**MINOR FINAL EVALUATION REPORT**



**MUSIC GENRE CLASSIFICATION**

**Evaluated By : Dr. Neetu Sardana & Dr. Shikha Mehta**

**Supervised By : Dr. Chetna Dabas**

**Submitted By:**

Naman Gupta 15103095

Vaibhav Jain 15103113  
 Aditya Aggarwal 15103085

Siddhant Pathak 15103108

**INTRODUCTION**

Music is categorized into subjective categories called genres. With the growth of the Internet and multimedia systems applications that deal with the musical databases gained importance and demand for Music Information Retrieval (MIR) applications increased. Musical genres have no strict definitions and boundaries as they arise through a complex interaction between marketing, historical, and cultural factors.

Our project is a research – based analysis of the music genres using spectral and rhythmic features of the music We have done a comparative study using various machine learning algorithms to classify the music into its various genres namely, blues, classical, country, disco, hip-hop, jazz, metal and pop respectively. We have used various audio features, such as Mel the public, Frequency Cepstral Coefficients (MFCC), Delta, Delta- Delta and temporal features, including beats and tempo to featurize our data. Various classification algorithms, such as Support Vector Machine(SVM), Decision Tree, k-Nearest Neighbors(KNN), Random Forest and Gradient Boosting Classifier are used in the classification of the data.

We believe that there is no prior work done to analyse the algorithms using features such as Delta, Delta-Delta and tempo.

**PROBLEM STATEMENT**

Today’s generation has become really tech savy . They want to blend technology with their moods . So , to faciliate the user to listen to the genre of music which matches their mood , we decided to create a project which classifies the music into different genres .

**TECHNOLOGIES & TOOLS USED**

* **Python**
* Librosa module
* Anaconda
* Glob module
* Pandas
* Numpy
* Scikit
* scipy.io for wav file
* **Flask**
* secure\_file (werkzeug)
* request
* **Android**
* Permissions (external storage permission and internet permission)
* Uploading music file using HTTP

**METHODOLOGY**

The architectural structure of our dataset is pre-processed. The pre-processed data is used for the training of each of our classifiers. The classified data is then tested using the test data.

**ALGORITHM**

1. Convert data from .au format to .wav format.
2. Preprocess the data using FFT.
3. Feature Extraction using MFCC, Delta, Delta- Delta and rhythmical features.
4. Feature Reduction using Principal Component Analysis.
5. Optimisation of hyper-parameters using Grid Search with Cross-Validation.
6. Training the classifier using various classification algorithms.
7. Testing the data and predicting the genre of our data files.
8. Compare performances of different classifiers using different benchmark.

**PROJECT DESCRIPTION**

**MUSIC DATASET:**

The dataset we have used for our music genre classification is GTZAN[]. This dataset contains 1000 song files, each of which is 30 seconds long. These songs are classified into 8 genres, namely, blues, classical, country, disco, hip-hop, jazz, metal and pop respectively. The sampling rate we have used for our data files is 22050 Hz. All these files were in .au format which were converted to .wav using online converter. This is done since .wav format files are easily read by python modules. We divided our dataset into training and testing data in the ratio 7.5 : 2.5. Spectogram of songs of different genres are depicted.

**PREPROCESSING:**

**a. Reading Files From System :**

GLOB : The [glob](https://docs.python.org/3/library/glob.html#module-glob) module finds all the pathnames matching a specified pattern according to the rules used by the Unix shell, although results are returned in arbitrary order. We have used this module to read the wav files of our dataset.

glob.**glob**(*pathname*, *\**, *recursive=False*)

**b. Audio File Preprocessing :**

The audio was converted to mono from stereo.

**c.Fast Fourier Transform:**

A FFT is a mathematical method to obtain DFT(Discrete Fourier Transform) for a sequence or the inverse of a sequence. We have performed Fourier Analysis to obtain a frequency domain representation of the original domain. Rapid computation of this transform by the factorization of Discrete Fourier Transform Matrix into a sparse factors' product is job done by an FFT. Because of which, the complexity of obtaining a DFT is reduced to O(n log n) from O(n^2) to , where n represents the data size. Let x0, ...., xN-1 denote complex numbers. The DFT is obtained by the formula Xk = ∑n to n=1 xe-i2πkn/N, where k = 0,1,….., N-1.

**FEATURES USED:**

1. **Short Time Fourier Transform(STFT):**

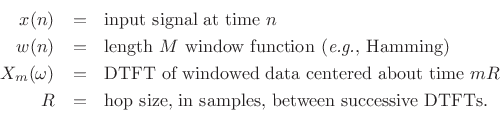
The Short-Time Fourier Transform (STFT) (or short-term Fourier transform) is a powerful general-purpose tool for audio [signal](http://www.dsprelated.com/dspbooks/filters/Definition_Signal.html) processing . It defines a particularly useful class of time-frequency distributions  which specify complex amplitude versus time and frequency for any signal. We are primarily concerned here with tuning the STFT parameters for the following applications:

1. Approximating the time-frequency analysis performed by the ear for purposes of spectral display.
2. Measuring model parameters in a short-time [spectrum](http://www.dsprelated.com/dspbooks/mdft/Example_Applications_DFT.html).

The usual mathematical definition of the [STFT](http://www.dsprelated.com/dspbooks/sasp/Short_Time_Fourier_Transform.html) is

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  | (8.1) |

where



1. **Mel Frequency Cepstral Coefficient :**

Mel Frequency Cepstral coefficients emphasize on obtaining the exact structure of the audio signal to extract linguistic features and discard the background noise. The linear cosine transform of a logarithmic power spectrum on a Mel scale which is non-linear is the basis of its calculation. A collection of Mel Frequency Cepstral Coefficients form a Mel frequency cepstrum. The Mel scale is calculated as M(f) = 1125 ln(1+ f/700).

The following computations are made to obtain the Discrete Fourier transform of the frame - Si(k) = ∑n to n=1 s(n)h(n)e-j2πkn/N where 1≤ k ≤ K h(n) – is the Analysis window for N samples K- Length of the Discrete Fourier Transform The power spectral estimate based on the Periodogram for si(n) , which is the speech frame is specified by Pi(k) = |Si(k)|2 This periodogram is further processed to obtain 26 cepstral co-efficients, DCT is applied on 26 log filter bank energies which is called the MFCC.

**Steps Involved in calculation of MFCC:**

* 1. Frame the signal into short frames.
  2. For each frame calculate the [periodogram estimate](http://en.wikipedia.org/wiki/Periodogram) of the power spectrum.
  3. Apply the melfilterbank to the power spectra, sum the energy in each filter.
  4. Take the logarithm of all filterbank energies.
  5. Take the DCT of the log filterbank energies.
  6. Keep DCT coefficients 2-13, discard the rest.

1. **Delta :**

The extraction of the cepstrum via the Inverse DFT from the previous section results in 12 cepstral coefficients for each frame. We next add a fourteenth feature: the energy from the frame. Energy correlates with phone identity and so is a useful cue for phone energy detection (vowels and sibilants have more energy than stops, etc). The energy in a frame is the sum over time of the power of the samples in the frame; thus for a signal x in a window from time sample t1 to time sample t2, the energy is:

Energy = ∑t=t1 to t2 x2[t]

1. **Delta- Delta :**

Another important fact about the speech signal is that it is not constant from frame to frame. This change, such as the slope of a formant at its transitions, or the nature of the change from a stop closure to stop burst, can provide a useful cue for phone identity. For this reason we also add features related to the change in cepstral features over time. We do this by adding for each of the 13 features a delta or velocity feature, and a double delta or acceleration feature. Each of the 13 delta features represents the change between frames in the corresponding cepstral/energy feature, while each of the 13 double delta features represents the change between frames in the corresponding delta features. A simple way to compute deltas would be just to compute the difference between frames; thus the delta value d(t) for a particular cepstral value c(t) at time t can be estimated as:

d(t) = [c(t+1) – c(t-1)] / 2

Instead of this simple estimate, however, it is more common to make more sophisticated estimates of the slope, using a wider context of frame.

1. **Tempo(Rhythmical Features) :**

In musical terminology, **tempo** is the speed or pace of a given piece. In classical music, tempo is usually indicated with an instruction at the start of a piece (often using conventional Italian terms). Tempo is usually measured in beats per minute (BPM).

To calculate tempo, onset strength envelope is calculated and then tempo is calculate by taking autocorrelation of various onset instants.

These **5 features** are concatenated to give a **40 - length feature vector.** Then we use different multiclass classifiers to obtain our result.

**CLASSIFICATION ALGORITHMS**

1. **k-Nearest Neighbors**

K nearest neighbors is an algorithm that stores all the currently classified new cases based on pervious available cases. It belongs to supervised learning domain and is non-parametric. It does not make any underlying assumptions about the distribution of data. A majority vote of the case’s neighbors classify it and each of the case being assigned to the class most common among its k-nearest neighbors is computed by a distance function. We have used Euclidean distance for computing our distance function. Generally, a higher k value reduces the overall noise. In our classification, k = 17 turned out to be the best k value.

1. **Support Vector Machine(SVM)**

SVM training algorithm builds a model which is a non-probablistic binary linear classifier used for classification. SVM finds a hyperplane which maximises the decision boundary between 2 classes. The vectors which define a hyperplane are known as support vectors. The data is mapped to a higher dimensional space after defining hyperplane. We have used the Radial Basis Function(RBF) kernel, which gives best results in case of non-linearly separable data points. This kernel in our case gives the best results.

1. **Decision Tree**

A decision tree constructs a tree structure based on regression or classification models. Dataset is divided into smaller subsets and simultaneously a tree is built. A node of the tree has 2 branches. A leaf node represents a decision or a classification. The root node corresponds to a predictor. The algorithm used for building decision trees is CART(Classification and Regression Tree). CART adopts a greedy approach in which decision trees are constructed in a top-down recursive divide-and-conquer manner. CART uses gini index as the splitting attribute for the tree. The gini index measures the impurity in a data partition. It considers the best binary split for each attribute. The point giving the minimum gini index for an attribute is taken as the split point of that attribute.

Gini(D) = 1 - ∑i=1 to m(pi2)

The reduction in impurity that would be incurred by a binary split on an attribute A is –

 ΔGini(A) = Gini(D) – GiniA(D)

1. **Random Forest**

Random forest algorithm is a supervised classification algorithm which is an ensemble of decision trees. There are more trees (dataset is divided into smaller partitions) in a forest which makes this algorithm more robust. It uses divide-and-conquer approach to improve the performance. The decision trees are weak learners whereas random forest is a strong learner. When a new input is entered, it is run down in all of the trees. The result may be average, or weighted average of the terminal nodes which are reached. Random forest is able to deal with unbalanced and missing data as well.

1. **Gradient Boosting Classifier**

A boosting algorithm which is used for reducing bias and variance in supervised learning. It is a weighted algorithm where each input class is given a weight and then is evaluated based on predicted output. The important data is given more weightage then low predicting data. This algorithm boosts the basic bagging algorithm by taking strong predictor model. The learning rate must be kept in range [0.00001 , 0.1 ] so that it takes precised inputs.

1. **Xtreme Gradient Boosting Classifier**

XGBoost belongs to a family of boosting algorithms that convert weak learners into strong learners. A weak learner is one which is slightly better than random guessing.

Boosting is a sequential process; i.e., trees are grown using the information from a previously grown tree one after the other. This process slowly learns from data and tries to improve its prediction in subsequent iterations.

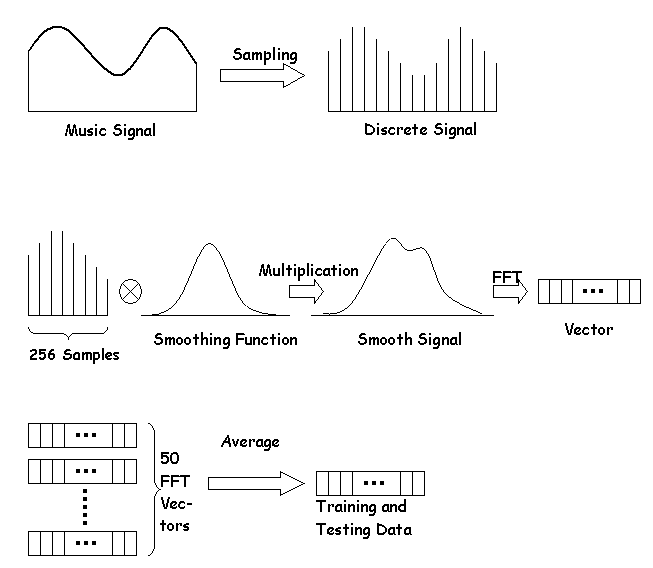
**ANALYSIS**

* **CONVERSION :**

We converted the files which were originally in .au format to .wav format because .wav format is easily compressible and can be easily read by the inbuilt python modules.

* **PRE-PROCESSING :**

The data was pre processed using fast fourier transform which divided the data file into number of frames. Given an audio clip, we first preprocess it to get our input data , that will be used both in training and testing phase. The preprocessing steps are as follows.



* **FEATURE EXTRACTION:**

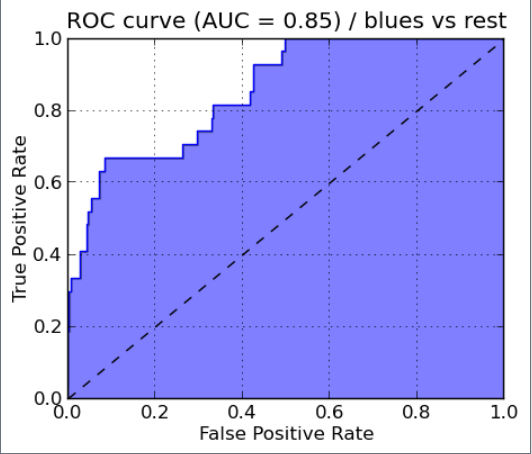
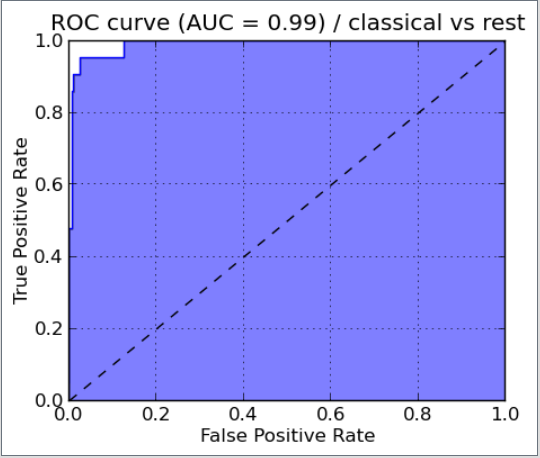
Features are used to identify the components of the audio signal that are good for identifying the linguistic content and discarding all the other stuff which carries information like background noise, emotion etc. **Librosa module** has been used to extract features.

* **STFT**

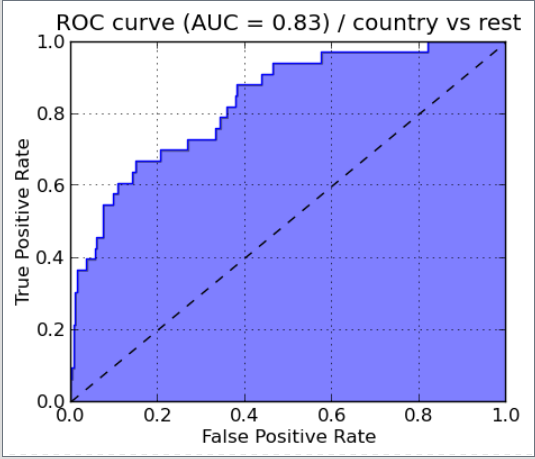
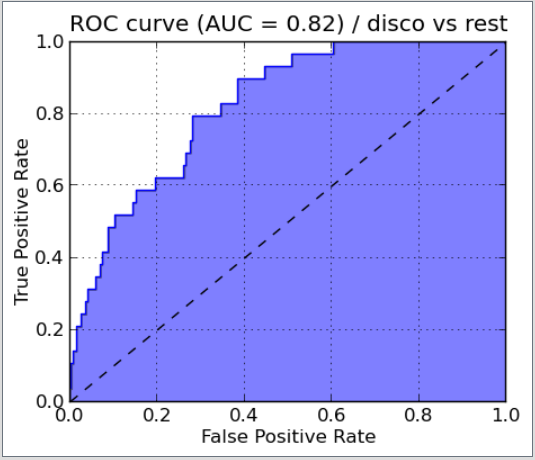
We have used STFT to calculate mel spectogram. It is used for measuring model parameters in a short-time [spectrum](http://www.dsprelated.com/dspbooks/mdft/Example_Applications_DFT.html) and short- time Fourier transform is applied to each frame to extract D -dimensional feature vector x m . Feature extraction can be seen as a mapping f : R N → R D , where D N . where s ( m, n ) = s ( n ) w ( n mL ) , and w ( n ) is a windowing function of N samples. This function is located at mL , where L is the shift-time step in samples. N represents also the number of discrete frequencies that is usually chosen to be a power-of-2 for using the fast Fourier transform (FFT). The overlap ratio between successive frames is ( N − L ) /N . The power spectrum density (PSD) is then computed.

At the sampling frequency f S , each windowed segment (frame) is represented by N -points PSD covering the frequency range [ − f S , f S [ . As power spectrum is symmetric, 2 2 it can be described by only N/ 2 discrete frequencies. Each frequency index k represents a discrete frequencies f k = kf /N , where 0 ≤ k < N/ 2.

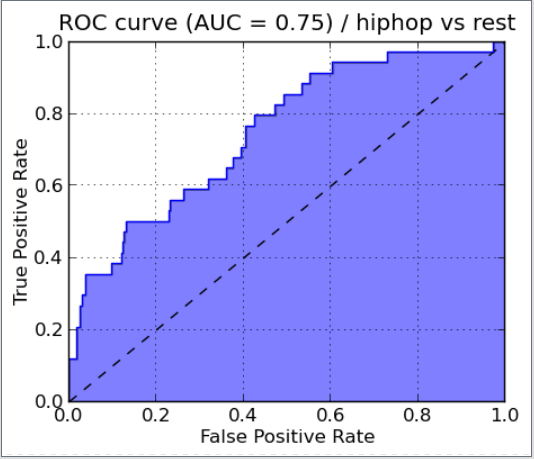
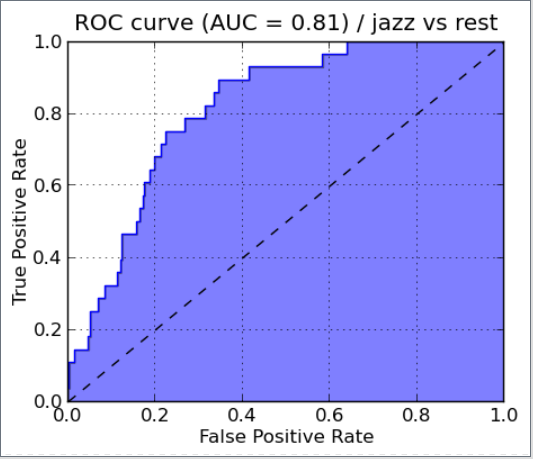
ONE VS ALL COMPARISION FOR ALL GENRES :

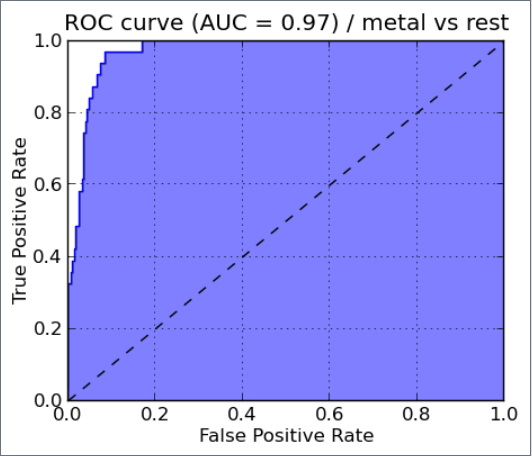
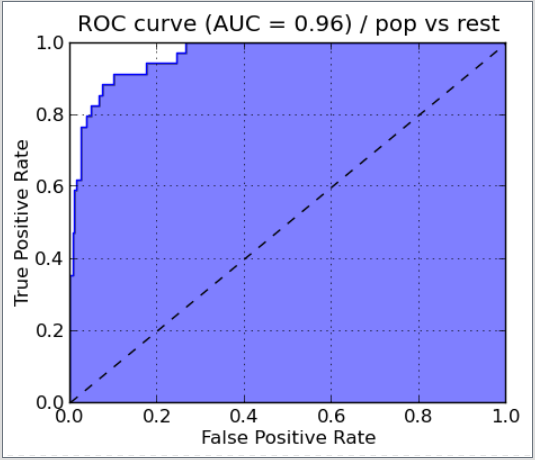
BLUES CLASSICAL

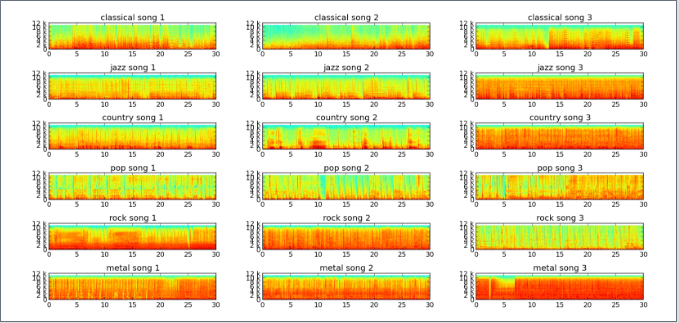
COUNTRY DISCO

HIP HOP JAZZ

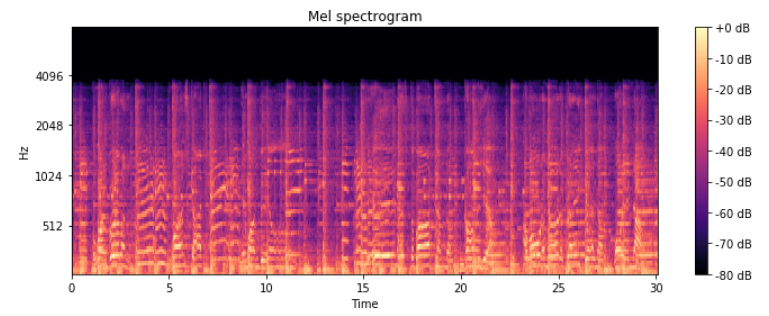
METAL POP



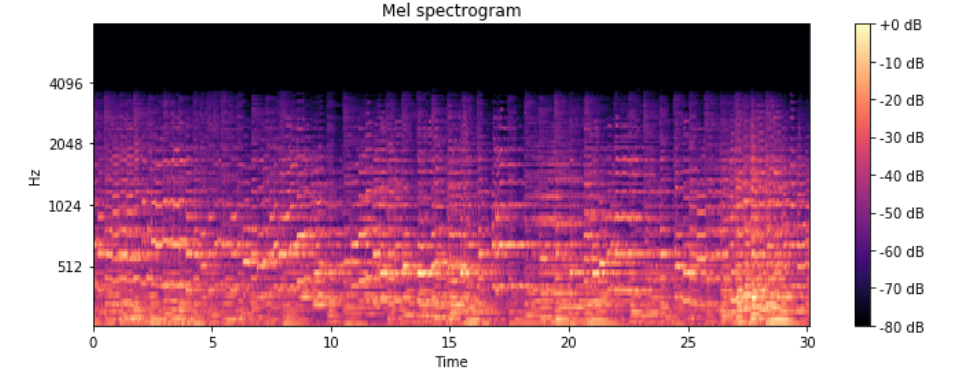
* **Mel Spectrogram and its derivatives (Delta and Delta-Delta)**:

After applying STFT cepstral features become the most important feature for the analysis of our audio file. To extract these features MFCC is used. Due to lack of ESR database MFCC features become a very important part of our analysis. It is then concatenated with other features to carry out the complete analysis work.  Proper feature optimisation must be performed because sometimes you don't need so many features, especially when they are do not separable. We got a 13 \* 325 dimension matrix as MFCC feature.

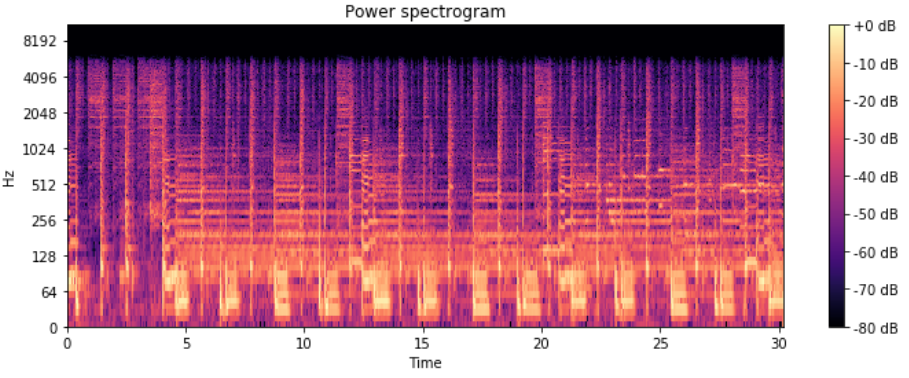
Since frames are dynamic in nature , we combine MFCC features with its differential(delta) and acceleration coefficents (delta-delta).They convey a better information about the frames and can improve the performance by about 30% .



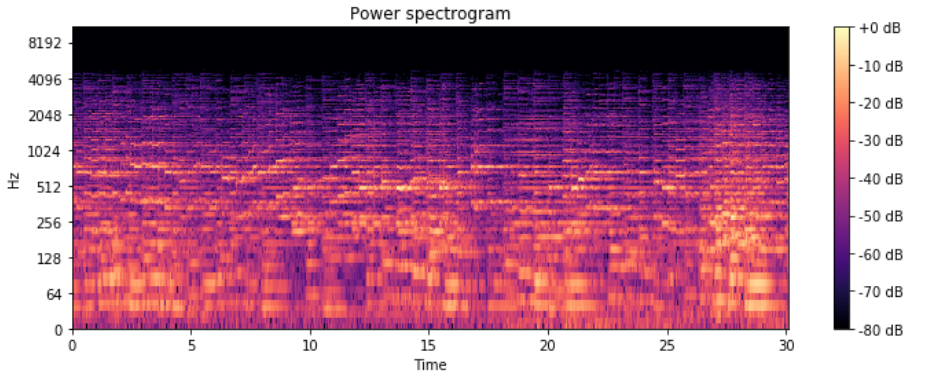
**Mel Spectrogram of Disco Genre files**

****

**Mel Spectrogram of Classical Genre files**

****

**Power Spectrogram Of Disco Genre files**

**Power Spectrogram Of Classical Genre files**

* **FEATURE REDUCTION :**

Principal Component Analysis (PCA) is the most important technique to visualize the data of dataset. It is a statistical procedure that is entirely based on orthogonal transformation , which converts some corelated values in the data set into set of linealry uncorelated data set.

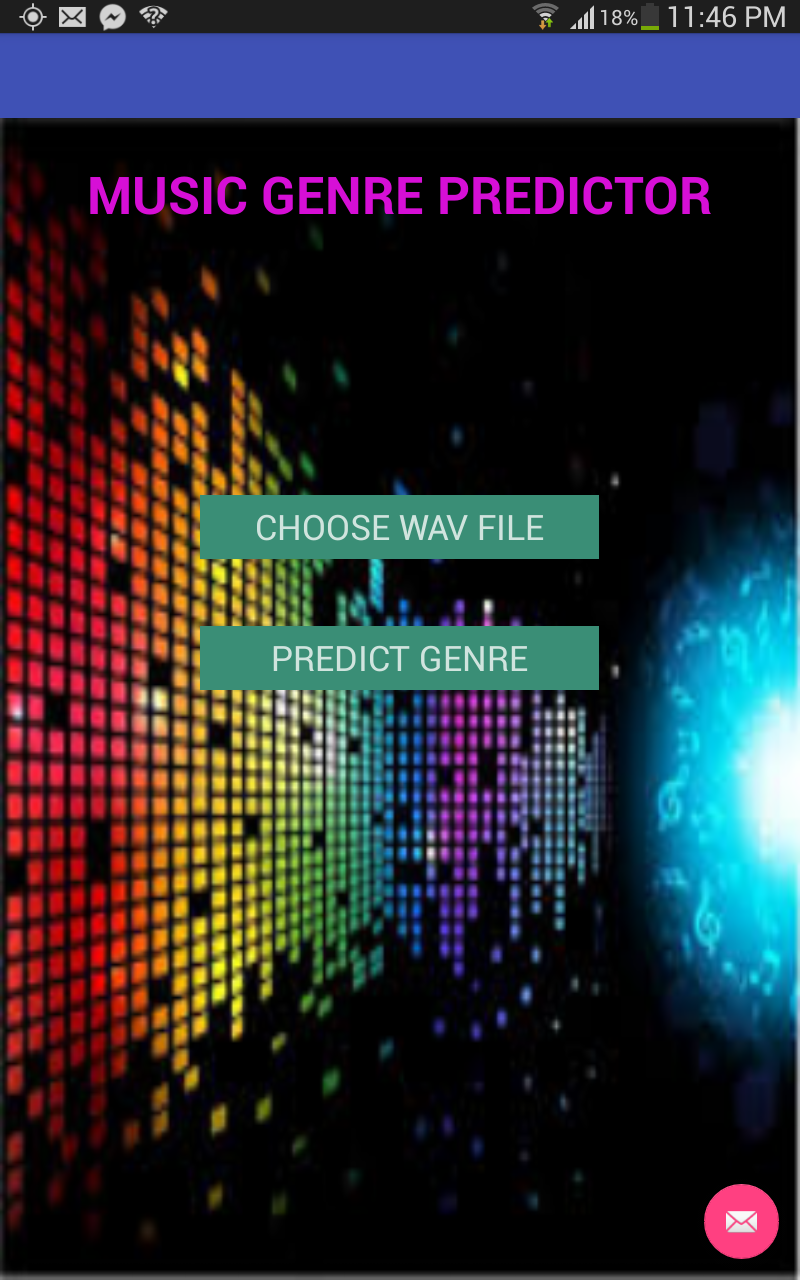
We have used PCA in our work so as to carry out the dimension reductibility. We have reduced from 40 features (13 mfcc , 13 delta ,13 delta-delta , 1 tempo) to 7 important features by applying explained variance to our data set.

* **OPTIMISATION (GRID SEARCH) :**

The most effective way of hyperparameter optimisation (C, gamma, kernel in case of SVM) is by the use of grid search. It basically does an exhaustive searching through a specified subset of hyperparameter. For our dataset we got the best results for '**C**': 10000, '**Gamma**': 0.0005, '**Kernel**': 'rbf'. There are many metrics for its evaluation but the most effective is cross validation , so we used it.

Till date, all those researchers who have worked in the field of music genre classification have not used grid search CV hyperparameter optimization to optimize their results. We have optimized our results using this technique which further improved our results.

**INTEGRATION WITH ANDROID**

****

**UI OF APPLICATION**

**WORKING OF APPLICATION with FLASK API:**

This App is integrated with Flask based API which act as a wrapper for our trained model. In this application, We have used HTTP Request class for interacting with the API.

The app reads the “.wav” file selected from file media and converts the music file into long string values (the values are basically amplitude of file sampled at 44100 Hz). The file is then written to data output stream class which is then send to Flask API (running on local server). The file received by the server is preprocessed (where main features such as MFCC , delta , delta order-2 are extracted). Then, these extracted features are used to make prediction through our trained model. The predicted value is recorded and finally sent back to application as a HTTP response.

**RESULTS**

We tested the dataset on different classifiers. Different classifiers gave different accuracies. The various parameters we have used to evaluate the result of our testing data are accuracy, precision, F1 score and recall.

* **ACCURACY**

Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations.

Accuracy = TP+TN/TP+FP+FN+TN

* **PRECISION**

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.

Precision = TP/TP+FP

* **RECALL**

Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes.

Recall = TP/TP+FN

* **F1 SCORE**

F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account.

F1 Score = 2\*(Recall \* Precision) / (Recall + Precision)

**USING PCA REDUCTION**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Classifier | Precision | Recall | F1 score | Support |
| Random Forest Classifier  K-Neighbors  SVM Classifier  Decision Tree  XG Boost  Gradient Boost  SVM with Grid Search CV | 0.44  0.46  0.51  0.40  0.47  0.51  0.53 | 0.45  0.42  0.51  0.41  0.47  0.52  0.53 | 0.43  0.41  0.50  0.39  0.47  0.51  0.53 | 200  200  200  200  200  200  200 |

The accuracies of different classifiers are mentioned below:

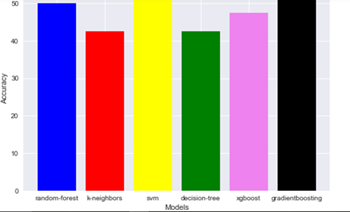
* Gradient Boost : accuracy = 52 %
* SVM : accuracy = 51 %
* Random Forest : accuracy = 50 %
* Decision Tree : accuracy = 42.5 %
* K nearest neighbors : accuracy = 42.5 %

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Classifier | Precision | Recall | F1 score | Support |
| Random Forest Classifier  K-Neighbors  SVM Classifier  Decision Tree  XG Boost  Gradient Boost  SVM with Grid Search CV | 0.50  0.32  0.40  0.48  0.56  0.53  0.53 | 0.49  0.34  0.24  0.46  0.56  0.54  0.53 | 0.49  0.31  0.24  0.45  0.55  0.53  0.53 | 200  200  200  200  200  200  200 |

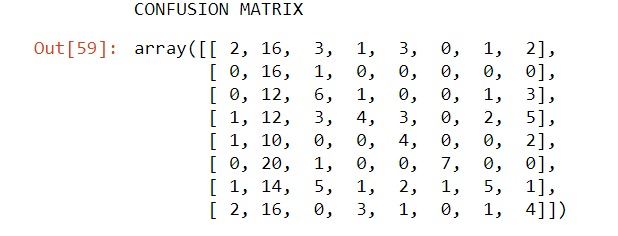
**WITHOUT PCA REDUCTION**

The accuracies of different classifiers are mentioned below:

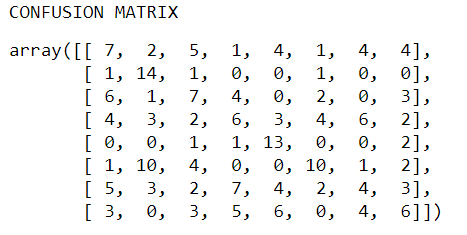
* Gradient Boost : accuracy = 53.5 %
* SVM : accuracy = 42.5 %
* Random Forest : accuracy = 49.5 %
* Decision Tree : accuracy = 45.5 %
* K nearest neighbors : accuracy = 33.5 %
* XG Boost : accuracy = 56 %



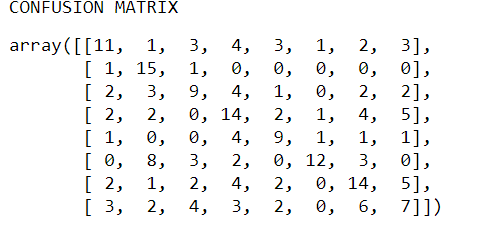
**CONFUSION MATRICES**

****

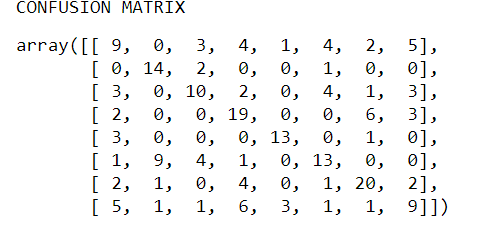
**SVM**

****

**KNN**

****

**DECISION TREE**



**GRADIENT BOOST**

**RESULT INTERPRETATION:**

Decision tree works well when the ratio of number of samples to number of features is balanced since in this situation there is no overfitting of data.

Decision Trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble,i.e. Random Forest.

SVM worked well in our dataset since it works well in high dimensional data and our dataset was high dimensional. It didn’t work as much as expected since the number of dimensions are greater than number of samples.

KNN gave the least accuracy since when we increased the value of k to increase the accuracy , it surpressed the effects of noise but makes the classification boundaries less distinct.

**FUTURE SCOPE**

We have classified the audio files into single genre in our project. We can extend this project to multi-class genre classification. We can use deep learning and neural networks to achieve this multi-class genre classification.

**REFERENCES**

1. G. Tzanetakis and P. Cook. Musical genre classification of audio signals. IEEE  Transactions on Speech and Audio Processing, 10(5):293–302, 2002.
2. Khunarsal,P.;Lursinsap,C.;Raicharoen,T.:Veryshorttimeenviron- mental sound classi cation based on spectrogram pattern matching. 2013, (in press). [Online]. Available: http://www.sciencedirect.com/ science/article/pii/S0020025513003113
3. Long Short Term Memory RNN Wikipedia: https://en.wikipedia.org/wiki/Long\_short-term\_memory.
4. Spicy: Discrete Fourier Transform Pack: http://docs.spicy.org/doc/spicy/refernce/fftpack.html
5. Pradeep Kumar D, Sowmya B J, Chetan, K G Srinivasa , A Comparative Study of Classifiers for Music Genre Classification based on Feature Extractors.
6. http://marsyas.info/downloads/datasets.html

|  |
| --- |
|  |
|  |