iJRASET
International Journal For Research in
Applied Science and Engineering Technology

Volume: 9 Issue: VI Month of publication: June 2021

DOI: https://doi.org/10.22214/ijraset.2021.35381

# Music Genre Classifier using Machine Learning Algorithms

Sparsh Nagpal
*Thadomal Shahani Engineering College*

***Abstract: Audio data extraction and analysis is important and less explored compared to other forms of data. Here we use an Audio dataset (GTZan) to extract musical information and categorize the musical genre based on the parameters of audio. We compared the study on seven Machine learning algorithms and tested on unvisited user data to see the model performance seeing the algorithms' accuracy ranging from 45% to 87%.***

## I. INTRODUCTION

The growth of musical data, user personalization and the exploration of its usages has grown in the past few years. The applications like Spotify, Gaana, I-Music utilize this data from users' music history and provide recommendations based on analysis done on such musical data.

Categorization of music into different fields of music is a very integral step of this process. The research here shows an approach to collect musical data from the user's input audio, different musical features like Mel-spectrum features, harmony, chroma etc were collected and machine learning algorithms are applied to those.

We use machine learning algorithms like KNN, Ensemble algorithms, Logistic Regression for the data and try to classify it into 10 musical genre labels.

The paper discusses various steps of the whole procedure from data augmentation, hyperparameter tuning, data extraction and analysis.

## II. DESIGN

### *A. Literature Survey*

*Tzanetakis and Cook (2002) [1]* were one of the first contributors in this field of musical data analysis. The GTZAN dataset was created by them and is to date considered as a standard for genre classification. Scaringella and Zoia(2005) gave a comprehensive survey of both features and classification techniques used in genre classification. Most of the work deals with supervised learning approaches. *Riedmiller(2012)[2]* et al use unsupervised learning to create a dictionary of features gives a detailed account of the evaluation of previous work on genre classification. Along with reference to these papers, the *official Librosa Documentations [3]* were referred for the implementation.

### *B. Diagram*

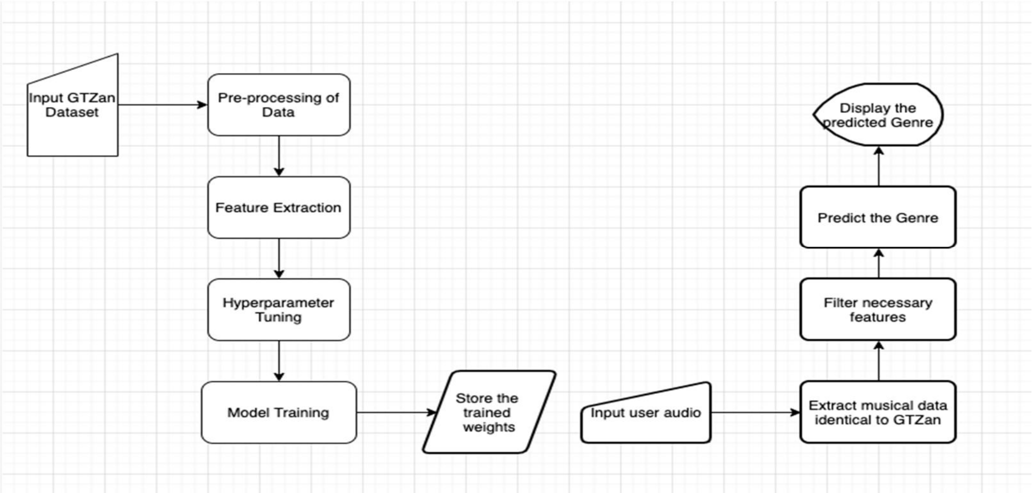


Figure: 2.2

## III. DATASET

The main musical datasets researched were: Million Song Dataset, GTZan Dataset and Midi Files. *Tzanetakis and Cook (2002) [1]* were the contributors to the GTZan dataset consisting of 10000 30 second songs with 10 different genre labels including Pop, Rock, Jazz, Reggae, etc. The dataset we used consisted of each music and its different musical features. The features included. Mel-frequency cepstral coefficients (MFCCs), spectral contrast, spectral roll-off and chroma features were some of the features. These features were displayed in a tabular form of 55 features for 30-second songs. The dataset also consisted of the same audio dataset for the songs split in 3 seconds each. The 3 seconds set of data increases the dataset by 10 times and much better accuracy. The musical audio sounds are given in the format of ".wav" which are converted in NumPy arrays later for computation purposes.

## IV. METHODOLOGY

### *A. Data-Preprocessing*

The audio data is converted into an array format. This array is made sure that it gets void of all the blank spaces in the audio which is limited to a duration of 30 seconds to have a fixed state of data. This 30 seconds audio data is now converted into 10 smaller arrays of 3 seconds data each to have a much wider dataset for our program in order to give a much better performance. The input parameters data was created in the form of a pandas data frame for easier processing.

### *B. Features*

There are a total of 55 parameters given in the GTZan dataset. The parameters are various musical features t help to characterize different genres. These parameters include:

1) *Mel-frequency Cepstral Coefficients (MFCC):* Introduced in the early 1990s by Davis and Mermelstein, MFCCs have been very useful features for tasks such as speech recognition *(Davis and Mermelstein, 1990)* [5]. First, the Short-Time Fourier-Transform (STFT) of the signal is taken with n fft=2048 and hop size=512 and a Hann window. Next, we compute the power spectrum and then apply the triangular MEL filter bank, which mimics the human perception of sound. This is followed by taking the discrete cosine transform of the logarithm of all filterbank energies, thereby obtaining the MFCCs. The parameter n Mels, which corresponds to the number of filter banks, was set to 20 in this study.
2) *Chroma Features:* This is a vector that corresponds to the total energy of the signal in each of the 12 pitch classes. (C, C#, D, D#, E, F, F#, G, G#, A, A#, B) *(Ellis, 2007) [6]*. The Chroma vectors are then aggregated and their mean and standard deviation is taken.
3) *RMSE Features*

The signal's energy is calculated as:

$$\sum_{n=1}^{k} |x(n)|^2$$

The Root Mean Square value can be calculated as:

$$\sqrt{\frac{1}{N}\sum_{n=1}^{N} |x(n)|^2}$$

RMSE is calculated frame by frame. We then take the average and standard deviation across all frames

4) *Spectral Centroid:* For each frame, this corresponds to the frequency around which most of the energy is centred *(Tjoa,2017) [7]*. It is a magnitude weighted frequency calculated as:

$$f_c = \frac{\sum_{k=0}^{n} S(k)f(k)}{\sum_{k=0}^{n} f(k)}$$

where S(k) refers to the spectral magnitude of the frequency of bin k and f(k) refers to a frequency corresponding to bin k

5) *Spectral Bandwidth:* The p-th order spectral bandwidth corresponds to the p-th order moment about the spectral centroid (Tjoa, 2017) [7] and is calculated as

[ k(S(k)f(k) - fc)p]1/p

$$[\sum_{k}(S(k)f(k) - f_c)^p]^{\frac{1}{p}}$$

For example, p= 2 is analogous to a weighted standard deviation

6) *Spectral Roll off:* This feature corresponds to the value of frequency below which 85% (this threshold can be defined by the user) of the total energy in the spectrum lies (Tjoa, 2017) [8]. For each of the spectral features described above, the mean and standard deviation of the values taken across frames is considered as the representative final feature that is fed to the model. The features described in this section would be used to train machine learning algorithms. The features that contribute the most to achieving a good classification performance will be identified and reported.
7) *Zero-Crossing:* Ratel A zero-crossing point refers to one where the signal changes sign from positive to negative *(Gouyon et al.,2000)* [8]. The 3-second audio data when converted to NumPy array is divided into smaller frames and the zero-crossing quantity is each frame is determined. The frame length is set to be 2048 points with a hop size of 512 points. Note that these frame parameters have been used consistently across all features discussed in this section. Finally, the average and standard deviation of the Zero-Crossing Rate across all frames are chosen as representative features.
8) *Tempo:* In general terms, tempo refers to how fast or slow a piece of music is; it is expressed in terms of Beats Per Minute (BPM). Different kinds of music genres would have different levels of tempos. Since the tempo of the audio piece can vary with time, we aggregate it by computing the mean across several frames. The functionality in Librosa first computes a tempogram following *(Grosche et al.,2010)* [10] and then estimates a single value for tempo.

### *C. Classifiers*

We tried executing various Classification models. The list included KNN, Logistic Regression and 5 Ensemble algorithms.

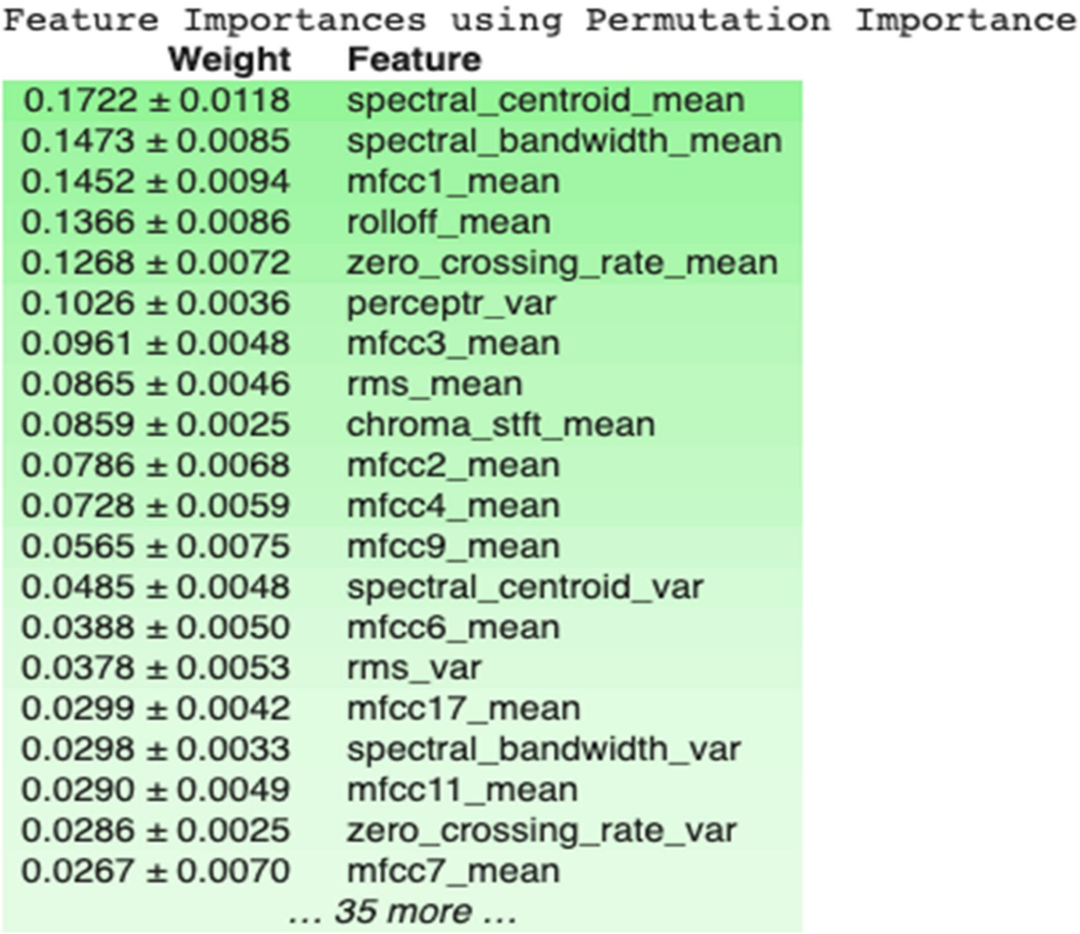
1) *Logistic Regression:* A supervised learning algorithm used for classification purposes. There are 3 main classifications in Logistic Regression: Binomial, Multinomial and Ordinal. We used a multinomial classifier in order to differentiate between the 10 genres categories.
2) *XGBoost Algorithm:* An ensemble learning algorithm. It stands for eXtreme Gradient Boosting which is known for its speed and performance. The Gradient Boosting is implemented on Decision Trees.
3) *AdaBoost Algorithm:* An ensemble learning algorithm. AdaBoost, which stands for Adaptive Boosting, is a statistical classification meta-algorithm formulated by *Yoav Freund and Robert Schapire. [4]*
4) *Gradient Boosting Method:* An ensemble learning algorithm. It combines results from multiple decision trees and generates a final decision. All weak learners are decision trees in gradient boosting.
5) *CatBoost Algorithm:* CatBoost stands for Categorical Boosting. The library works well on different Categories of data. Boosting since the algorithm is based on Gradient Boosting Algorithm.
6) *Random Forest Algorithm:* A supervised ensemble learning algorithm used for both Regression and Classification (Here classification). It makes use of multiple decision trees and then generates the result using voting.
7) *KNN:* A simple supervised ensemble learning algorithm that categorizes the data based on their similarities. It is used mainly for classification.

### *D. Hyperparameter Tuning*

Hyperparameter tuning refers to the optimizing of learning algorithms by setting particular parameters which leads the models to learn in a certain way. The learning method is controlled by initialized variables. XGBoost and Random Forest are two algorithms where run hyperparameters and the results were extensively good.

### *E. Features Selection*

There are in total 55 features extracted from model analysis. Out of the 55 features, we select the 30 most important features necessary for our analysis. We use the eli5 library along with logistic regression to select our top 30 features from permutation importance.



Feature Importances using Permutation Importance

| Weight | Feature |
|---|---|
| 0.1722 ± 0.0118 | spectral_centroid_mean |
| 0.1473 ± 0.0085 | spectral_bandwidth_mean |
| 0.1452 ± 0.0094 | mfcc1_mean |
| 0.1366 ± 0.0086 | rolloff_mean |
| 0.1268 ± 0.0072 | zero_crossing_rate_mean |
| 0.1026 ± 0.0036 | perceptr_var |
| 0.0961 ± 0.0048 | mfcc3_mean |
| 0.0865 ± 0.0046 | rms_mean |
| 0.0859 ± 0.0025 | chroma_stft_mean |
| 0.0786 ± 0.0068 | mfcc2_mean |
| 0.0728 ± 0.0059 | mfcc4_mean |
| 0.0565 ± 0.0075 | mfcc9_mean |
| 0.0485 ± 0.0048 | spectral_centroid_var |
| 0.0388 ± 0.0050 | mfcc6_mean |
| 0.0378 ± 0.0053 | rms_var |
| 0.0299 ± 0.0042 | mfcc17_mean |
| 0.0298 ± 0.0033 | spectral_bandwidth_var |
| 0.0290 ± 0.0049 | mfcc11_mean |
| 0.0286 ± 0.0025 | zero_crossing_rate_var |
| 0.0267 ± 0.0070 | mfcc7_mean |

*… 35 more …*

## V. EVALUATION

### *A. Metrics*

We used 4 main metrics for model analysis: F1 Score, Accuracy, Precision and Recall. Each model also gave a different set of analysis based on how they performed for a particular genre. The final accuracy noted was a mean of all.

| Parameter→<br>Algorithm↓ | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| Logistic Regression | 0.675 | 0.671 | 0.675 | 0.672 |
| Random Forest | 0.828 | 0.829 | 0.828 | 0.827 |
| AdaBoost | 0.456 | 0.433 | 0.455 | 0.421 |
| GBM | 0.762 | 0.764 | 0.762 | 0.762 |
| XGBoost | 0.879 | 0.879 | 0.879 | 0.878 |
| CatBoost | 0.863 | 0.864 | 0.863 | 0.863 |
| KNN | 0.867 | 0.870 | 0.867 | 0.867 |

### *B. Confusion Matrices and Model Analysis*

A confusion matrix is a tabular representation of model analysis of a classification model of a set of data for which true values are given.

#### *1) Logistic Regression*

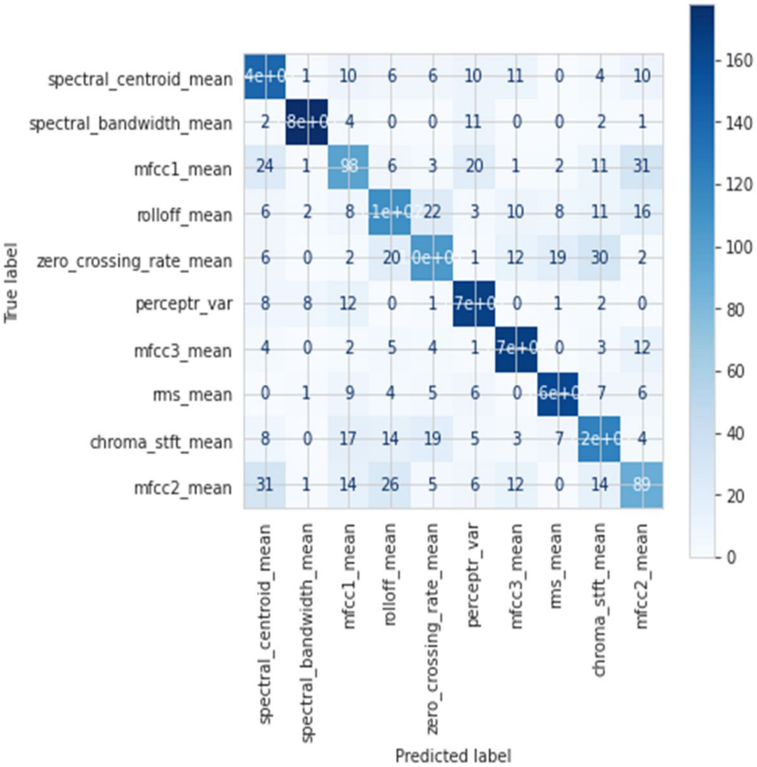
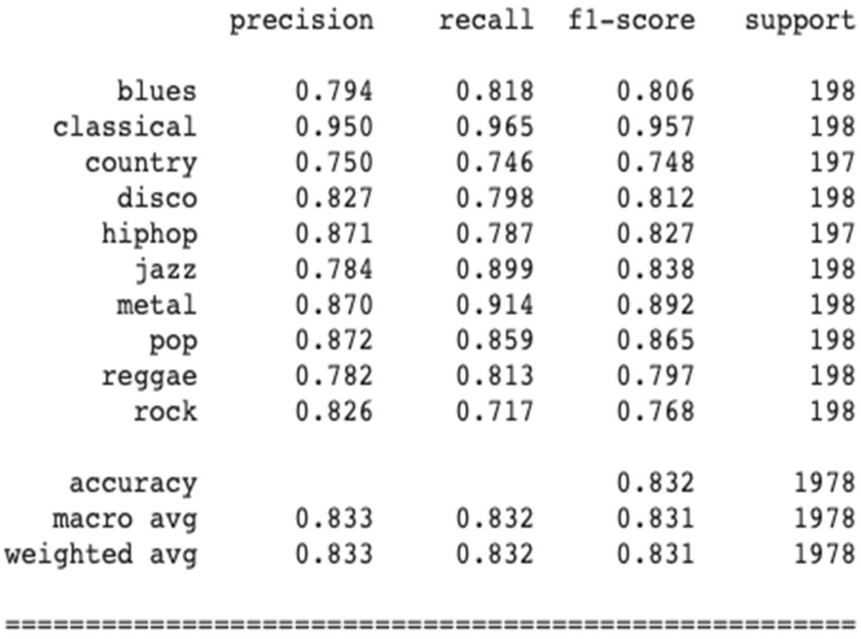


Figure: 5.2.1



```
              precision    recall  f1-score   support

       blues      0.794     0.818     0.806       198
   classical      0.950     0.965     0.957       198
     country      0.750     0.746     0.748       197
       disco      0.827     0.798     0.812       198
      hiphop      0.871     0.787     0.827       197
        jazz      0.784     0.899     0.838       198
       metal      0.870     0.914     0.892       198
         pop      0.872     0.859     0.865       198
      reggae      0.782     0.813     0.797       198
        rock      0.826     0.717     0.768       198

    accuracy                          0.832      1978
   macro avg      0.833     0.832     0.831      1978
weighted avg      0.833     0.832     0.831      1978

=====================================================
```

*2) XGBoost Algorithm*

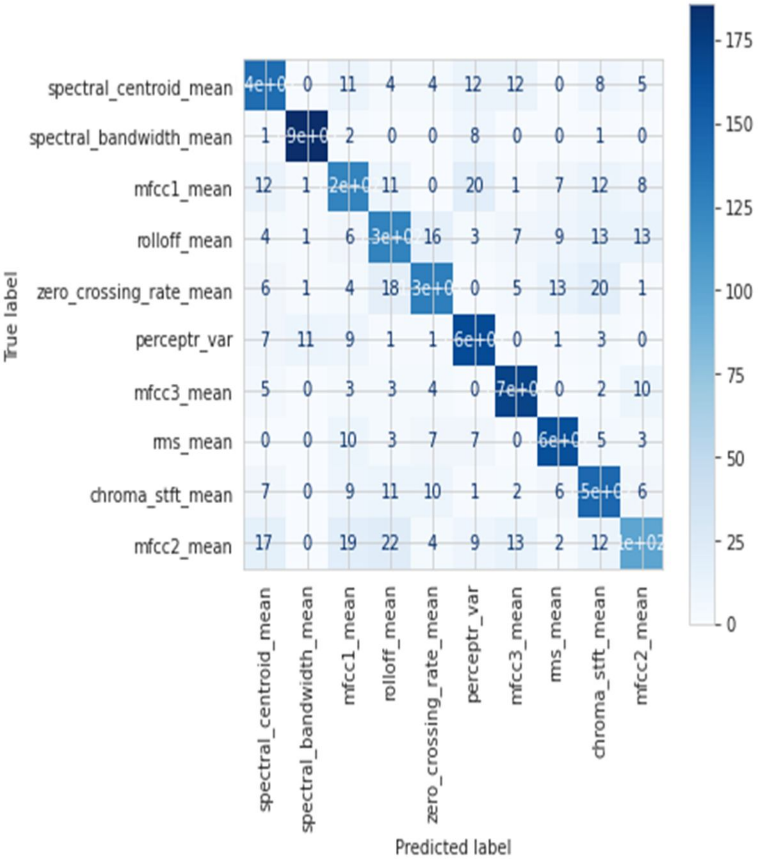
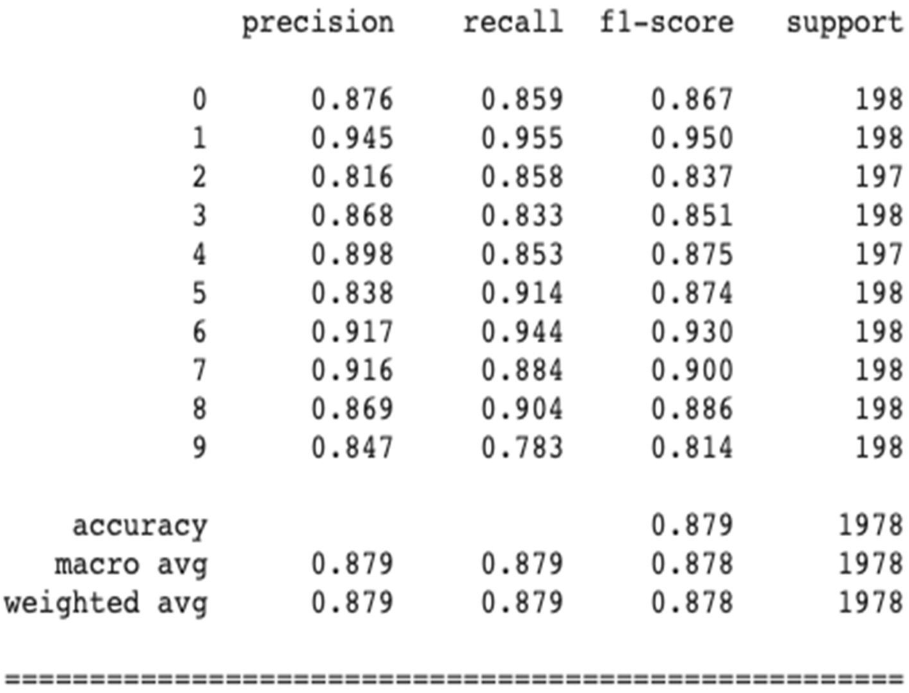


Figure: 5.2.2



```
              precision    recall  f1-score   support

           0      0.876     0.859     0.867       198
           1      0.945     0.955     0.950       198
           2      0.816     0.858     0.837       197
           3      0.868     0.833     0.851       198
           4      0.898     0.853     0.875       197
           5      0.838     0.914     0.874       198
           6      0.917     0.944     0.930       198
           7      0.916     0.884     0.900       198
           8      0.869     0.904     0.886       198
           9      0.847     0.783     0.814       198

    accuracy                          0.879      1978
   macro avg      0.879     0.879     0.878      1978
weighted avg      0.879     0.879     0.878      1978

=====================================================
```

*3) AdaBoost Algorithm*

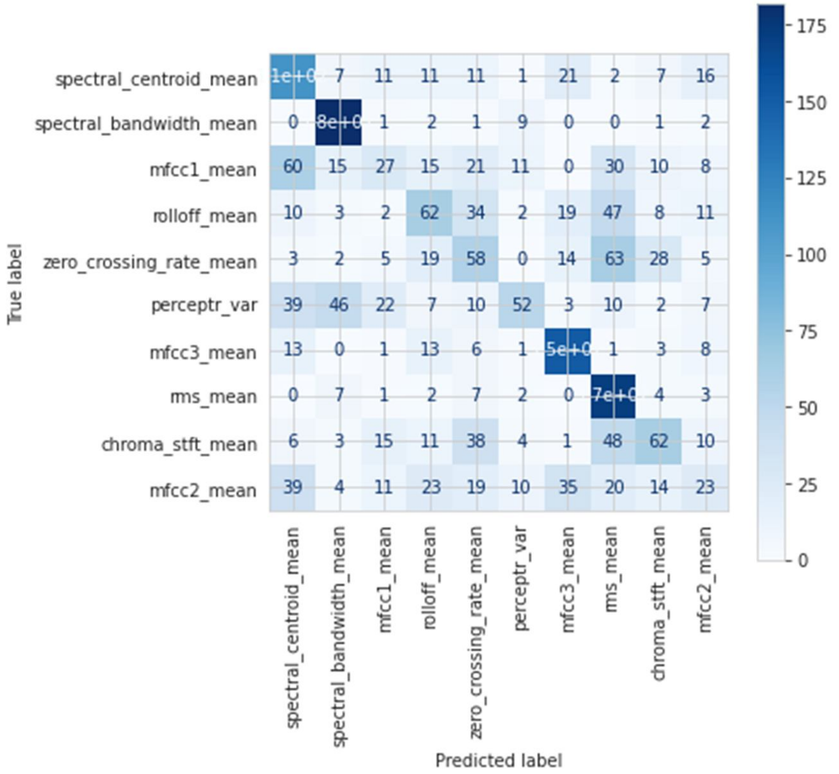
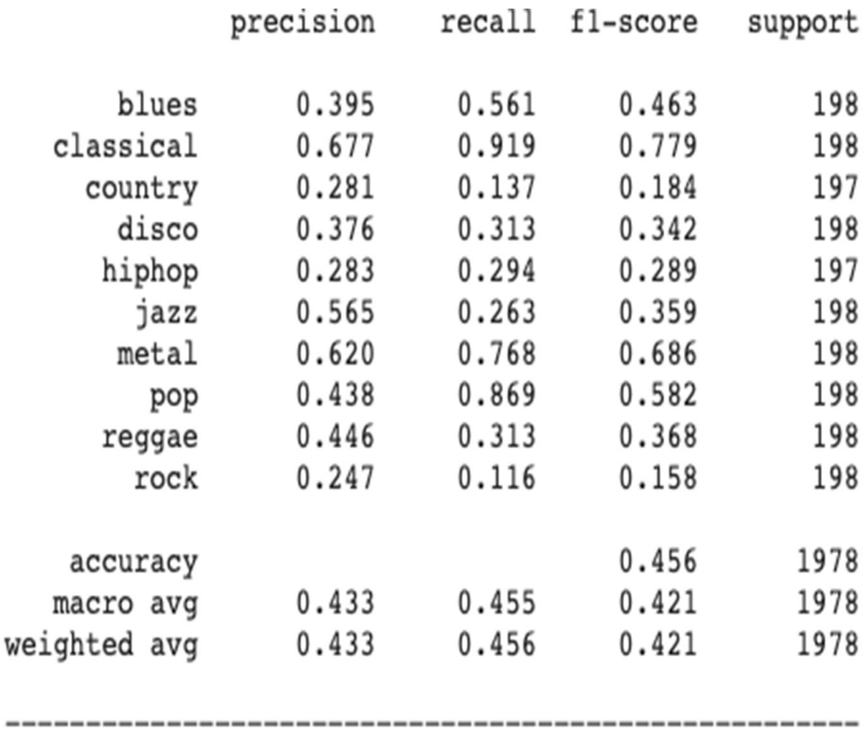


Figure: 5.2.3



```
              precision    recall  f1-score   support

       blues      0.395     0.561     0.463       198
   classical      0.677     0.919     0.779       198
     country      0.281     0.137     0.184       197
       disco      0.376     0.313     0.342       198
      hiphop      0.283     0.294     0.289       197
        jazz      0.565     0.263     0.359       198
       metal      0.620     0.768     0.686       198
         pop      0.438     0.869     0.582       198
      reggae      0.446     0.313     0.368       198
        rock      0.247     0.116     0.158       198

    accuracy                          0.456      1978
   macro avg      0.433     0.455     0.421      1978
weighted avg      0.433     0.456     0.421      1978

-----------------------------------------------------
```

*4) Gradient Boosting Method*

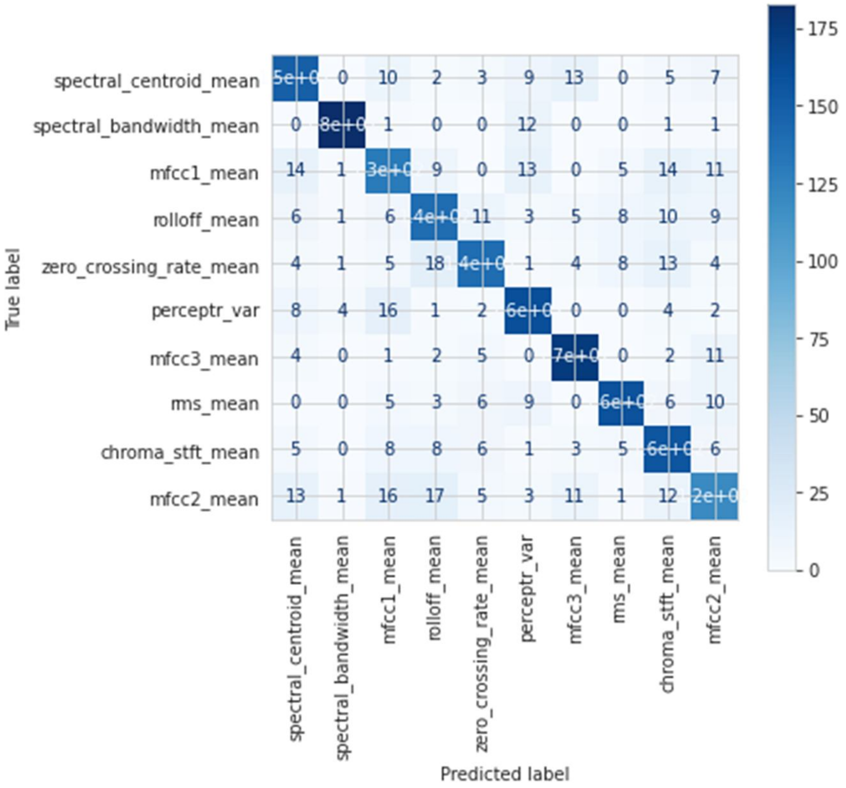
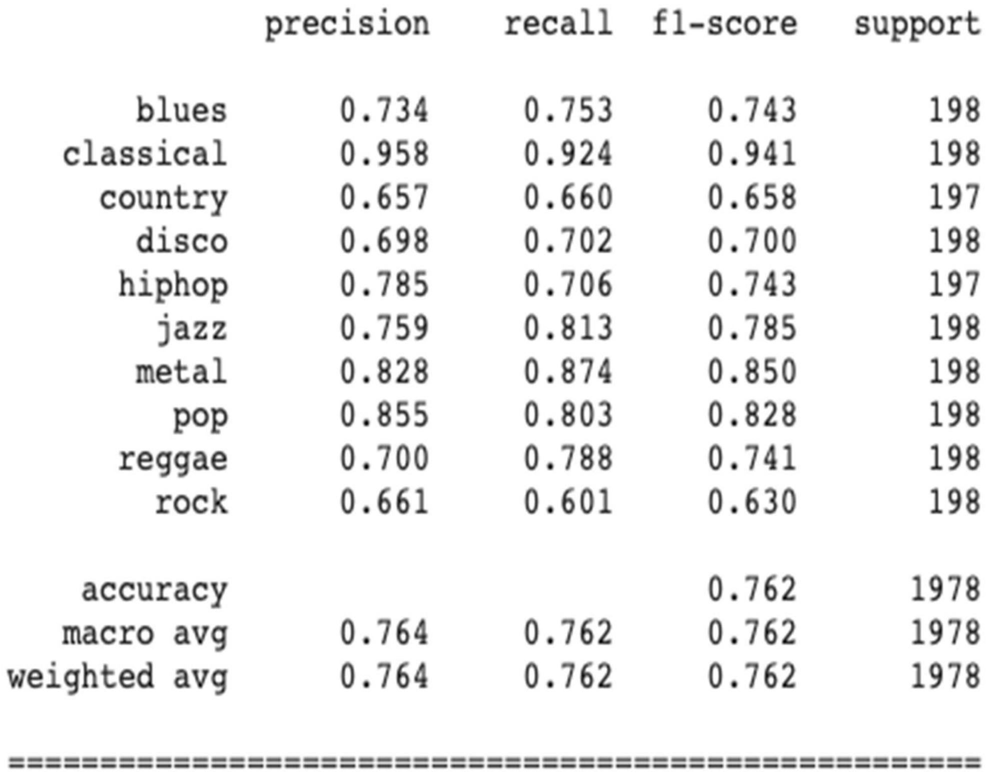


Figure: 5.2.4



```
              precision    recall  f1-score   support

       blues      0.734     0.753     0.743       198
   classical      0.958     0.924     0.941       198
     country      0.657     0.660     0.658       197
       disco      0.698     0.702     0.700       198
      hiphop      0.785     0.706     0.743       197
        jazz      0.759     0.813     0.785       198
       metal      0.828     0.874     0.850       198
         pop      0.855     0.803     0.828       198
      reggae      0.700     0.788     0.741       198
        rock      0.661     0.601     0.630       198

    accuracy                          0.762      1978
   macro avg      0.764     0.762     0.762      1978
weighted avg      0.764     0.762     0.762      1978

=====================================================
```

*5) CatBoost Algorithm*

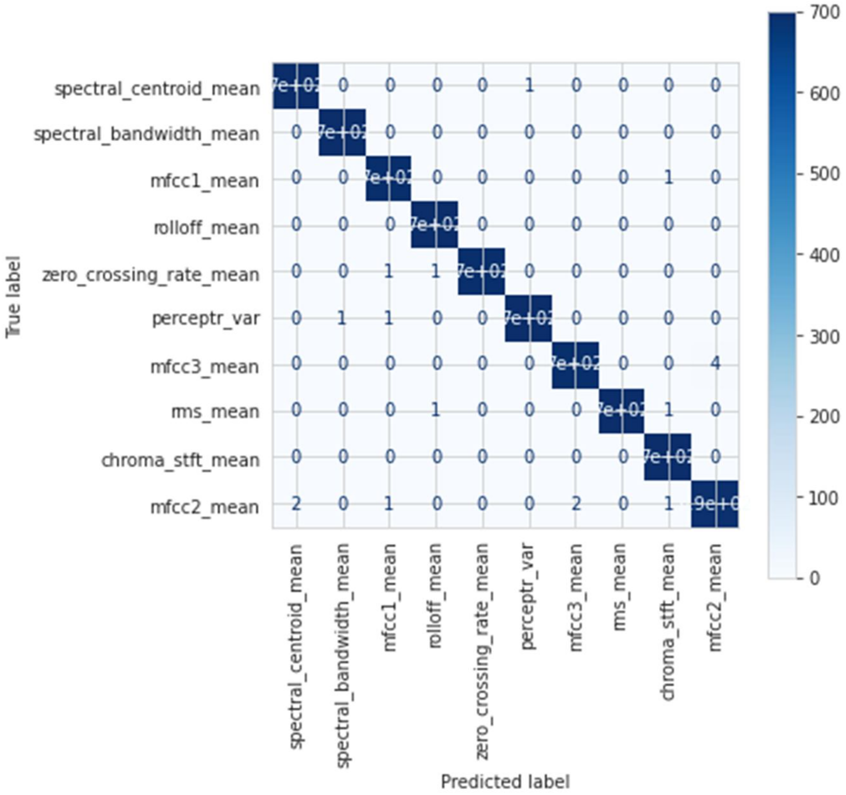
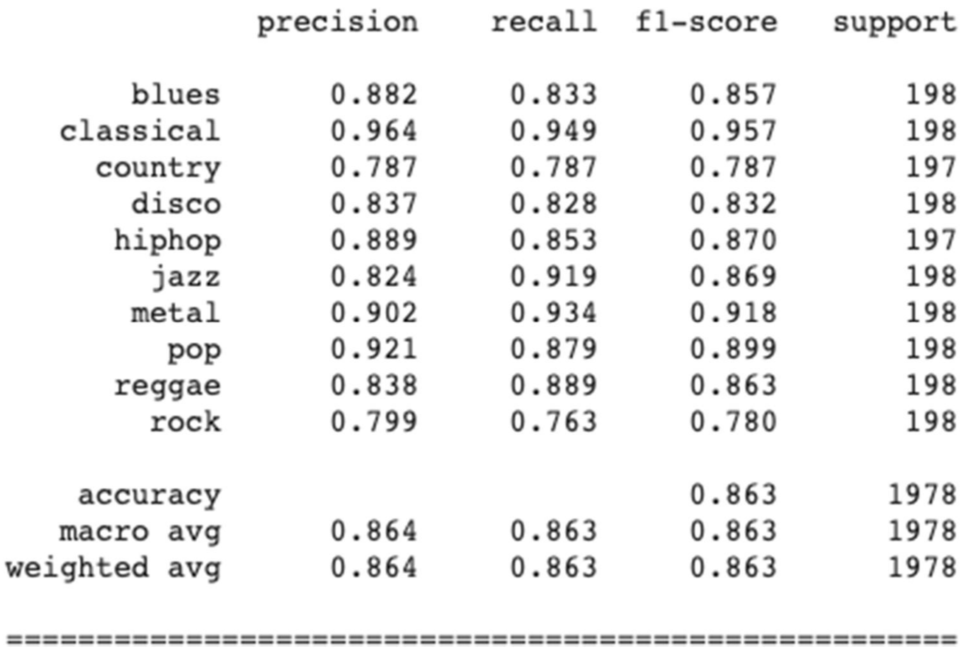


Figure: 5.2.5



```
              precision    recall  f1-score   support

       blues      0.882     0.833     0.857       198
   classical      0.964     0.949     0.957       198
     country      0.787     0.787     0.787       197
       disco      0.837     0.828     0.832       198
      hiphop      0.889     0.853     0.870       197
        jazz      0.824     0.919     0.869       198
       metal      0.902     0.934     0.918       198
         pop      0.921     0.879     0.899       198
      reggae      0.838     0.889     0.863       198
        rock      0.799     0.763     0.780       198

    accuracy                          0.863      1978
   macro avg      0.864     0.863     0.863      1978
weighted avg      0.864     0.863     0.863      1978

=====================================================
```

*6) Random Forest Algorithm*

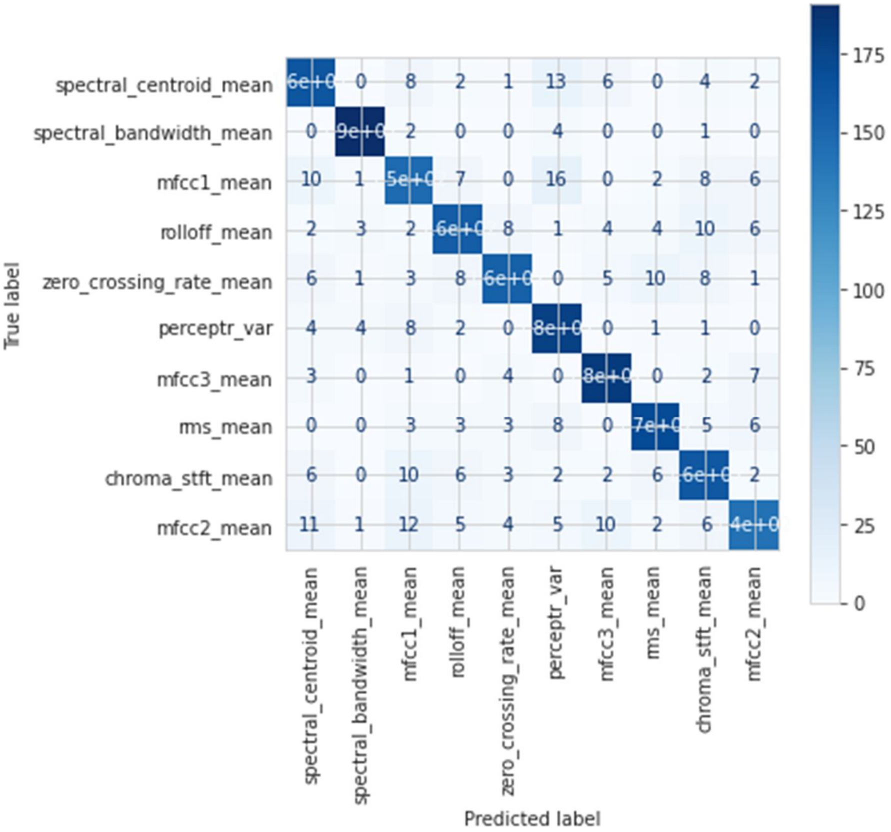
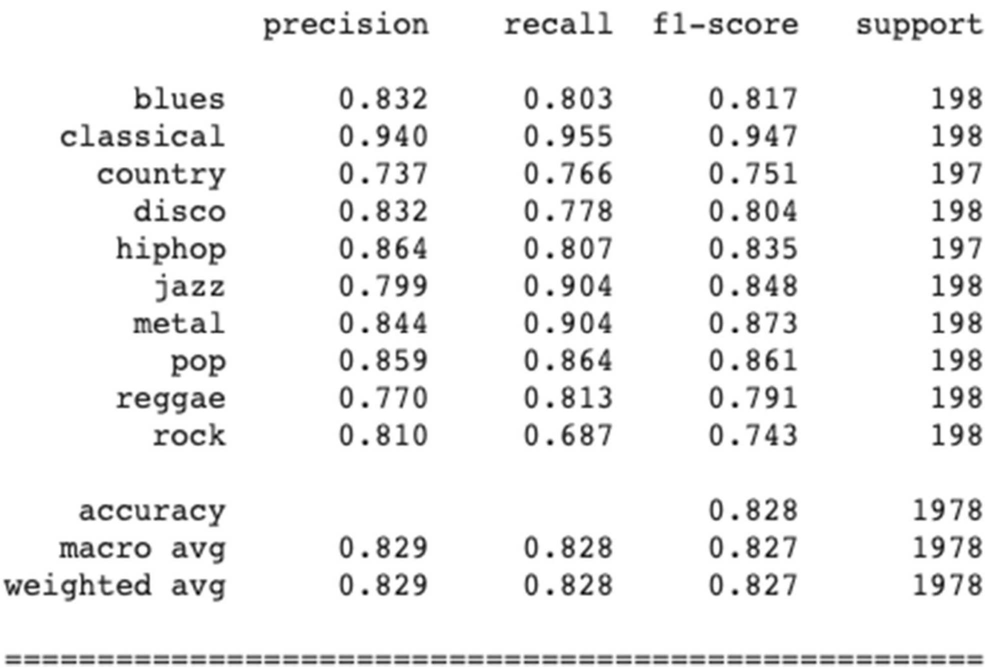
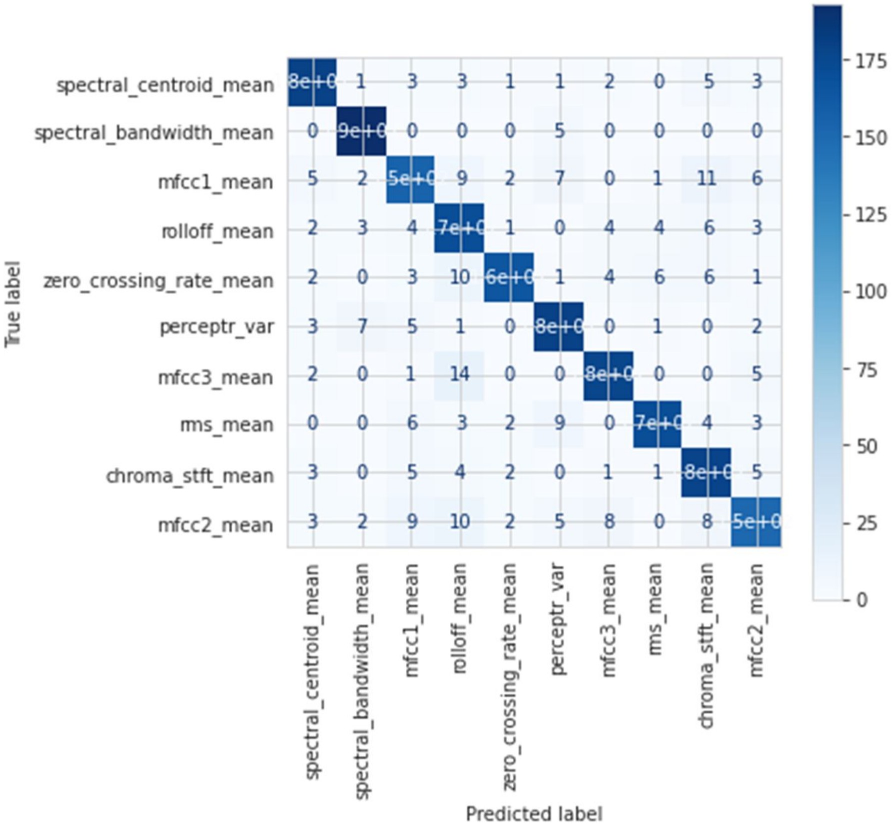


Figure: 5.2.6



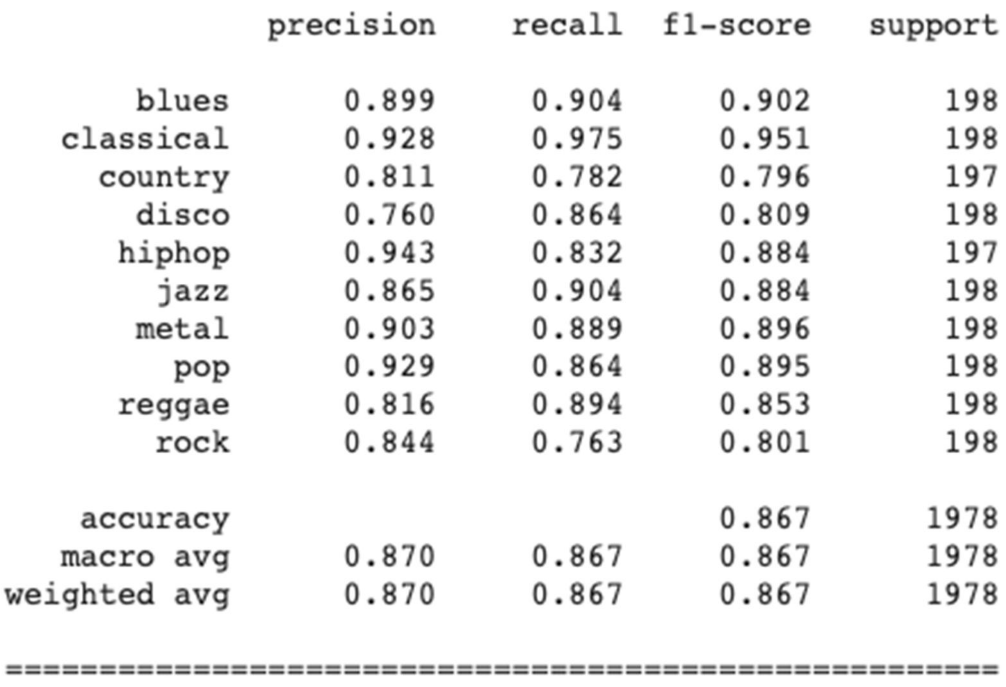
```
              precision    recall  f1-score   support

       blues      0.832     0.803     0.817       198
   classical      0.940     0.955     0.947       198
     country      0.737     0.766     0.751       197
       disco      0.832     0.778     0.804       198
      hiphop      0.864     0.807     0.835       197
        jazz      0.799     0.904     0.848       198
       metal      0.844     0.904     0.873       198
         pop      0.859     0.864     0.861       198
      reggae      0.770     0.813     0.791       198
        rock      0.810     0.687     0.743       198

    accuracy                          0.828      1978
   macro avg      0.829     0.828     0.827      1978
weighted avg      0.829     0.828     0.827      1978

=====================================================
```

*7) KNN*



| | spectral_centroid_mean | spectral_bandwidth_mean | mfcc1_mean | rolloff_mean | zero_crossing_rate_mean | perceptr_var | mfcc3_mean | rms_mean | chroma_stft_mean | mfcc2_mean |
|---|---|---|---|---|---|---|---|---|---|---|
| spectral_centroid_mean | 8e+0 | 1 | 3 | 3 | 1 | 1 | 2 | 0 | 5 | 3 |
| spectral_bandwidth_mean | 0 | 9e+0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 |
| mfcc1_mean | 5 | 2 | 5e+0 | 9 | 2 | 7 | 0 | 1 | 11 | 6 |
| rolloff_mean | 2 | 3 | 4 | 7e+0 | 1 | 0 | 4 | 4 | 6 | 3 |
| zero_crossing_rate_mean | 2 | 0 | 3 | 10 | 6e+0 | 1 | 4 | 6 | 6 | 1 |
| perceptr_var | 3 | 7 | 5 | 1 | 0 | 8e+0 | 0 | 1 | 0 | 2 |
| mfcc3_mean | 2 | 0 | 1 | 14 | 0 | 0 | 8e+0 | 0 | 0 | 5 |
| rms_mean | 0 | 0 | 6 | 3 | 2 | 9 | 0 | 7e+0 | 4 | 3 |
| chroma_stft_mean | 3 | 0 | 5 | 4 | 2 | 0 | 1 | 1 | 8e+0 | 5 |
| mfcc2_mean | 3 | 2 | 9 | 10 | 2 | 5 | 8 | 0 | 8 | 5e+0 |

Figure: 5.2.7



```
              precision    recall  f1-score   support

       blues      0.899     0.904     0.902       198
   classical      0.928     0.975     0.951       198
     country      0.811     0.782     0.796       197
       disco      0.760     0.864     0.809       198
      hiphop      0.943     0.832     0.884       197
        jazz      0.865     0.904     0.884       198
       metal      0.903     0.889     0.896       198
         pop      0.929     0.864     0.895       198
      reggae      0.816     0.894     0.853       198
        rock      0.844     0.763     0.801       198

    accuracy                          0.867      1978
   macro avg      0.870     0.867     0.867      1978
weighted avg      0.870     0.867     0.867      1978

=====================================================
```

### *C. Output*

The program checks the accuracy of different machine learning models. The trained models are then tested on the input musical data. The user inputs .wav music files which on white space removal converts the audio into a NumPy array format. The data is split into 10 three seconds data files that run on the trained models. The most recurred output is then selected as the prediction.

## VI. RESULT

The classifier on inputting audio takes in the first 30 seconds and extracts the audio features and tries to detect the genre. The accuracy of detection depends on the model applied, each model given varied efficiency, from 45.6% (average) on AdaBoost and 87.9% (average) on XGBoost. The model also saw variation in accuracies for independent genres. The maximum average accuracy of a genre prediction is classical music and the minimum being country in most cases.

## VII. CONCLUSION

The classifiers that produce the best accuracies are KNN and XGBoost. The 3 seconds split audio data set is much more efficient at giving the final results. Hyperparameter tuning worked very effectively in the case of XGBoost and Random Forest. Classical music gave the best prediction accuracy while the worst genre varied model to model (maximum being Classical. We concluded that the input audio files with noise removal and preprocessing gave much better results. There are other methods that could be followed as an approach to classify genre.

## REFERENCES

[1] George Tzanetakis and Perry Cook. 2002. Musical genre classification of audio signals. IEEE Transactions on speech and audio processing 10(5):293-302

[2] Jan W¨ulfing and Martin Riedmiller. Unsupervised learning of local features for music classification. In ISMIR, pages 139–144, 2012.

[3] Brian McFee , Colin Raffel , Dawen Liang , Daniel P.W. Ellis , Matt McVicar, Eric Battenbergk , Oriol Nieto. librosa: Audio and Music Signal Analysis in Python.

[4] Yoav Freund and Robert Schapire. A Short Introduction to Boosting.

[5] Steven B Davis and Paul Mermelstein. 1990. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. In Readings in speech recognition, Elsevier, pages 65-74.

[6] Dan Ellis. 2007. Chroma features analysis and synthesis. Resources of Laboratory for the Recognition and Organization of Speech and Audio-LabROSA

[7] Steve Tjoa. 2017. Music information retrieval. https://musicinformationretrieval.com/spectral_features.html

[8] Fabien Gouyon, François Pachet, Olivier Delerue, et al. 2000. On the use of zero-crossing rate for an application of classification of percussive sounds. In Proceedings of the COST G-6 conference on Digital Audio Effects (DAFX-00), Verona, Italy

[9] Peter Grosche, Meinard M¨uller, and Frank Kurth. 2010. Cyclic tempograma mid-level tempo representation for music signals. In Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on. IEEE, pages 5522–5525.