A PROJECT REPORT ON

MUSIC GENRE CLASSIFICATION

BY

Jadhao Nikhil Sanjay	Roll No-1321
Salunkhe Ashutosh Ashok	Roll No -1342

Under the Guidance of

Mr.AKSHAY TILEKAR

POST GRADUATE DIPLOMA IN BIG DATA ANALYTICS FEB-2020

INSTITUTE FOR ADVANCED COMPUTING AND SOFTWARE DEVELOPMENT, AKURDI, PUNE

CERTIFICATE

This is to certify that the Project Entitled

MUSIC GENRE CLASSIFICATION

Submitted by:

Jadhao Nikhil Sanjay	**Roll No-1321**
Salunkhe Ashutosh Ashok	**Roll No -1342**

is a bonafide student of this institute and the work has been carried out by him/her under the supervision of Mr. AKSHAY TILEKARand it is approved for the partial fulfillment of the requirement of Post Graduate Diploma in Big Data Analytics.

Mr. Prashant Karhale	**Mr. Akshay Tilekar**
Centre Coordinator	**Project Guide**

ACKNOWLEDGEMENT

It gives us great pleasure in presenting the preliminary project report on 'MUSIC GENRE CLASSIFICATION'.

I would like to take his opportunity to thank my internal guide Mr. AKSHAY TILEKAR for giving me all the help and guidance I needed. I am really grateful to them for their kind support. Their valuable suggestions were very helpful.

I am also grateful to Mr. PRASHANT KARHALE, Centre Coordinator, Akurdi , Pune for his indispensable support, suggestions. In the end our special thanks to IACSD for providing various resources such as laboratory with all needed software platforms, continuous Internet connection, for our Project.

JADHAO NIKHIL SANJAY

SALUNKHE ASHUTOSH ASHOK

INDEX

Table of Contents

List of Figures and Tables

CHAPTER 1

Introduction

Sound is represented in the form of an audio signal having parameters like frequency, decibel, bandwidth etc. A typical audio signal can be expressed as a function of Amplitude and Time. These audio signals come in various different formats which make it possible and easy for the computer to read and analyze them. Some formats are: mp3 format, WMA (Windows Media Audio) format and wav (Waveform Audio File) format.

Companies (such as Soundcloud, Apple Music, Spotify, Wynk and etc) use music classification, either to place recommendations to their customers, or simply as a product (like Shazam). To be able to do any of the above two mentioned functions, determining music genres is the first step. To achieve this, we can take the help of Machine Learning algorithms. These machine learning algorithms prove to be very handy in Music Analysis too.

Music Analysis is done based on a song's digital signatures for some factors, including acoustics, danceability, tempo, energy etc., to determine the kind of songs that a person is interested to listen to. Music is characterized by giving them categorical labels called genres. These genres are created by humans. A music genre is segregated by the characteristics which are commonly shared by its members. Typically, these characteristics are related to the rhythmic structure, instrumentation, and the harmonic content of the music. To categorize music files into their respective genres, it is a very challenging task in the area of Music Information Retrieval (MIR), a field concerned with browsing, organizing and searching large music collections.

Classification of genre can be very valuable to explain some interesting problems such as creating song references, tracking down related songs, discovering societies that will like that specific song, sometimes it can also be used for survey purposes.

Automatic musical genre classification can assist humans or even replace them in this process and would be of a very valuable addition to music information retrieval systems. In addition to this, automatic classification of music into genres can provide a framework for development and evaluation of features for any type of content-based analysis of musical signals.

The concept of automatic music genre classification has become very popular in recent years as a result of the rapid growth of the digital entertainment industry. Dividing music into genres is arbitrary, but there are perceptual criteria that are related to instrumentation, structure of the rhythm and texture of the music that can play a role in characterizing particular genre. Until now genre classification for digitally available music has been performed manually. Thus, techniques for automatic genre classification would be a valuable addition to the development of audio information retrieval systems for music.

1.1 Problem Statement:

Music plays a very important role in people’s lives. Music bring like-minded people together and is the glue that holds communities together. Communities can be recognized by the type of songs that they compose, or even listen to. Different communities and groups listen to different kinds of music. One main feature that separates one kind of music from another is the genre of the music. The aim of this project is:

1. To build a machine learning model which classifies music into its respective genre.

2. To compare the accuracies of this machine learning model and the pre-existing models.

1.2 Objective :

1. Developing a machine learning model that classifies music into genres shows that there exists a solutions which automatically classifies music into its genres based on various different features, instead of manually entering the genre.

2. Another objective is to reach a good accuracy so that the model classifies new music into its genre correctly.

3. This model should be better than at least a few preexisting models.

1.3 Software Life Cycle Model:

In order to make this Project we are going to use Classic LIFE CYCLE MODEL .Classic life cycle model is also known as WATER FALL MODEL. The life cycle model demands a Systematic sequential approach to software development that begins at the system level and process through analysis design coding , testing and maintenance.

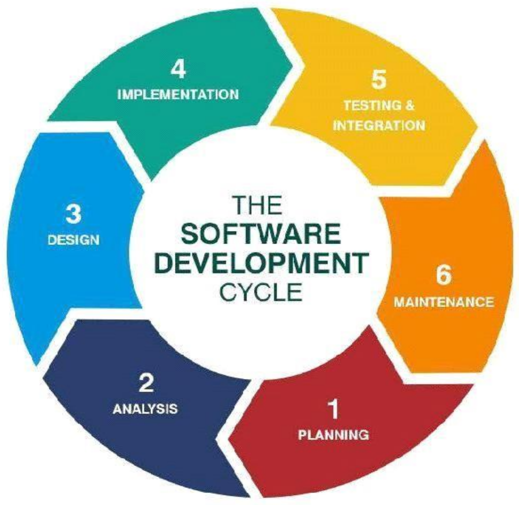


Figure 1- Software life cycle model

1.3.1 The Classic Life Cycle Model :

The waterfall model is sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of conception initiation Analysis, Design (validation), construction. Testing and Maintained.

SYSTEM ENGINEERING AND ANALYSIS:

Because software is always a part of larger system work. Begins by establishing requirement for all system elements and Then allocating some subset of these requirement to the software system Engineering and analysis encompasses the requirement gathering at the system level with a small amount of top level design and analysis.

SOFTWARE REQUIREMENT ANALYSIS:

The requirement gathering process is intensified and focused specifically on the software Requirement for the both system and software are discussed and reviewed with the customer. The customer specifies the entire requirement or the software and the function to be performed by the software.

DESIGN:

Software design is actually a multi-step process that focuses on distinct attributes of the program data structure, software architecture, procedural detail and interface of the software that can be assessed or quality before coding begins .Like requirement the design is documented and becomes part of the software.

CODING:

The design must be translated into a machine readable form. The coding step performs this task. If design is programmed in a detailed manner, coding can be accomplished mechanically.

TESTING:

Once code has been generated programmed testing begins. The testing process focuses on the internals of the software ensuring that all statement have been tested and on the functional externals hat is conducting tests to uncover the errors and ensure that defined input will produce the results that agree with the required results.

Unit testing:

In computer programming, Unit testing is software Verification and validation method where the programmer gains confidence that individual units of source code are fit to use A unit is the smallest testable part of an application.

In procedural programming a unit may be an individual programmed, function, procedure, etc. while in object-oriented programming, the smallest Unit is a class, which may belong to a base/super class abstract class or derived/child class.

Benefits:

The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict written contract that the piece of code must satisfy.

Documentation:

Unit testing provides a sort of living documentation of the system. Developers looking to learn what functionality is provided by a unit and how to use it can look at the tests to gain a basic understanding of the unit API.

Limitation of unit testing:

Testing cannot be expected to catch error in the program –It is impossible to evaluate all execution paths for all but the most trivial programs. The same is true for unit testing. Additionally, by unit testing only types the functionality if the units themselves.

MAINTENANCE:

Software will undoubtedly undergo change after it is Delivered to the customer .Change will occur because errors have been encountered because the software must be able adopted to accommodate changes in its external environment because the customer requires functional or performance enhancement enhancements. The classic life cycle is the oldest and most widely used paradigm or software engineering

CHAPTER 2

Overall Description

2.1 System Environment

The system environment is evolving around the ML which will classify the genre based on frequency and time domain

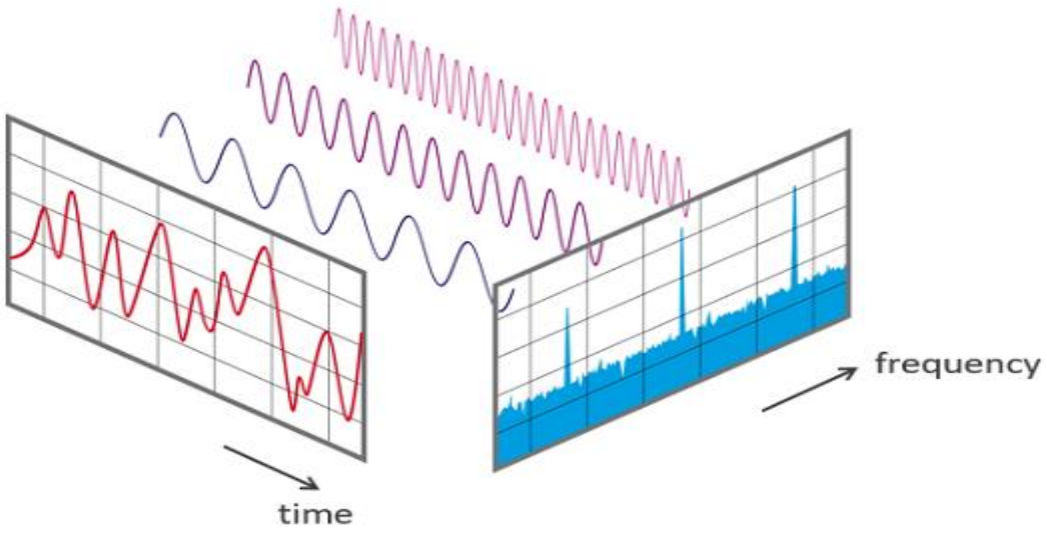
2.2 Functional requirement specification:

Use case: classify the genre based on frequency and time domain

Brief Description:

Audio Data Handling using Python

Sound is represented in the form of an **audio** signal having parameters such as frequency, bandwidth , decibel, etc. A typical audio signal can be expressed as a function of Amplitude and Time.



There are devices built that help you catch these sounds and represent it in a computer-readable format. Examples of these formats are

- wav (Waveform Audio File) format
- mp3 (MPEG-1 Audio Layer 3) format
- WMA (Windows Media Audio) format

A typical audio processing process involves the extraction of acoustics features relevant to the task at hand, followed by decision-making schemes that involve detection, classification, and knowledge fusion. Thankfully we have some useful python libraries which make this task easier.

Audio Libraries :

Python has some great libraries for audio processing like Librosa and PyAudio. There are also built-in modules for some basic audio functionalities.

1.Librosa:

It is a Python module to analyze audio signals in general but geared more towards music. It includes the nuts and bolts to build a MIR(Music information retrieval) system.
Installation:

```
pip install librosa
or
conda install -c conda-forge librosa
```

To fuel more audio-decoding power, you can install *ffmpeg* which ships with many audio decoders.

2.IPython.display.Audio

IPython.display.Audio lets you play audio directly in a jupyter notebook.

2.3 Classifier :

This section provides a brief overview of the four machine learning classifiers adopted in this study.

1. K-Nearest Neighbor (K-NN):

A k-nearest-neighbor algorithm, often abbreviated K-NN , is an approach to data classification that estimates how likely a data point is to be a member of one group or the other depending on what group the data points nearest to it are in.

2. Logistic Regression (LR):

Logistic regression is used to obtain odds ratio in the presence of more than one explanatory variable. The procedure is quite similar to multiple linear regression, with the exception that the response variable is binomial. The result is the impact of each variable on the odds ratio of the observed event of interest.

3. Random Forest (RF):

Random Forest is a ensemble learner that combines the prediction from a pre-specified number of decision trees. It works on the integration of two main principles:

1) each decision tree is trained with only a subset of the training samples which is known as bootstrap aggregation (or bagging) (Breiman, 1996),

2) each decision tree is required to make its prediction using only a random subset of the features (Amit and Geman, 1997). The final predicted class of the RF is determined based on the majority vote from the individual classifiers.

4. Support Vector Machines (SVM):

SVMs transform the original input data into a high dimensional space using a kernel trick (Cortes and Vapnik, 1995). The transformed data can be linearly separated using a hyperplane. The optimal hyperplane maximizes the margin. In this study, a radial basis function (RBF) kernel is used to train the SVM because such a kernel would be required to address this non-linear problem. Similar to the logistic regression setting discussed above, the SVM is also implemented as a one-vs-rest classification task.

CHAPTER 3

DATASET

The GTZAN genre collection dataset was collected in 2000-2001. It consists of 1000 audio files each having 30 seconds duration. There are 10 classes (10 music genres) each containing 100 audio tracks. Each track is in .wav format. It contains audio files of the following 10 genres:

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

CHAPTER 4
System Design

4.1 Flowchart of the System:

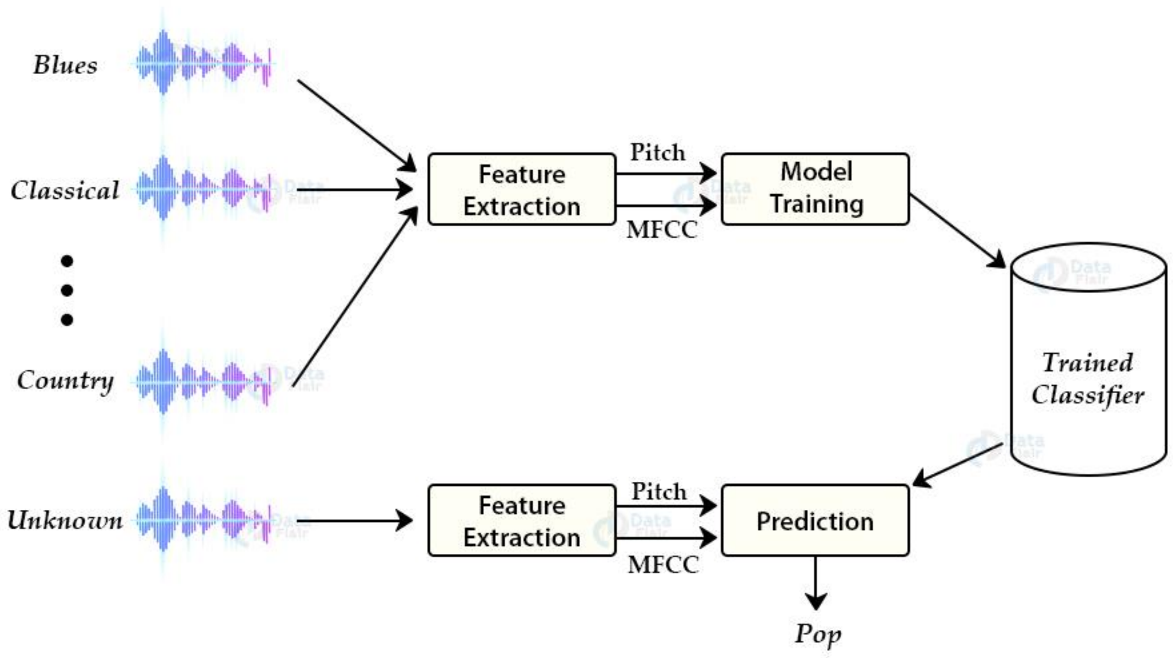


Figure 2 – Flowchart of project

The above flow chart show how the process is going to take place

- Firstly , the input is applied to system which is audio file in this project which consist of 1000 audio file.

- Secondly, Feature extracted from that audio files which are in .wav format
- After extracting the features we get features like Mffcs , Zero-crossing rate , chroma Features , spectral centroid , spectral roll-of, spectral bandwidth.
- This extracted features are used to train the model
- And after training the model the system will used the model which gives best accuracy for classification.
- And based on that classifier the system will do the predication.
- After the training the model the system will take a unknown .wav file   to test the system works properly or not, based the classifier which has been trained.
- Again the features are extracted from the unknown .wav file or testing .wav and output or genre prediction is done.

  .

Chapter 5

Methodology

5.1 Extraction of features :

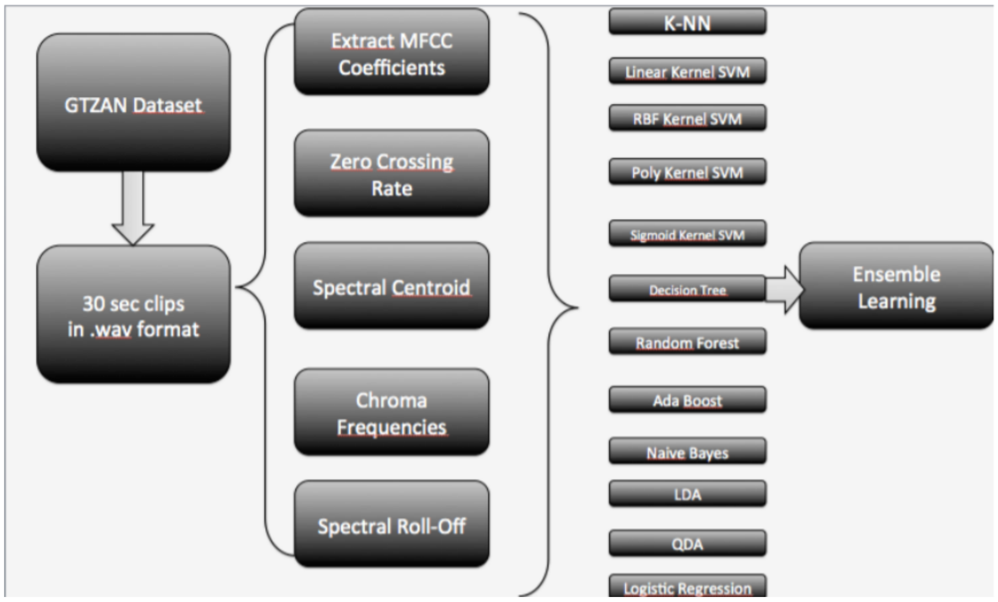
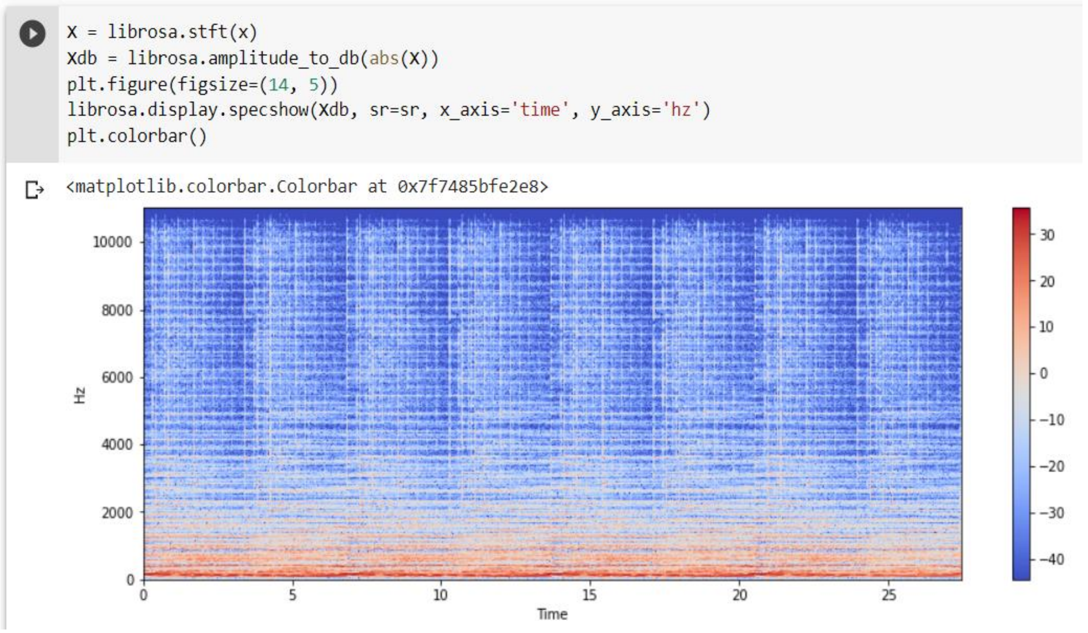


Figure 3 - Diagrammatic representation of the method

Spectrogram:

A **spectrogram** is a visual way of representing the signal strength, or "loudness", of a signal over time at various frequencies present in a particular waveform. Not only can one see whether there is more or less energy at, for example, 2 Hz vs 10 Hz, but one can also see how energy levels vary over time.

Spectrogram



```
X = librosa.stft(x)
Xdb = librosa.amplitude_to_db(abs(X))
plt.figure(figsize=(14, 5))
librosa.display.specshow(Xdb, sr=sr, x_axis='time', y_axis='hz')
plt.colorbar()
```

```
<matplotlib.colorbar.Colorbar at 0x7f7485bfe2e8>
```

Figure 4 -Spectrogram

Above spectrogram is the raw seismogram, drawn using the same horizontal time axis as the spectrogram (including the same tick marks), with the vertical axis representing wave amplitude. This plot is analogous to webicorder-style plots (or seismograms) that can be accessed via other parts of our website. Collectively, the spectrogram-seismogram combination is a very powerful visualization tool, as it allows you to see raw waveforms for individual events and also the strength or "loudness" at various frequencies. The frequency content of an event can be very important in determining what produced the signal.

Time Domain Features:

These are the features which were extracted from the raw audio signal.

- **Central moments:** This consists of the mean, standard deviation, skewness and kurtosis of the amplitude of the signal.
- **Zero Crossing Rate (ZCR):** This point is where the signal changes sign from positive to negative. The entire 30 seconds is divided into smaller frames, and the number

of zero-crossings present in each frame are determined. The average and Standard deviation of the ZCR across all the frames are chosen as representative features.

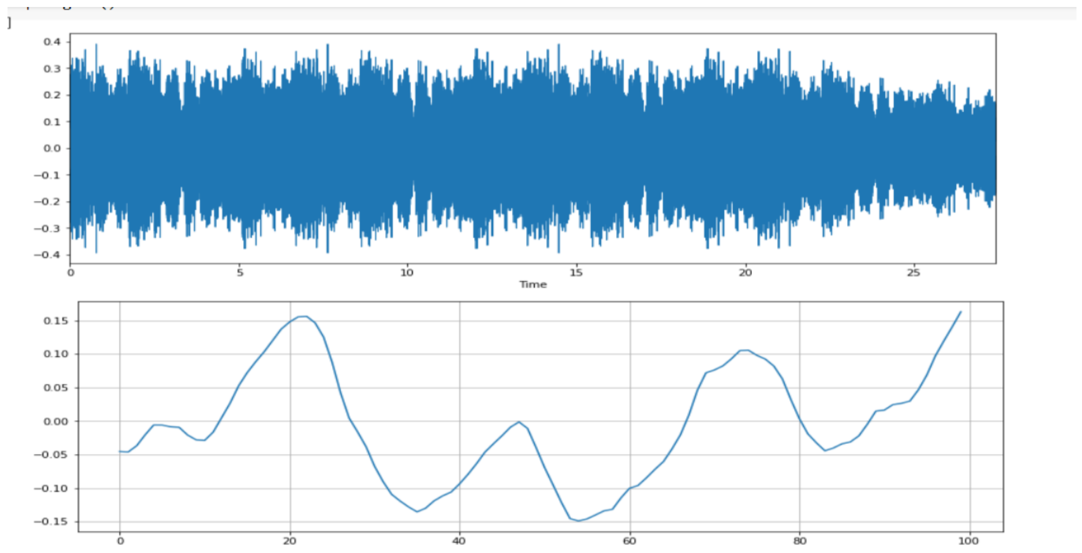


Figure 5- Zero-Crossing Rate

- **Root Mean Square Energy (RMSE) : The energy signal in a signal is calculated as**

$$\sum_{n=0}^{N} |x(n)|\ \hat{}2$$

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

Frequency Domain Features:

The audio signal is first transformed into the frequency domain using the Fourier Transform. Then the following features are extracted.

- **Mel-Frequency Cepstral Coefficients (MFCC):** Mel-frequency cepstral coefficients (**MFCCs**) are coefficients that collectively make up an MFC. They are derived from a type of cepstral representation of the audio clip (a nonlinear "spectrum-of-a-spectrum").

```
(20, 1181)
<matplotlib.collections.QuadMesh at 0x7faf6bb54320>
```

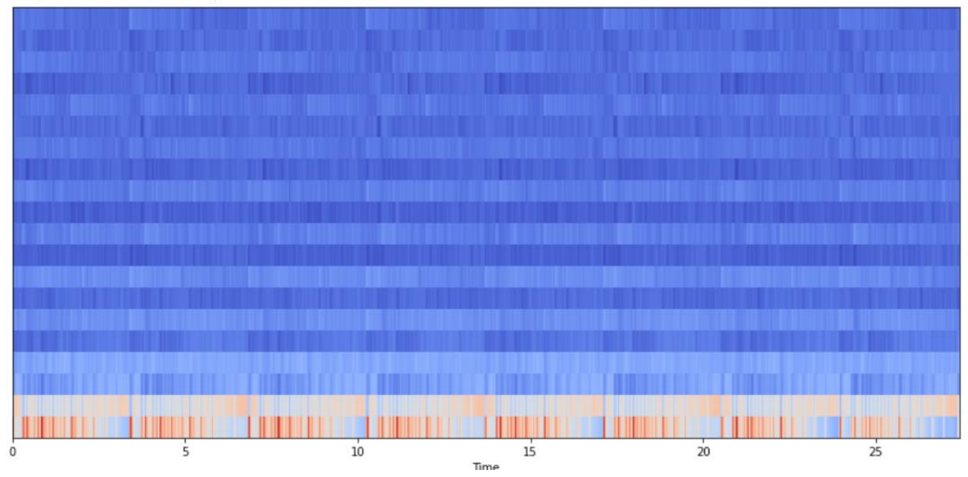


Figure 6 - MFFCs

- **Chroma Features:** This is a vector which 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). Then the aggregate of the chroma vectors is taken to get the mean and standard deviation.

6 . Chroma feature

```
[13] hop_length = 512
     chromagram = librosa.feature.chroma_stft(x, sr=sr, hop_length=hop_length)
     plt.figure(figsize=(15, 5))
     librosa.display.specshow(chromagram, x_axis='time', y_axis='chroma', hop_length=hop_length, cmap='coolwarm')
```

```
<matplotlib.collections.QuadMesh at 0x7faf6bd77550>
```

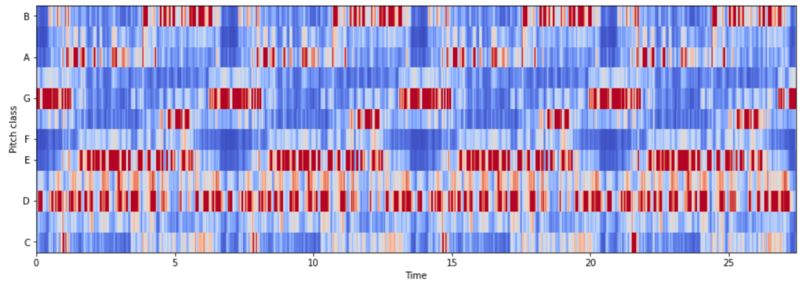
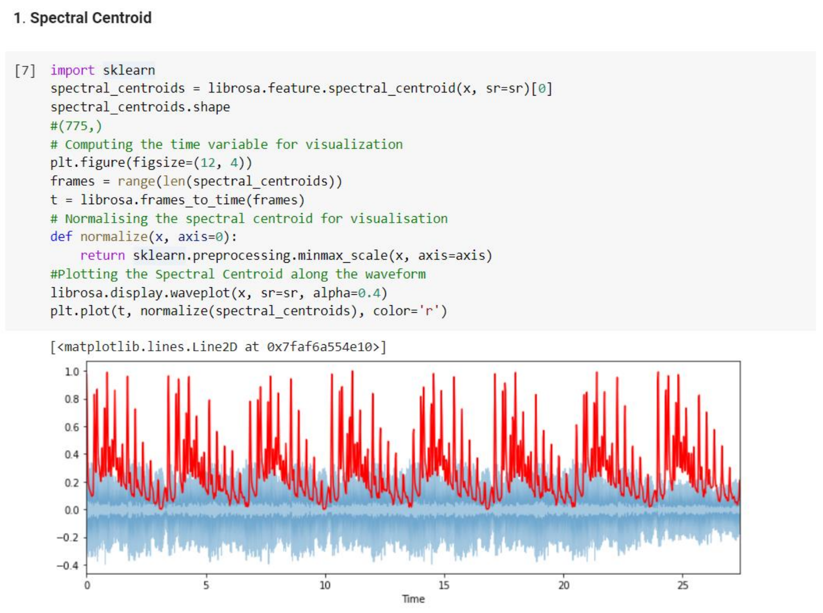


Figure 7- Chroma Features

- **Spectral Centroid:** This corresponds to the frequency around which most of the energy is centered.

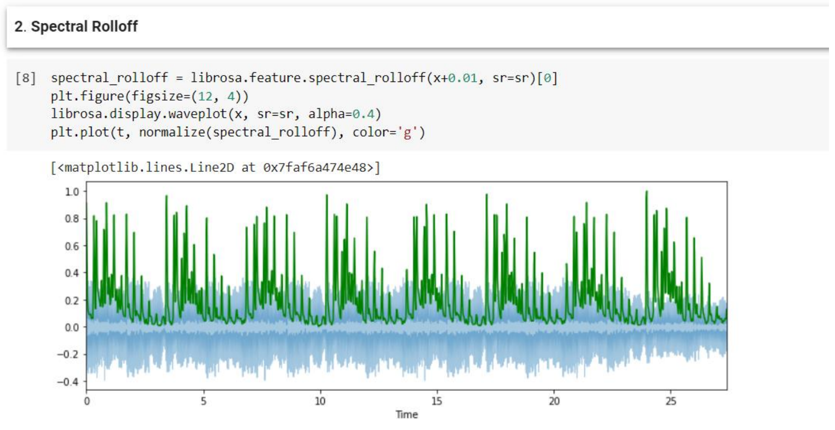

1. Spectral Centroid

```
[7] import sklearn
    spectral_centroids = librosa.feature.spectral_centroid(x, sr=sr)[0]
    spectral_centroids.shape
    #(775,)
    # Computing the time variable for visualization
    plt.figure(figsize=(12, 4))
    frames = range(len(spectral_centroids))
    t = librosa.frames_to_time(frames)
    # Normalising the spectral centroid for visualisation
    def normalize(x, axis=0):
        return sklearn.preprocessing.minmax_scale(x, axis=axis)
    #Plotting the Spectral Centroid along the waveform
    librosa.display.waveplot(x, sr=sr, alpha=0.4)
    plt.plot(t, normalize(spectral_centroids), color='r')
```

```
[<matplotlib.lines.Line2D at 0x7faf6a554e10>]
```

Figure 8- Spectral Centroids

- **Spectral Contrast:** Each frame is divided into a prespecified number of frequency bands. And, within each frequency band, the spectral contrast is calculated as the difference between the maximum and minimum magnitudes.
- **Spectral Roll-off:** This feature corresponds to the value of frequency below which 85% of the total energy in the spectrum lies. 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 thmodel.

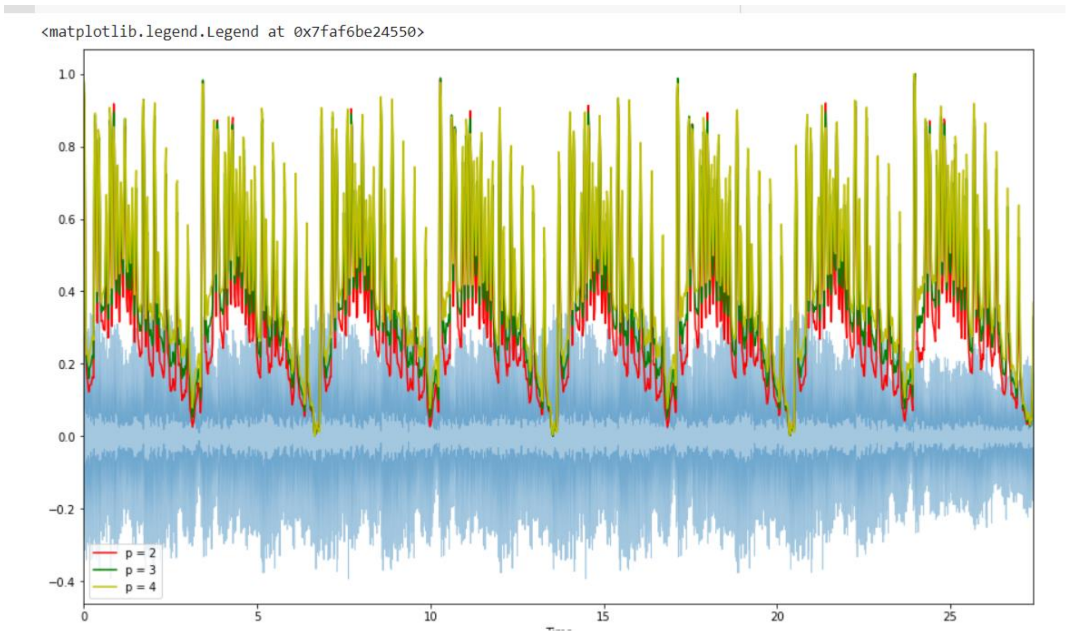

2. Spectral Rolloff

```
[8] spectral_rolloff = librosa.feature.spectral_rolloff(x+0.01, sr=sr)[0]
    plt.figure(figsize=(12, 4))
    librosa.display.waveplot(x, sr=sr, alpha=0.4)
    plt.plot(t, normalize(spectral_rolloff), color='g')
```

```
[<matplotlib.lines.Line2D at 0x7faf6a474e48>]
```

Figure 9- Spectral Rolloff

- **Spectral Bandwidth:** The **spectral bandwidth** of a spectrophotometer is related to the physical slit-width and optical dispersion of the monochromator system.



```
<matplotlib.legend.Legend at 0x7faf6be24550>
```

Figure 10 - Spectral Bandwidth

librosa.feature.spectral_bandwidth computes the order-pp spectral bandwidth:

(∑kS(k)(f(k)−fc)p)1p(∑kS(k)(f(k)−fc)p)1p

where S(k)S(k) is the spectral magnitude at frequency bin kk,
f(k)f(k) is the frequency at bin kk,
and fcfc is the spectral centroid.
When p=2, this is like a weighted standard deviation.

CHAPTER 6

RESULT

6.1 Model Accuracy:

The below table represent the accuracy before and after tunning model.

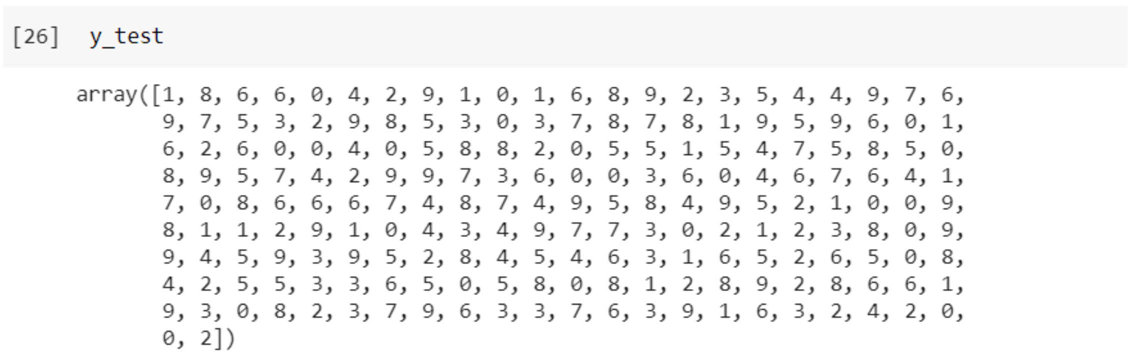
Models	Accuracy	Accuracy after tunning
K-Nearest Neighbor (K-NN)	**62.50%**	**64.75%**
Logistic Regression (LR)	**64.23%**	**65.13%**
Random Forest (RF)	**64.45%**	**75.54%**
Support Vector Machines (SVM)	**65.50%**	**73%**

Table 1 - Model Accuracy

From the above table it can be observe that the best accuracy is given by **Random Forest (RF)** that is **75.54%.**

6.2 Model Result :

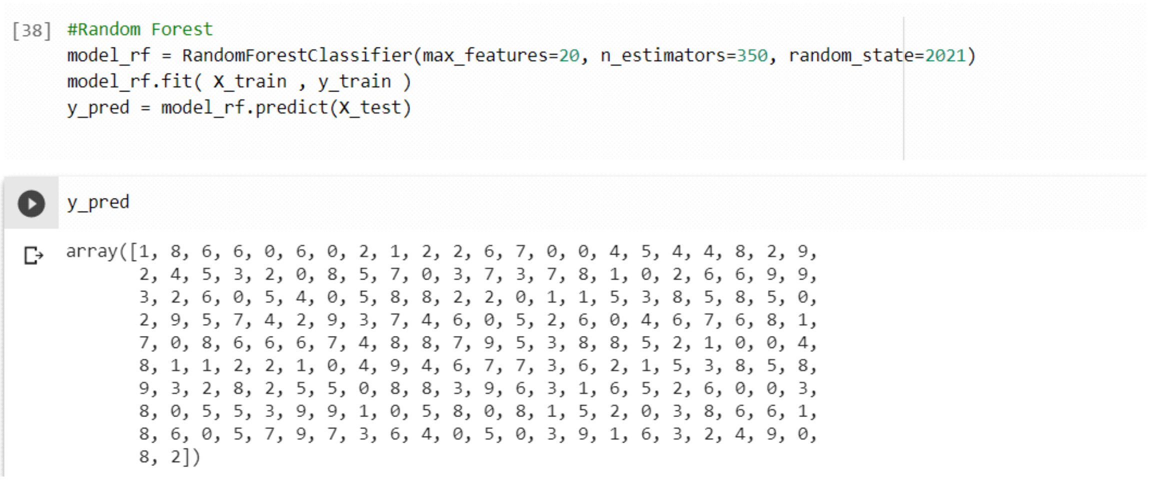
The above table show as that the test accuracy for classification is given by **Random Forest ,** so Random Forest is select in this project for classification system. The based on it system is used for the classification which is shown below.



```
[26]  y_test

      array([1, 8, 6, 6, 0, 4, 2, 9, 1, 0, 1, 6, 8, 9, 2, 3, 5, 4, 4, 9, 7, 6,
             9, 7, 5, 3, 2, 9, 8, 5, 3, 0, 3, 7, 8, 7, 8, 1, 9, 5, 9, 6, 0, 1,
             6, 2, 6, 0, 0, 4, 0, 5, 8, 8, 2, 0, 5, 5, 1, 5, 4, 7, 5, 8, 5, 0,
             8, 9, 5, 7, 4, 2, 9, 9, 7, 3, 6, 0, 0, 3, 6, 0, 4, 6, 7, 6, 4, 1,
             7, 0, 8, 6, 6, 6, 7, 4, 8, 7, 4, 9, 5, 8, 4, 9, 5, 2, 1, 0, 0, 9,
             8, 1, 1, 2, 9, 1, 0, 4, 3, 4, 9, 7, 7, 3, 0, 2, 1, 2, 3, 8, 0, 9,
             9, 4, 5, 9, 3, 9, 5, 2, 8, 4, 5, 4, 6, 3, 1, 6, 5, 2, 6, 5, 0, 8,
             4, 2, 5, 5, 3, 3, 6, 5, 0, 5, 8, 0, 8, 1, 2, 8, 9, 2, 8, 6, 6, 1,
             9, 3, 0, 8, 2, 3, 7, 9, 6, 3, 3, 7, 6, 3, 9, 1, 6, 3, 2, 4, 2, 0,
             0, 2])
```

Model saving and Loading



```
[38] #Random Forest
     model_rf = RandomForestClassifier(max_features=20, n_estimators=350, random_state=2021)
     model_rf.fit( X_train , y_train )
     y_pred = model_rf.predict(X_test)
```

```
y_pred

array([1, 8, 6, 6, 0, 6, 0, 2, 1, 2, 2, 6, 7, 0, 0, 4, 5, 4, 4, 8, 2, 9,
       2, 4, 5, 3, 2, 0, 8, 5, 7, 0, 3, 7, 3, 7, 8, 1, 0, 2, 6, 6, 9, 9,
       3, 2, 6, 0, 5, 4, 0, 5, 8, 8, 2, 2, 0, 1, 1, 5, 3, 8, 5, 8, 5, 0,
       2, 9, 5, 7, 4, 2, 9, 3, 7, 4, 6, 0, 5, 2, 6, 0, 4, 6, 7, 6, 8, 1,
       7, 0, 8, 6, 6, 6, 7, 4, 8, 8, 7, 9, 5, 3, 8, 8, 5, 2, 1, 0, 0, 4,
       8, 1, 1, 2, 2, 1, 0, 4, 9, 4, 6, 7, 7, 3, 6, 2, 1, 5, 3, 8, 5, 8,
       9, 3, 2, 8, 2, 5, 5, 0, 8, 8, 3, 9, 6, 3, 1, 6, 5, 2, 6, 0, 0, 3,
       8, 0, 5, 5, 3, 9, 9, 1, 0, 5, 8, 0, 8, 1, 5, 2, 0, 3, 8, 6, 6, 1,
       8, 6, 0, 5, 7, 9, 7, 3, 6, 4, 0, 5, 0, 3, 9, 1, 6, 3, 2, 4, 9, 0,
       8, 2])
```

CHAPTER 7

Future scope

Firstly, This Project can be used for Recommendation system to recommend the music based on genre.

Secondly, more features can be used like tempo, pitch etc. for better accuracy which will give as better classification .

Thirdly, by using Spectrogram images this project can further Implement Deep Learning method which can be used like Convolutional Neural Networks which can give as better performance.

CHAPTER 8

Conclusion

This project is based on automated system for music genre classification. MFCC features, Chroma features, spectral centroid, spectral roll-off, ZCR are used as the feature vectors and trained the system using classifiers like k-NN, Logistic Regression, Random Forest and Support Vector Machine.

The below table represent the accuracy before and after tunning model.

Models	**Accuracy**	**Accuracy after tunning**
K-Nearest Neighbor (K-NN)	**62.50%**	**64.75%**
Logistic Regression (LR)	**64.23%**	**65.13%**
Random Forest (RF)	**64.45%**	**75.54%**
Support Vector Machines (SVM)	**65.50%**	**73%**

Above table shows that Random Forest classifier giving accuracy of 75.54% which is highest compare to other model used so Random Forest Classifier is used for this system.

CHAPTER 9

REFERENCES

[1] Michaël Defferrard, Kirell Benzi , Pierre Vandergheynst , Xavier Bresson. FMA: A Dataset For Music Analysis. Sound; Information Retrieval. arXiv:1612.01840v3, 2017.

[2] Tom LH Li, Antoni B Chan, and A Chun. Automatic musical pattern feature extraction using convolutional neural network. In Proc. Int. Conf. Data Mining and Applications, 2010.

[3] Hareesh Bahuleyan, Music Genre Classification using Machine Learning Techniques, University of Waterloo, 2018

[4] Loris Nanni, Yandre MG Costa, Alessandra Lumini,Moo Young Kim, and Seung Ryul Baek. Combining visual and acoustic features for music genre classification. Expert Systems with Applications 45:108–117, 2016.

[5] Thomas Lidy and Alexander Schindler. Parallel convolutional neural networks for music genre and mood classification. MIREX2016, 2016.

[6] Chathuranga, Y. M. ., & Jayaratne, K. L. Automatic Music Genre Classification of Audio Signals with Machine Learning Approaches. GSTF International Journal of Computing, 3(2), 2013.

[7] Fu, Z., Lu, G., Ting, K. M., & Zhang, D. A survey of audio based music classification and annotation. IEEE Transactions on Multimedia, 13(2), 303–319, 2011.

[8] Liang, D., Gu, H., & Connor, B. O. Music Genre Classification with the Million Song Dataset 15-826 Final Report, 2011.

[9] G Tzanetakis and P Cook. Musical genre classification of audio signals. IEEE Trans. on Speech and Audio Processing, 2002.

[10] D PW Ellis. Classifying music audio with timbral and chroma features. In ISMIR, 2007.