Music genre classification project from a machine learning approach

Youssef Ahmed Hassan Aly Rashid
Bachelor study , exchange program
yahara@kth.se

INTRODUCTION:

The study of music has been over the years very complicated yet interesting field of study.
A music genre is a great way to classify music and facilitate its study. Unfortunately, categorizing music in genres is a relative matter that depends from someone to another or from different eras but that hasn't stopped people from categorizing it to make it easier to deal with music on the daily basis. Today, we use machine learning algorithms to do the classification based on the music features (e.g: rhythmatic and timberal features).

METHOD:

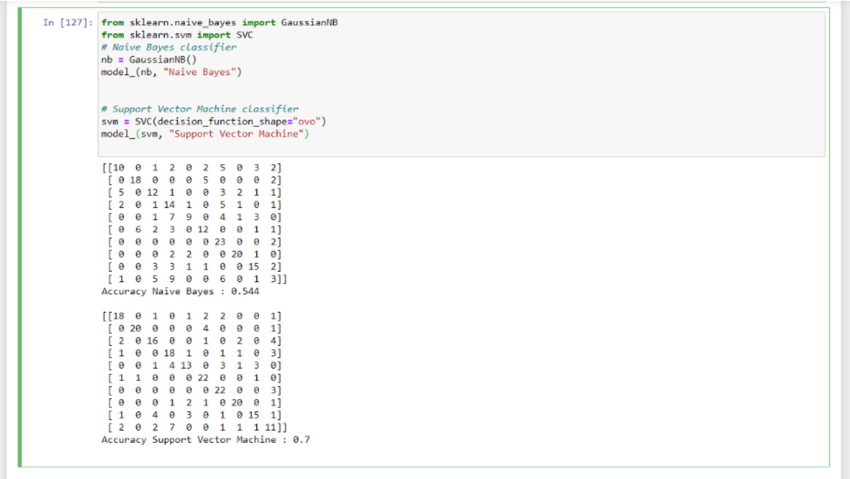
The method I used to classify the music was based on extracting the features below using librosa :

1. Chroma function
2. Spectral centroid
3. Spectral bandwidth
4. Spectral rolloff
5. RMS
6. Zero crossings
7. Mel Frequency Cepstral Coefficients

I used the data set GTZAN which consists of 1000 songs classified into 10 genres with a 100 songs each. Each song is a 30 secs long.
I've included the extracted features into a CSV file and made into a pandas dataframe to be easy to access and deal with. The data I used was partitioned randomly into Train, validation & test. Also, I used the fault filtered partition as well to see the difference.
First, I tried some basic supervised machine learning classifiers (Naive Bayes & Support Vector Machine) then I implemented an ANN and trained the model (once without the VAL. And once using it).At the end, I made the program tells me what genre it classified each song in the test set to be.

RESULTS:

1. The Naive Bayes classifier performed as expected with a low accuracy but surprisingly the Support Vector Machine has performed almost as good as the ANN



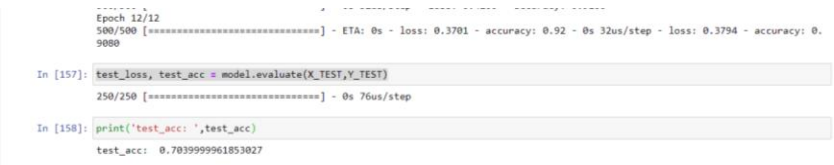
```
In [127]: from sklearn.naive_bayes import GaussianNB
          from sklearn.svm import SVC
          # Naive Bayes classifier
          nb = GaussianNB()
          model_(nb, "Naive Bayes")


          # Support Vector Machine classifier
          svm = SVC(decision_function_shape="ovo")
          model_(svm, "Support Vector Machine")

          [[10  0  1  2  0  2  5  0  3  2]
           [ 0 18  0  0  0  5  0  0  0  2]
           [ 5  0 12  1  0  0  3  2  1  1]
           [ 2  0  1 14  1  0  5  1  0  1]
           [ 0  0  1  7  9  0  4  1  3  0]
           [ 0  6  2  3  0 12  0  0  1  1]
           [ 0  0  0  0  0  0 23  0  0  2]
           [ 0  0  0  2  2  0  0 20  1  0]
           [ 0  0  3  3  1  1  0  0 15  2]
           [ 1  0  5  9  0  0  6  0  1  3]]
          Accuracy Naive Bayes : 0.544

          [[18  0  1  0  1  2  2  0  0  1]
           [ 0 20  0  0  0  4  0  0  0  1]
           [ 2  0 16  0  0  1  0  2  0  4]
           [ 1  0  0 18  1  0  1  1  0  3]
           [ 0  0  1  4 13  0  3  1  3  0]
           [ 1  1  0  0  0 22  0  0  1  0]
           [ 0  0  0  0  0  0 22  0  0  3]
           [ 0  0  0  1  2  1  0 20  0  1]
           [ 1  0  4  0  3  0  1  0 15  1]
           [ 2  0  2  7  0  0  1  1  1 11]]
          Accuracy Support Vector Machine : 0.7
```

2. I trained the model using the training dataset and then I did the training/validation



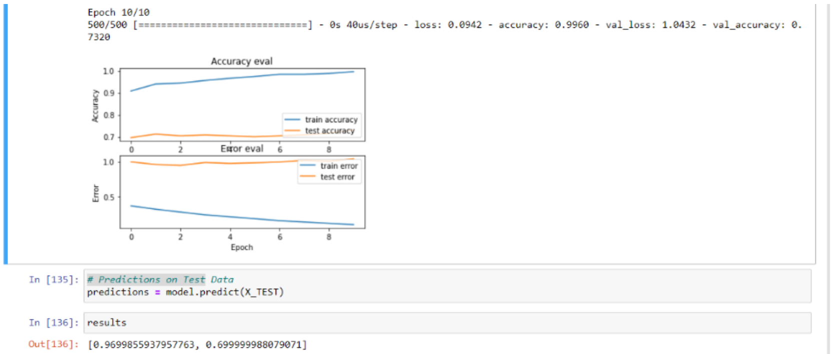
```
          Epoch 12/12
          500/500 [==============================] - ETA: 0s - loss: 0.3701 - accuracy: 0.92 - 0s 32us/step - loss: 0.3794 - accuracy: 0.
          9080

In [157]: test_loss, test_acc = model.evaluate(X_TEST,Y_TEST)

          250/250 [==============================] - 0s 76us/step

In [158]: print('test_acc: ',test_acc)

          test_acc:  0.7039999961853027
```



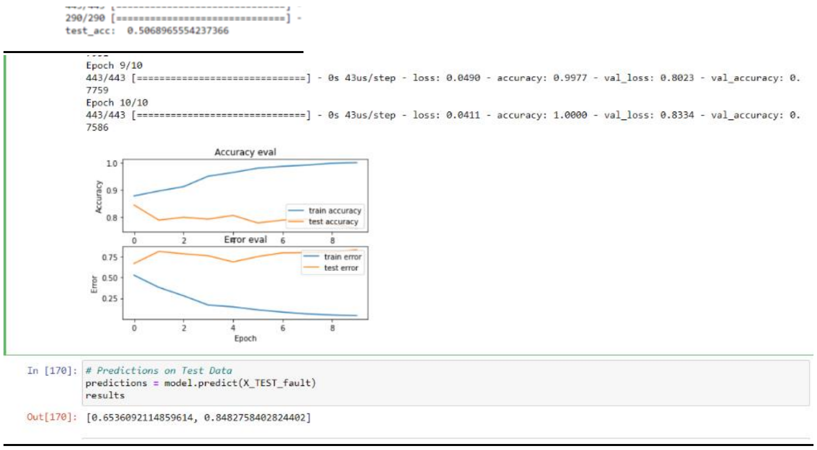
```
          Epoch 10/10
          500/500 [==============================] - 0s 40us/step - loss: 0.0942 - accuracy: 0.9960 - val_loss: 1.0432 - val_accuracy: 0.
          7320
```

```
In [135]: # Predictions on Test Data
          predictions = model.predict(X_TEST)

In [136]: results

Out[136]: [0.9699855937957763, 0.699999988079071]
```

3. The previous process was repeated again using the fault filtered partitions



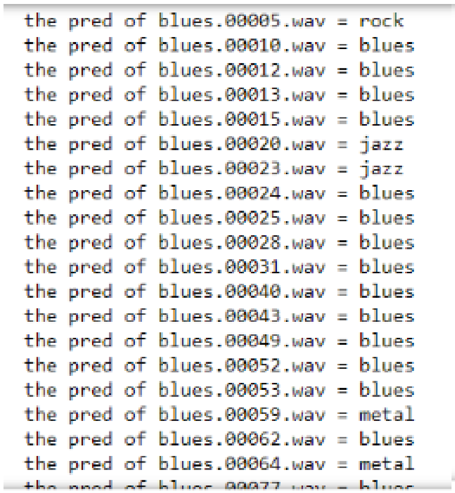
```
290/290 [==============================] -
test_acc:  0.5068965554237366
```

```
Epoch 9/10
443/443 [==============================] - 0s 43us/step - loss: 0.0490 - accuracy: 0.9977 - val_loss: 0.8023 - val_accuracy: 0.
7759
Epoch 10/10
443/443 [==============================] - 0s 43us/step - loss: 0.0411 - accuracy: 1.0000 - val_loss: 0.8334 - val_accuracy: 0.
7586
```

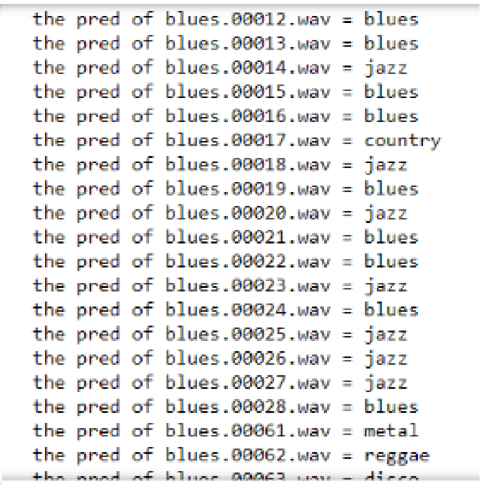
```
In [170]: # Predictions on Test Data
          predictions = model.predict(X_TEST_fault)
          results

Out[170]: [0.6536092114859614, 0.8482758402824402]
```

4. The program classified each song in the test set as shown in the fig below: (fig.1 the random partion , fig2. Fault partion)



```
the pred of blues.00005.wav = rock
the pred of blues.00010.wav = blues
the pred of blues.00012.wav = blues
the pred of blues.00013.wav = blues
the pred of blues.00015.wav = blues
the pred of blues.00020.wav = jazz
the pred of blues.00023.wav = jazz
the pred of blues.00024.wav = blues
the pred of blues.00025.wav = blues
the pred of blues.00028.wav = blues
the pred of blues.00031.wav = blues
the pred of blues.00040.wav = blues
the pred of blues.00043.wav = blues
the pred of blues.00049.wav = blues
the pred of blues.00052.wav = blues
the pred of blues.00053.wav = blues
the pred of blues.00059.wav = metal
the pred of blues.00062.wav = blues
the pred of blues.00064.wav = metal
the pred of blues.00077.wav = blues
```



```
the pred of blues.00012.wav = blues
the pred of blues.00013.wav = blues
the pred of blues.00014.wav = jazz
the pred of blues.00015.wav = blues
the pred of blues.00016.wav = blues
the pred of blues.00017.wav = country
the pred of blues.00018.wav = jazz
the pred of blues.00019.wav = blues
the pred of blues.00020.wav = jazz
the pred of blues.00021.wav = blues
the pred of blues.00022.wav = blues
the pred of blues.00023.wav = jazz
the pred of blues.00024.wav = blues
the pred of blues.00025.wav = jazz
the pred of blues.00026.wav = jazz
the pred of blues.00027.wav = jazz
the pred of blues.00028.wav = blues
the pred of blues.00061.wav = metal
the pred of blues.00062.wav = reggae
the pred of blues.00063.wav = disco
```

Discussion & conclusion :

- My model has performed around 70% accuracy using the mean and var. of 7 features and it could have performed better if it had different features
- The Support Vector Machine performed almost as good as the ANN model with less processing time. Which also means that working with deep neural network would be more efficient with bigger datasets
- The model was overfitted a no matter how the hyperparameters was changed.
- Some genres like reggae has performed better than the others which means that the algorithms doesn't work equally with all the genres

References :

Tzanetakis, George & Cook, Perry. (2002). Musical Genre Classification of Audio Signals. IEEE Transactions on Speech and Audio Processing. 10. 293 - 302. 10.1109/TSA.2002.800560.

Basili, Roberto & Serafini, Alfredo & Stellato, Armando. (2004). Classification of musical genre: a machine learning approach..

GITHUB LINK :

https://github.com/YoussefRashid/music-genres-classification