Music Genre Classification

Name:	**Marut Priyadarshi**
Registration No./Roll No.:	19191
Institute/University Name:	IISER Bhopal
Program/Stream:	DSE
Problem Release date:	February 02, 2022
Date of Submission:	April 23, 2022

1 Introduction

With the rapid expansion of the music streaming industry, music genre classificaton has become quite an integral part of the algorithm that serves music to customers as different genres of music are generally associated with different moods and emotions and thus a better classificaton technique will increase the accuracy with which an algorithm recommends a song to a user that fits their current state of mind, and as a consequence increases user retention

The current dataset consists of 59 different features and four thousand five hundred datapoints. I have shown the correlation heatmap below for visualization purposes (Figure 1)

After the initial look at the data, the plan of action is to try out some feature selection methods, proceeded by different data normalization techniques. then test those different methods independently with different classifiers to see which combination gives us the best results. I plan to use SVM, Random Forest, K Nearest Neighbors, Gaussian Naive Bayes and Decision Tree based Bagging Classifier.

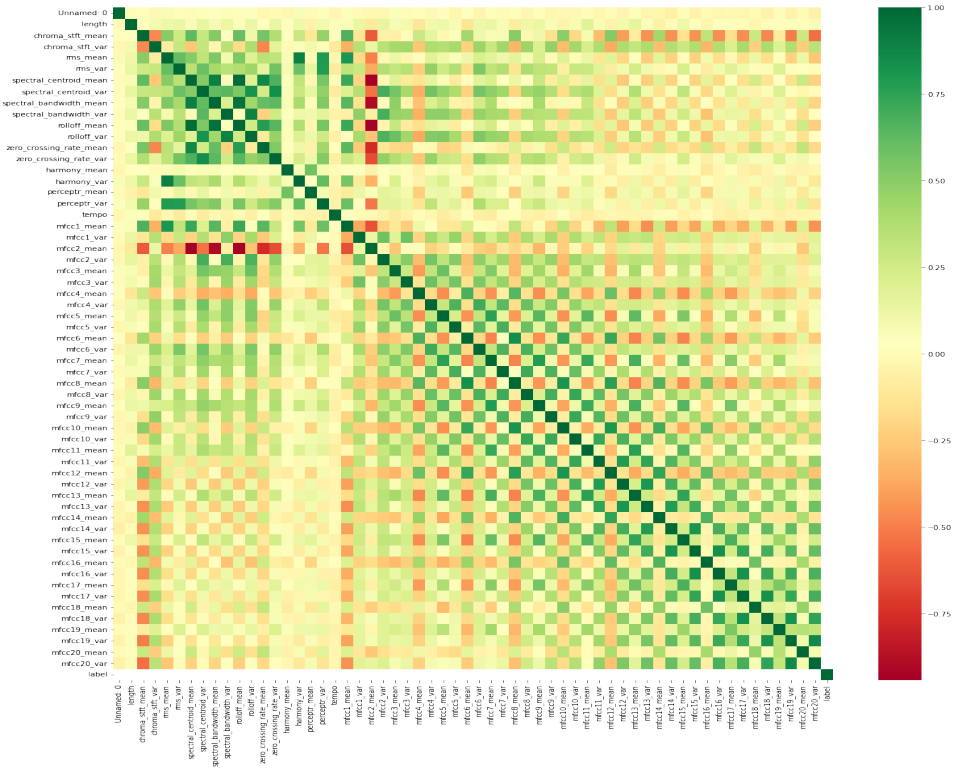


Figure 1: Correlation Heatmap of the Features

2 Methods

We have divided our tasks into three parts:
• Data Preprocessing
• Feature Selection
• Training the classifiers

2.1 Data Preprocessing

Looking at the data it can be seen that there for some features there is a lot of variance while for others, the values are situated more close together. While it is true that values with high variance may store more information, there is also the downside that these features cause the classifier being trained to be heavily biased towards them regardless of how important those features actually are and thus dilute the effect of features with lower variance. This often results from a difference in scale of the features as features with a lower scale may be considered less important by the classifiers because the numerical values do not differ by the same amount as the features with a larger scale. This effect is especially pronounced in classifiers that work on the distance between points, like SVM or K-Nearest Neighbors.

To overcome this we are going to use scalars which will transform the data and bring all the attributes to the same scale, which reduces the bias soleley due to the scale of the attributes. For our work we have used Standard Scalar, MinMax Scalar and the more outlier resistant Robust Scalar.

Now, our labels for our training data are categorical and the labels that we will generate for the test data will also need to be in categorical form. But for working with the labels, numerical data is prefere. So we use LabelEncoder to encode the categorical labels into numeric labels and convert the final generated numeric labels back to the desired categorical data.

2.2 Feature Selection

Now that our data is on the same scale, we can move to feature selection. In our dataset, we have 59 features. While it can be possible that all of them are important, we have to test and see if the results are improved if we select the most important of the features, as suggested by different algorithm. While training the data we will use the trimmed and the original data with all features and see which of them give better results.

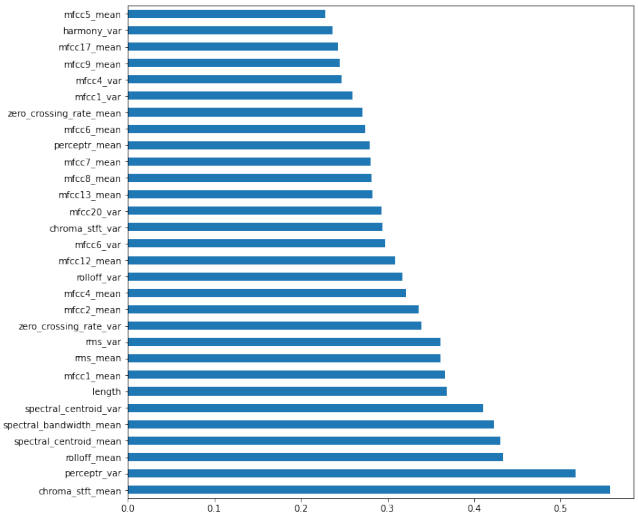
To select the best features we used Feature Importance (part of ExtraTreesClassifier) to give us the best thirty features out of 59. It is an impurity based feature importance, the higher the score, the more important the feature. The importance of a feature is computed as the (normalized) total reduction of the criterion brought by that feature. It is also known as the Gini importance. It is shown in Figure 2b

Then we used mutual info classif to do the same. Mutual information between two random variables is a non-negative value, which measures the dependency between the variables. It is equal to zero if and only if two random variables are independent, and higher values mean higher dependency. It is shown in Figure 2a

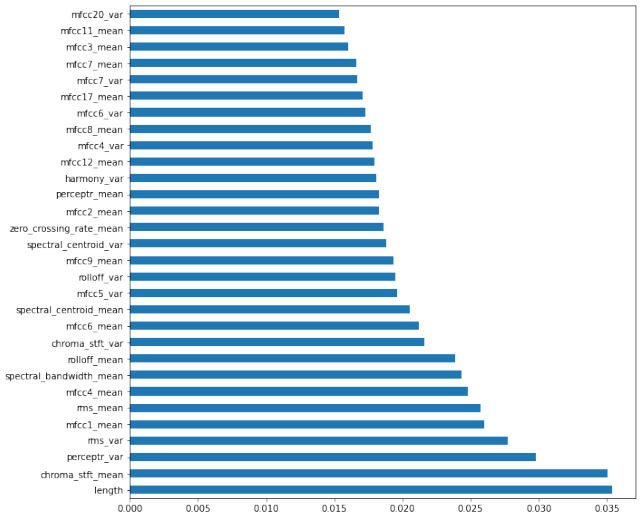
Then finally we made a correlation matrix of all the features and then we searched for the features that were corrlelated by a factor of more than 75%. Then, we selected those features that were present both in the best features listed in the previous two methods. Then we removed the feature from the highly correllated pairs that were not among the best features given by those methods. At the end of the trimming, we ended up with 32 features, as a result of feature selection.

2.3 Training the classifiers

We are going to try out different combinations of techniques and scalars with both original dataset with all 59 features and the trimmed dataset with the 32 featured selected in the previous step.For our techniques, we are going with Random Forest, Support vector classifier, K nearest neighbors, Gaussian Naive Bayes and Bagging classifier with Decision tree base.



(a) Best 30 Features based on mutual_info_classif



(b) Best 30 features based on Feature Importance

Figure 2: The two techniques

For hyperparameter tuning we are using GridSearchCV to try out different parameters and pick the one that gives the best score when using k-fold cross validation. It trains the classifiers with all the specified values for different parameters and gives us the best perfocming values for those parameters

The code for this is uploaded to GitHub at this Link.

3 Evaluation Criteria

Precision is the ratio of the number of true positives to the total number of true positives and false positives, ie:

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

Recall on the other hand is the ratio of the number of True positives to the total number of true positives and false negatives, ie:

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives}$$

F-Measure combines both of these as it is the Harmonic mean of precision and recall, ie:

$$F - Measure = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

F-measure can have a value ranging from 0 to 1, with 1 being perfect accuracy and 0 being the worst possible score. f-macro average is the unweighted average of the f-measure scores for each class. Since it is unweighted, it is useful in the cases when a certain class has fewer members. For most of our decisions we will be looking at f-measure and f-macro averages to determine the performance of our classifiers

4 Analysis of Results

We finally get the results of the different classifiers with and without feature scaling performing on different data normalized by different scalars. Note that we have not used Random Forest and Decision Tree based Bagging Classifier on differently scaled data as Decision Trees and Ensemble Classifiers do not require feature scaling as they are not sensitive to the variance in data. We can see the scores in Table 1

Table 1: Maximum F1 Macro Scores of Different Techniques

Model Name	Scaler		
	Standard	Robust Scaler	MinMax Scalar
Random Forest	75.9367%	75.9367%	-
Support Vector Classifier	72.1506%	76.2189%	**76.5616%**
Bagging Classifier	-	74.2561%	-
Gaussian NB	42.6214%	-	-
K-Nearest Neighbor	63.8760%	-	-

In Table 2, we see how feature selection affected the performance of the different classifiers used. In most cases, the perfomace is increased by a significant margin after performing feature selection. Gaussian NB is the exception here, but since it is relatively immune to overfitting, more data and more features only improves its performance. In the rest of the classifiers, feature selection has a positive impact on their performance.

Table 2: Performance comparison of classifiers with and without Feature Selection

Model Name	Features	
	All Features	32 Features
Random Forest	75.0700%	**75.9367%**
Support Vector Classifier	72.2222%	**76.5616%**
Bagging Classifier	72.4449%	**74.2561%**
Gaussian NB	**49.1326%**	42.6214%
K-Nearest Neighbor	51.2821%	**63.8760%**

Thus, by looking at the performances, the best performing classifier is Support Vector Machine with an rbf kernel, and thus we will be using it to generate the final labels for the given test data.

5 Discussions and Conclusion

Support Vector Machine is so far the best performing classifier on our dataset. With 59, different features, feature selection had a definite impact. Still, the accuracy of the predictions are still around 75%, which are good, but nowhere as perfect as desired. This goes to show how even with all this data, music genre classification is a pretty complex task. So there are a lot of improvements that can be done to it, better feature selection, better hyperparamter tuning and possibly better classification techniques that can improve upon our current results.

As it currently is, it can be used to fairly reliably predict the genre of an audio file, given the different data. This code can only work if all the data is provided in a spcecific format, so another future scope for this project is a version that is capable of extracting all the relevant information from the audio file itself, making it much more versatile.