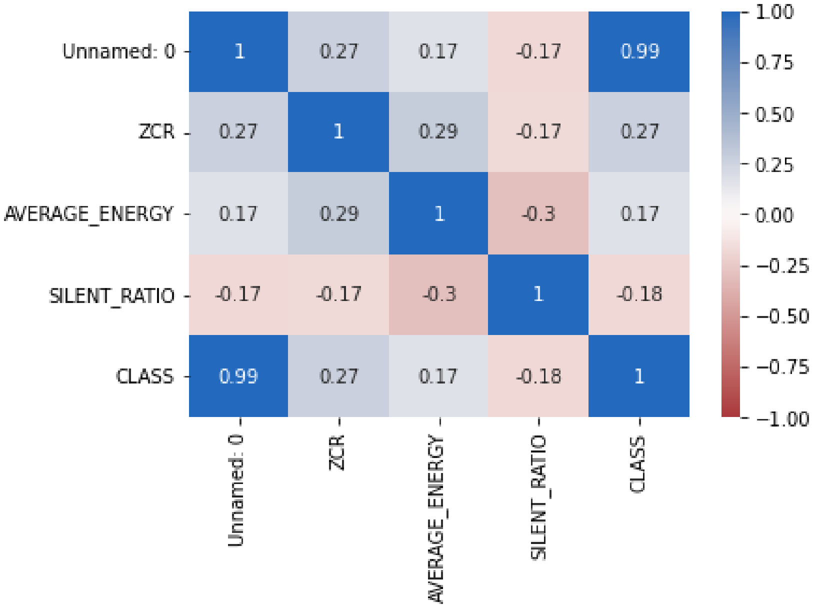

Figure 2: Annotated Heatmap of the correlation of extracted features

From this visualization, it is evident that the three features are independent enough to be non-redundant in the classification.

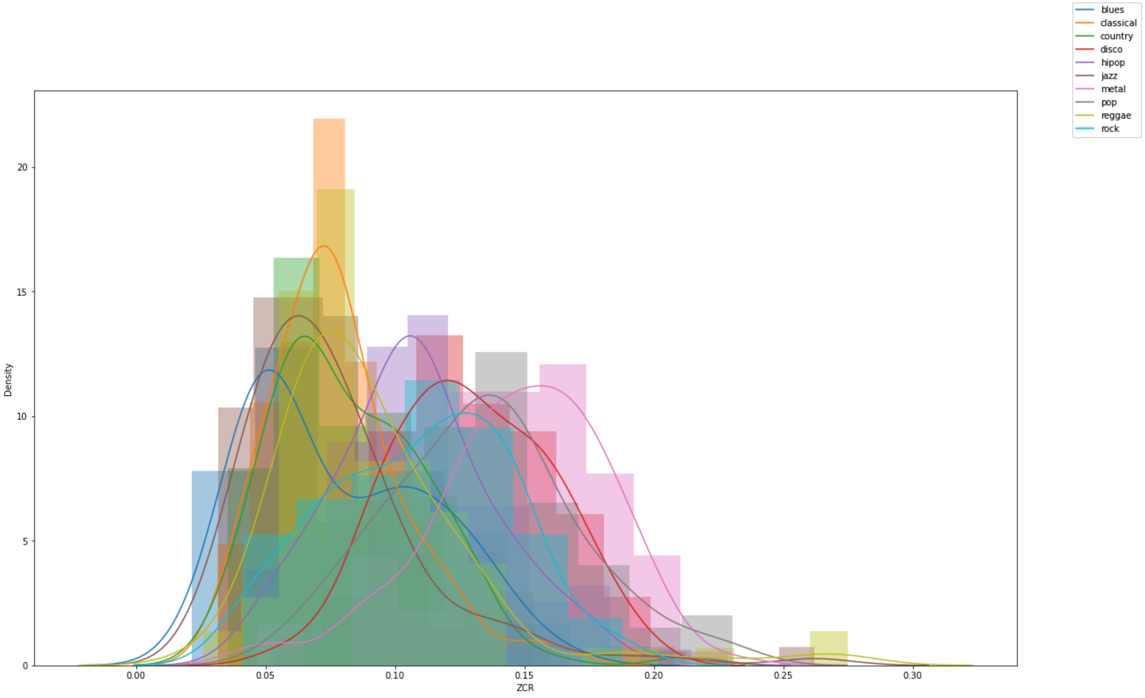

Figure 3: Distribution Plot of ZCR feature

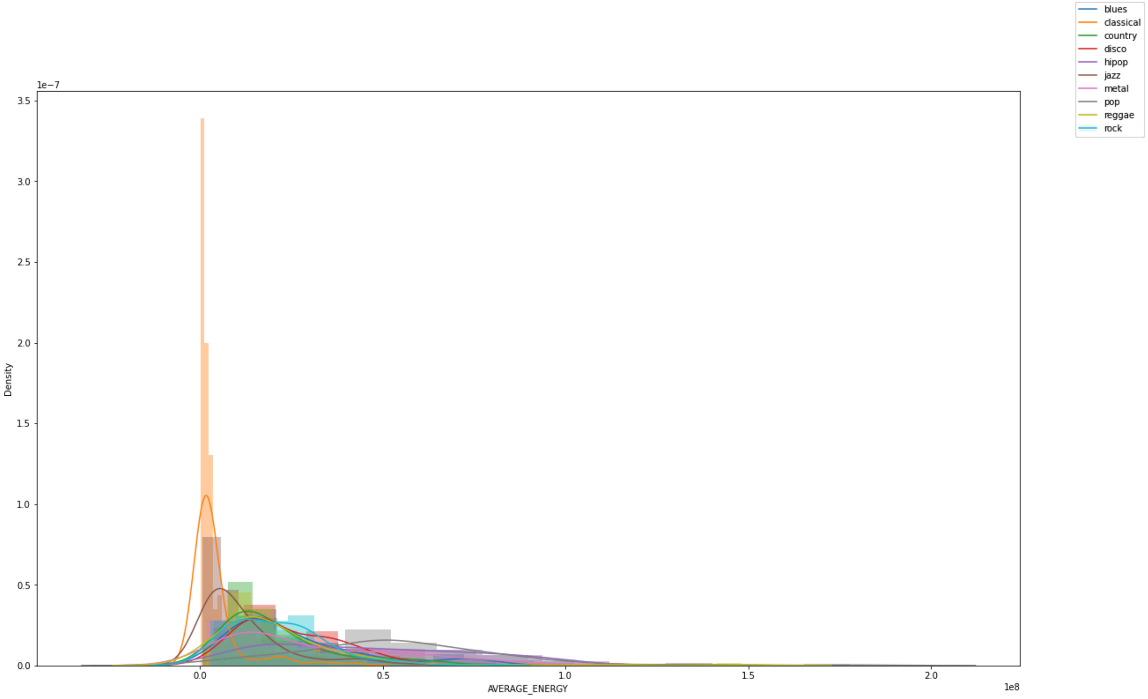

Figure 4: Distribution Plot of Average Energy feature

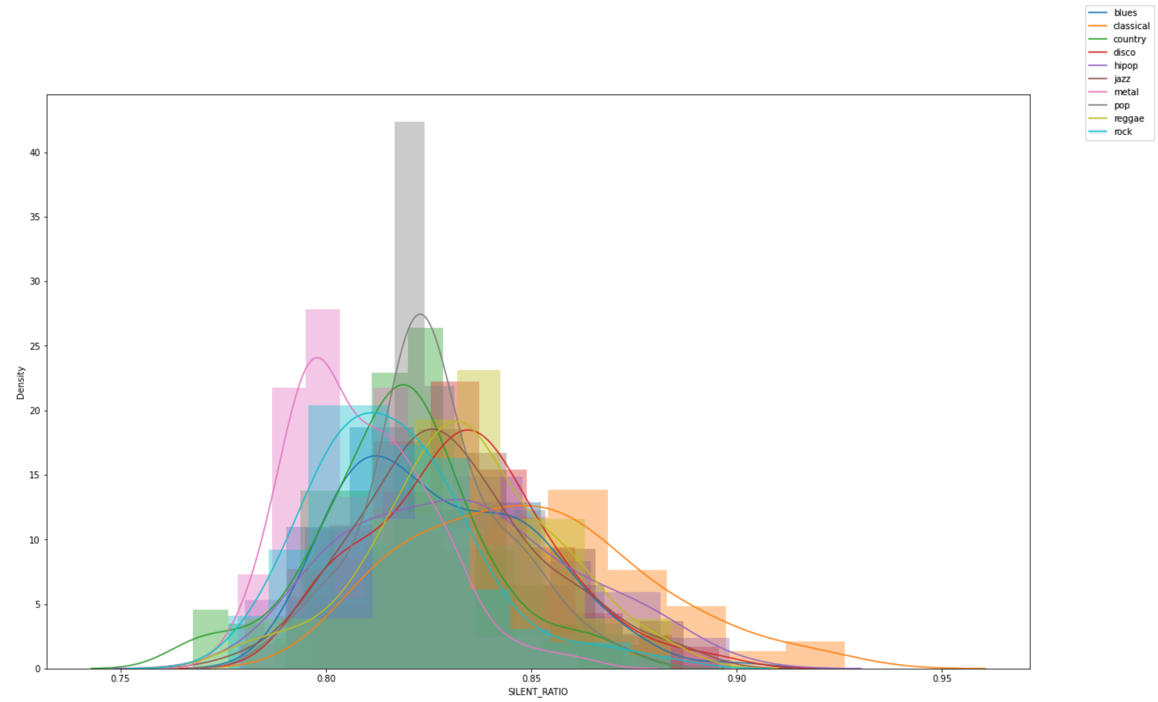

Figure 5: Distribution Plot of Silent Ratio feature

2 Classifiers

k-Nearest Neighbours Classifier

In this section it will be explained which methodologies have been used to implement the music genre classifier and how the code has been structure to achieve the results exposed at the end of the report

Nearest Centroid Classifier

Multiclass SVM Classifier

References

[1] Fleischman E. *WAVE and AVI Codec Registries*. 1998. URL: `https://datatracker.ietf.org/doc/html/rfc2361`.

[2] Tamatjita Elizabeth Nurmiyati and Mahastama Aditya Wikan. "Comparison of Music Genre Classification Using Nearest Centroid Classifier and k-Nearest Neighbours". In: *International Conference on Information Management and Technology (ICIMTech)* (2016), pp. 118–123. DOI: `978-1-5090-3352-2`.

[3] Leben Jakob. *Music Analysis, Retrieval and Synthesis for Audio Signals*. URL: `http://marsyas.info/downloads/datasets.html`.