# GUJARAT TECHNOLOGICAL UNIVERSITY

CHANDKHEDA, AHMEDABAD
AFFILIATED

# G H PATEL COLLEGE OF ENGINERING & TECHNOLOGY

A
PROJECT REPORT
ON

MUSIC GENRE CLASSIFICATION

# UDP

UNDER SUBJECT OF
**PROJECT-II (2180706)**
B. E SEMESTER – VIII
(COMPUTER ENGINEERING BRANCH)


SUBMITTED BY:
GROUP:

| SR. | NAME OF STUDENT | ENROLLMENT NO. |
|---|---|---|
| 1) | JAY DAVE | 150110107013 |
| 2) | TRUSHIT PATEL | 150110107041 |

PROF. KHYATI MEHTA
FACULTY GUIDE

DR. MAULIKA S. PATEL
HEAD OF THE DEPARTMENT


ACADEMIC YEAR
(2018-2019)

# INDEX

# CERTIFICATE

This is to certify that the Project work entitled "**Music Genre Classification**" has been carried out by **Jay Dave (**Enrollment No.:**150110107013)** and **Trushit Patel (**Enrollment No.: **150110107041)** under my guidance and supervision for the partial fulfillment of subject PROJECT-I in Computer Engineering (Semester - VIII) at G H Patel College of Engineering & Technology, Vallabh Vidyanagar during the academic year **2018-19**.

**Date: -**

**Guide:**

Mrs. KHYATI MEHTA
DEPT COMPUTER ENGINEERING
GCET

**Head of Department:**

DR. MAULIKA S PATEL
DEPT COMPUTER ENGINEERING
GCET

# ACKNOWLEDGEMENTS

We take this opportunity to express our profound gratitude and deep regards to our project mentor **Mrs. Khyati Mehta** for her exemplary guidance, monitoring and constant encouragement throughout the course of this thesis.

We would also like to express our deep sense of gratitude to **Dr. Maulika Patel,** the Head of Department, GCET Computer Engineering, for her cordial support, valuable information and guidance which helped us a lot in completing the task of project development through various stages.

We are also obliged to the **faculty members** of the Computer Department, for the valuable information provided by them on various occasions in their respective fields. We are grateful for their cooperation during the period of our assignment.

We also thank our family members and friends for their constant encouragement without which this project would not have been completed.

Lastly, we thank **Gujarat Technological University** for providing such an opportunity that enables the students to test and expand their levels of creativity and programming skills through the medium of project development.

# 1. INTRODUCTION

## 1.1 Problem Summary

Many companies nowadays use music classification, either to be able to place recommendations to their customers (such as Spotify, Sound cloud), or simply as a product (for example Shazam). Determining specific music genres is a first step towards this goal.

Wikipedia states -that "music genre is a conventional category that identifies pieces of music as belonging to a shared tradition or set of conventions".

The term "genre" is a subject to interpretation and it is often the case that genres may very fuzzy in their definition. Further, genres do not always have sound music theoretic foundations, e.g. - Indian genres are geographically defined, Baroque is classical music genre based on time period.

Defining music genre from classification algorithms are most widely used method it is been state of art from last 2-3 years.

It is fairly simple for a human being to identify the genre of a song. One thinks about how fast the beat of the song is, the mood the song, the video of the song, etc. All these help create a mental picture of the song and thus the genres associated with it are determined. Automatic genre classification can be useful to solve some very interesting problems such as making song recommendations, finding similar songs, finding people who will like that particular song. Intelligence and automation are the core ideas that drove us to making this system.

## 1.2 Aim and Objective

**Aim:** Making an automated system that can identify music genre using machine learning methods.

**Objective:** For making identify the music genre we are going to use GTZAN dataset then we are going to extract features out of each song. There are 1000 songs available in this dataset. We are going to use different classification algorithms for identifying genres properly. Many algorithms will be used to check them against accuracy of each.

## 1.3 Problem Specifications

Traditional approach of identifying genre is based on appending information regarding genre. It becomes headache if we have a big database containing songs without label specifying genre. Then we have to listen music and decide which genre it belongs to. We also need an expert to do so. Genre classification, till now, had been done manually by appending it to metadata of audio files or including it in album info. This project however aims at content-based classification, focusing on information within the audio rather than extraneously appended information The method that we are going to use is traditional approach of machine learning find the suitable features of data, train classifier on feature data, make predictions.

Traditional approach of identifying genre is taking help of musician or singer or experts in music world. Each time we need to find genre of a song, what we have to do is, consult an expert for it. That becomes hectic.

Now, another problem is that identifying genre of music is based on certain features that are only known by an expert like texture, tonality, music meter, timbre | tone colour | tone quality, dynamics, articulation, tempo. This features are for humans to understand and get some inferences form them but for an auto mated system, there should be concrete realization of above listed features. Signal analysis is performed so as to find concrete features like MFCC, LPC, Zero crossing rate, Chroma features and many more.

Now, Pre-processing is applied on these feature vectors so as to get the ready-to-process dataset. Pre-processing includes data mining methodologies like correlation analysis, normalization, smoothing out errors, noise and other unwanted.

After pre-processing step, we train ML model for prediction task. There are many models like linear regression, logistic regression, SVM, k-nearest neighbour, neural network and other unsupervised learning algorithm. Python libraries are used to get pre-implemented and optimized codes for the same and are used. All of them are applied to check efficiency and accuracy of each, that helps in finding the best model.

## 1.4 Literature reviews

### 1. Unsupervised classification of music genre using hidden markov model: [1]

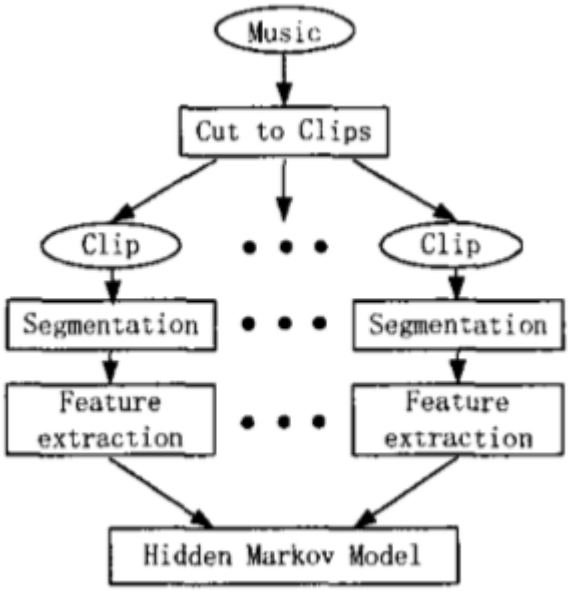
In order to discriminate music genre properly. They consider segmenting the music clip according to its intrinsic rhythmic structure.

**Features Used:**

- *MFCC-male frequency ceptral coefficient*
- *LPC- linear prediction coefficient*
- Delta values
- Acceleration values

**Algorithms Used:**

- Unsupervised algorithm-hidden markov
- labour-intensive
- error prone



## 2. Automatic Music Genre Classification of audio signals with Machine Learning approach [2]:

**Features Used:**

- *High Level:*
  - *Mood & Instruments*
- Low Level:
  - MFCC-male frequency ceptral coefficient| STFT- Timbre
  - ZCR | power spectrum | Amplitude- Temporal
  - RP | RH | SSD- Rhythm

**Algorithms Used:**

- K-NN
- Linear Kernel SVM
- Poly Kernel SVM
- Decision Tree
- Random Forest
- Logistic Regression

**Dataset Used:**

- GTZAN

## 3. Automatic Music Genre Classification by low rank semantic mapping [3]:

**Features Used:**

- MFCC-male frequency ceptral coefficient| STFT- Timbre| RP | RH | SSD- Rhythm
- Auditory cortical representation extraction- mimicking cochlea
- Chroma features - harmonic content
- low rank semantic mapping

**Algorithms Used:**

- LRSM | SVM | SRC | NN

**Dataset Used:**

- GTZAN | ISMIR | Homburg | 1517-Artists

## 4. Automatic music genre classification based on musical instrument track separation [4]:

The aim of this article is to investigate whether separating music track sat the preprocessing phase and extending feature vector by parameters related to the specific musical instruments

**Features Used:**

- *MFCC-male frequency ceptral coefficient| STFT- Timbr| RP | RH | SSD- Rhythm*
- *Chroma features - harmonic content*

**Algorithms Used:**

- LRSM | SVM | SRC | NN

**Dataset Used:**

- GTZAN | ISMIR | Homburg | 1517-Artists

## 5. Music Genre Classification by Indian Institute of Technology, Kanpur [5]:

**Features Used:**

- *MFCC-male frequency ceptral coefficient*
- *Chroma features - harmonic content*
- *Spectral Centroid*
- *Zero Crossing Rate*

**Algorithms Used:**

- K-NN
- Linear Kernel SVM
- Poly Kernel SVM
- Decision Tree
- Random Forest
- Logistic Regression

**Dataset Used:**

- GTZAN

## 1.5 Plan of Work:

**August**

- Week 1 : Referential Reading of research papers and study of technology used
- Week 2 : Dataset collection from different sources online
- Week 3 : Study of various classification and feature extraction algorithms
- Week 4 : Reading research paper on Music genre classification using signal processing and machine learning

**September**

- Week 1 : Integration of the information from different sources
- Week 2 to 4 : Documentation of all the study, generating project report

**January**

- Week 1 & 2: We eftracted 3eatures o3 the audio file using librosa library.
- Week 3 & 4 :visualized the data.

**February**

- Week 1 &2 : WE tried to reduce features by PCA and Normali3ation and discreti3ation. We applied Linear and Logistic regression and Neural network algorithm.
- Week 3& 4 : We applied SVM using sklearn library on 627 feature columns and got 63% accuracy and then on 194 features and got higher accuracy

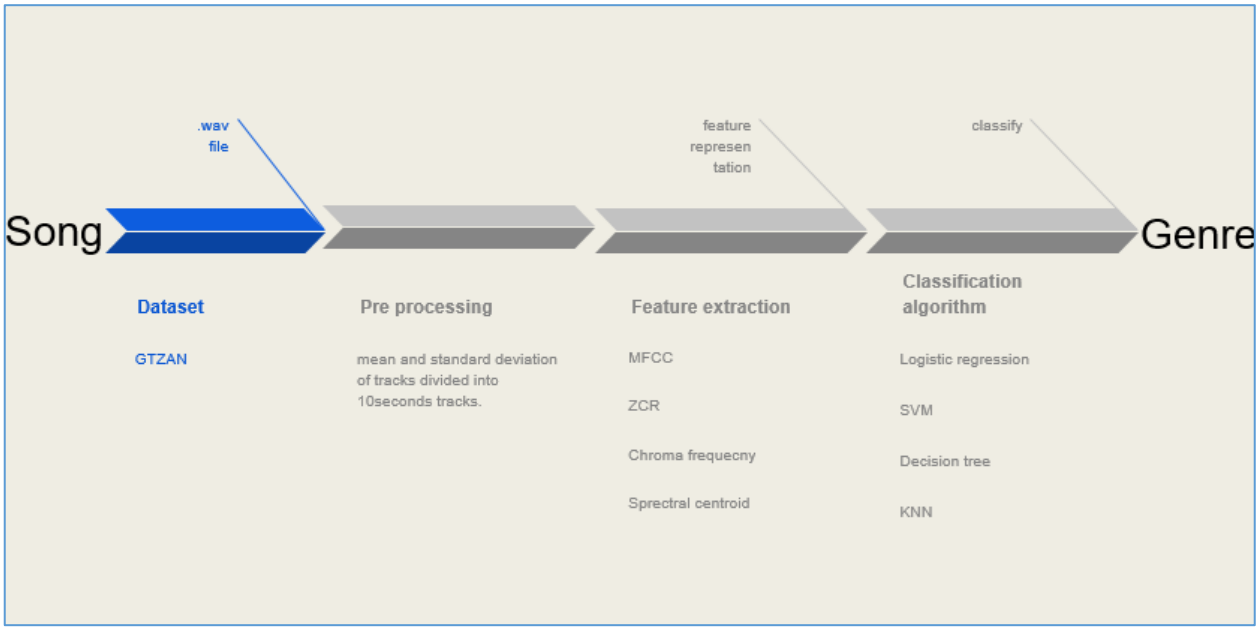
**March**

- Week 1 & 2: We developed a website for the user interface to access the facility of Genre prediction.
- Week 3 & 4:We tried to optimize algorithm.

**April**

- Documentation of all the study, generating project report

## 1.6 System Flow:




**Dataset:**

We will use the GTZAN dataset from the MARYSAS [6] website. It contains 9 music genres; each genre has 100 audio clips in .au format. The genres are - blues, classical, country, disco, pop, jazz, reggae, rock, metal. Each audio clips have a length 30 seconds, are 22050Hz Mono 16-bit. files. The dataset incorporates samples from variety of sources like CDs, radios, microphone recordings, etc.

**Pre-Processing:**

We will convert us .mp3 files to .wav file and we can also use some standard scalar like min-max scaler, data transformation function if the classification algorithms accuracy of algorithm is not well.

**Feature Extraction:**

- **Spectral Centroid:**

  It describes where the ”center of mass” for sound is. It essentially is the weighted mean of the frequencies presents in the sound. Consider two songs, one from blues and one from metal. A blues song is generally consistent throughout it length while a metal song usually has more frequencies accumulated towards the end part. So

spectral centroid for blues song will lie somewhere near the middle of its spectrum while that for a metal song would usually be towards its end.
spectral_centroid = librosa.feature.spectral_centroid(y=signal, sr=sample_rate, n_fft=frame_size, hop_length=hop_size)

- **MFCC:**

  Mel Frequency Cepstral Coefficients (MFCCs) [7] are a feature widely used in automatic speech and speaker recognition. They were introduced by Davis and Mermelstein in the 1980's, and have been state-of-the-art ever since.
  It generally represents energy that is available in each frame.
  It gives us 13 co-efficient.

- **Zero crossing rate:**

  It represents the number of times the waveform crosses 0.
  The zero-crossing rate is the rate of sign-changes along a signal, i.e., the rate at which the signal changes from positive to negative or back.
  zero_crossing_rate=librosa.feature.zero_crossing_rate(y=signal, frame_length=frame_size, hop_length=hop_size)

- **Chroma frequencies:**

  Chroma-based features, which are also referred to pitch class profiles, are a powerful tool for analyzing music whose pitches can be meaningfully categorized (often into twelve categories.

**Classification Algorithms:**

- **Logistic Regression:**
  It is one of the algorithms used in classification algorithms and has a different linear, polynomial function that we can use.

- **Support Vector Machine:**
  This type of algorithms can be implemented in scikit learn library. SVM is optimized classification algorithm and has a very good tuning parameters in scikit library like gamma, margin and c.

- **Decision Tree:**
  In this algorithm we divide the dataset features repeatedly using information gain and entropy.

- **K nearestest neighbor:**
  It is algorithm that uses Euclidian distance between data points and based on k value it determines the class label.

## 1.7 Materials and Tools used:

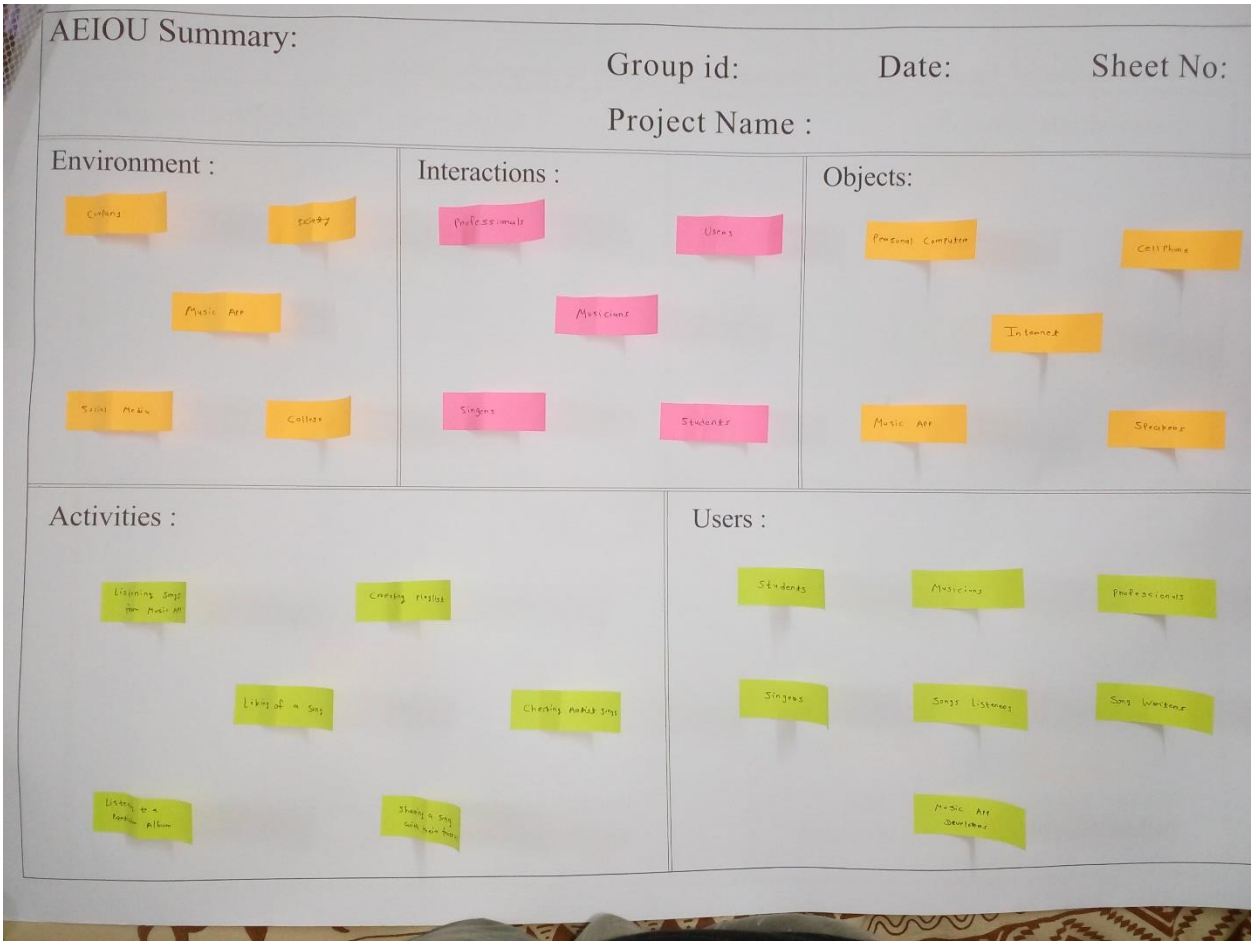
We are going to build automated system using machine learning and we found that Python can be the best language for implementation.

**Tools and Libraries:**
In-according to do that we will going to use various python libraries Pandas, SK Learn, Numpy, Librosa etc. Python language editors available are jupyter notebook, spyder, jetBrains. We are planning to use jupyter notebook.

**Materials:**
We also read various research papers and articles, online tutorials on machine learning.

# 2. Design Engineering-Canvas Activity

## 2.1 AEIOU Summary Canvas




AEIOU Framework. AEIOU is a heuristic to help interpret observations gathered by ethnographic practice in industry. Its two primary functions are to code data, and to develop building blocks of models that will ultimately address the objectives and issues of a client.

1. **Activities that have to be performed are as follows**
    1. Train a model that classifies the given song based on its genre.
    2. Select a song.
    3. Pre-process selected song.
    4. Apply model to classify song.
    5. Find genre of given song.

2. **Environment that our product resides are as follows.**
    1. College
    2. Social media

3. Mobile app
4. Web app

**3. Interactions**

1. Students
2. Musicians
3. Professionals
4. Users
5. Singers
6. Experts

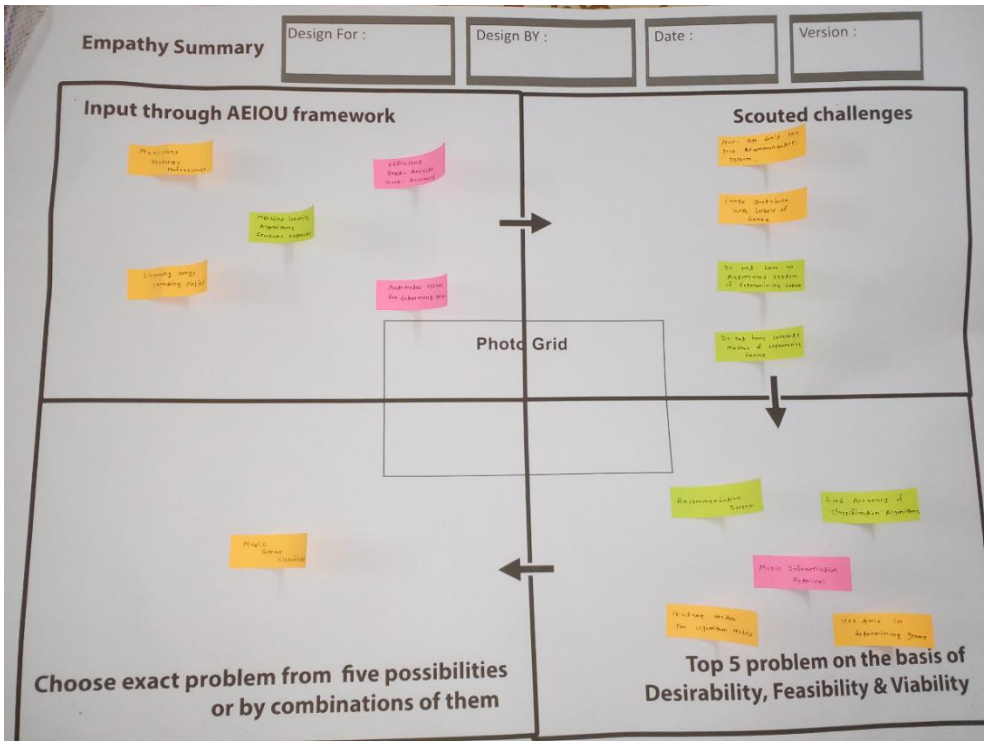
**4. Objects**

1. Music app
2. Mobile phone
3. Computer
4. Internet
5. Songs

**5. Users**

1. Normal users
2. Musicians
3. Professionals
4. Students
5. Singers
6. Websites of music content provider

## 2.2 Empathy Summary Canvas

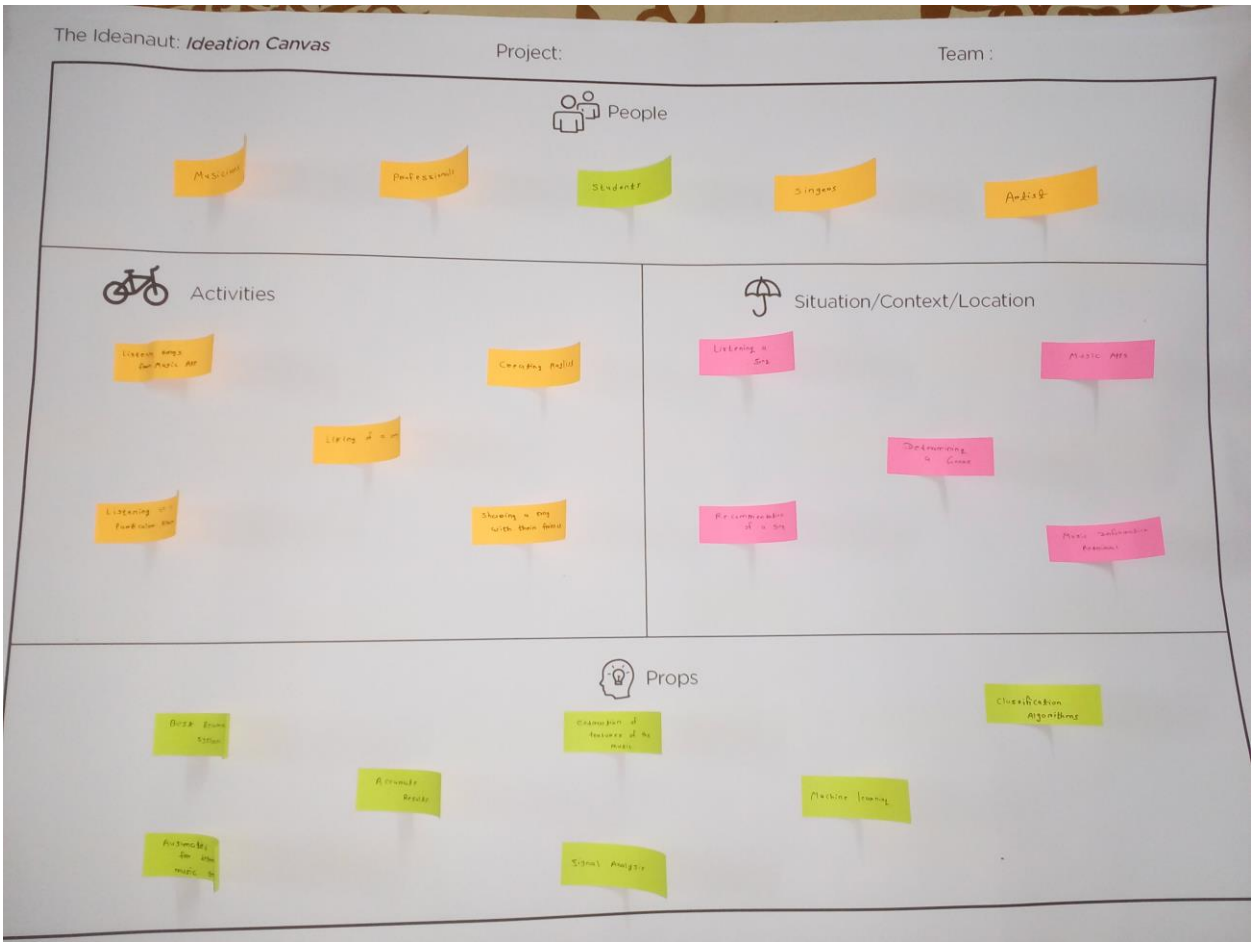



**Input through AEIOU framework**

1. **Scouted challenges**
    1. Music apps don't have good recommendation system
    2. Large dataset with labels of genre
    3. No automated system to classify
    4. No concrete methods to determine genre

2. **Top 5 problems on the basis of Desirability, Feasibility & Viability**
    1. Music information retrieval
    2. Less time in finding genre
    3. Good accuracy
    4. Recommendation system
    5. Feature vectors for algorithm
3. **Choose exact problem from five possibilities or by combinations of them**
    1. Music genre classification

## 2.3 Ideation Canvas




1. **People**
    1. Normal users
    2. Musicians
    3. Professionals
    4. Students
    5. Singers
    6. Websites of music content provider

2. **Activities**
    1. Train a model that classifies the given song based on its genre. (Pre requisite)
    2. Select a song.
    3. Pre-process selected song.
    4. Apply model to classify song.
    5. Find genre of given song.

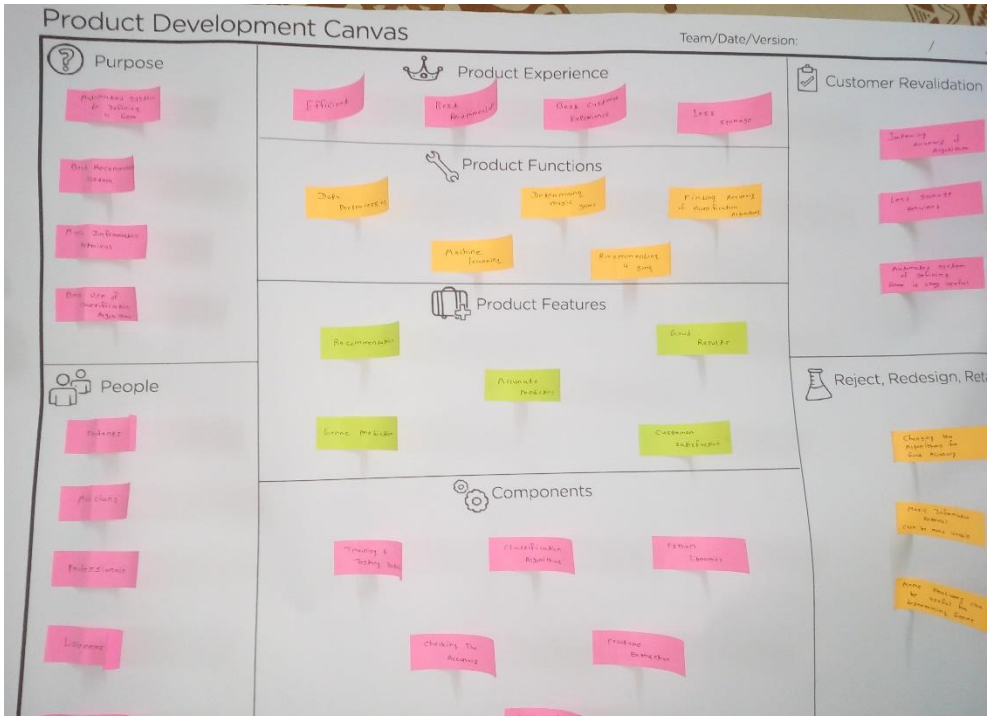
3. **Situation**
    1. Listening to a song
    2. Determining genre of song

3. Getting recommendation of song
4. Music information retrieval

**4. Props**

1. Signal analysis
2. Machine learning
3. Extraction of features
4. Concrete features from raw features

## 2.4 Product Development Canvas




1. **Purpose**
    1. To automate the system of classifying
    2. Best recommendation system development
    3. Music information retrieval
    4. Ml to remove the need of experts

2. **People**
    1. Normal users
    2. Musicians
    3. Professionals
    4. Students
    5. Singers
    6. Websites of music content provider

3. **Product Experience**
    1. Best Recommendation system
    2. No need to search online
    3. Efficient
    4. Reduces efforts

4. **Product Function**
    1. Data gathering
    2. Pre-processing

3. Training a model
4. Applying model
5. Classifying
6. Applying Data-mining methodology

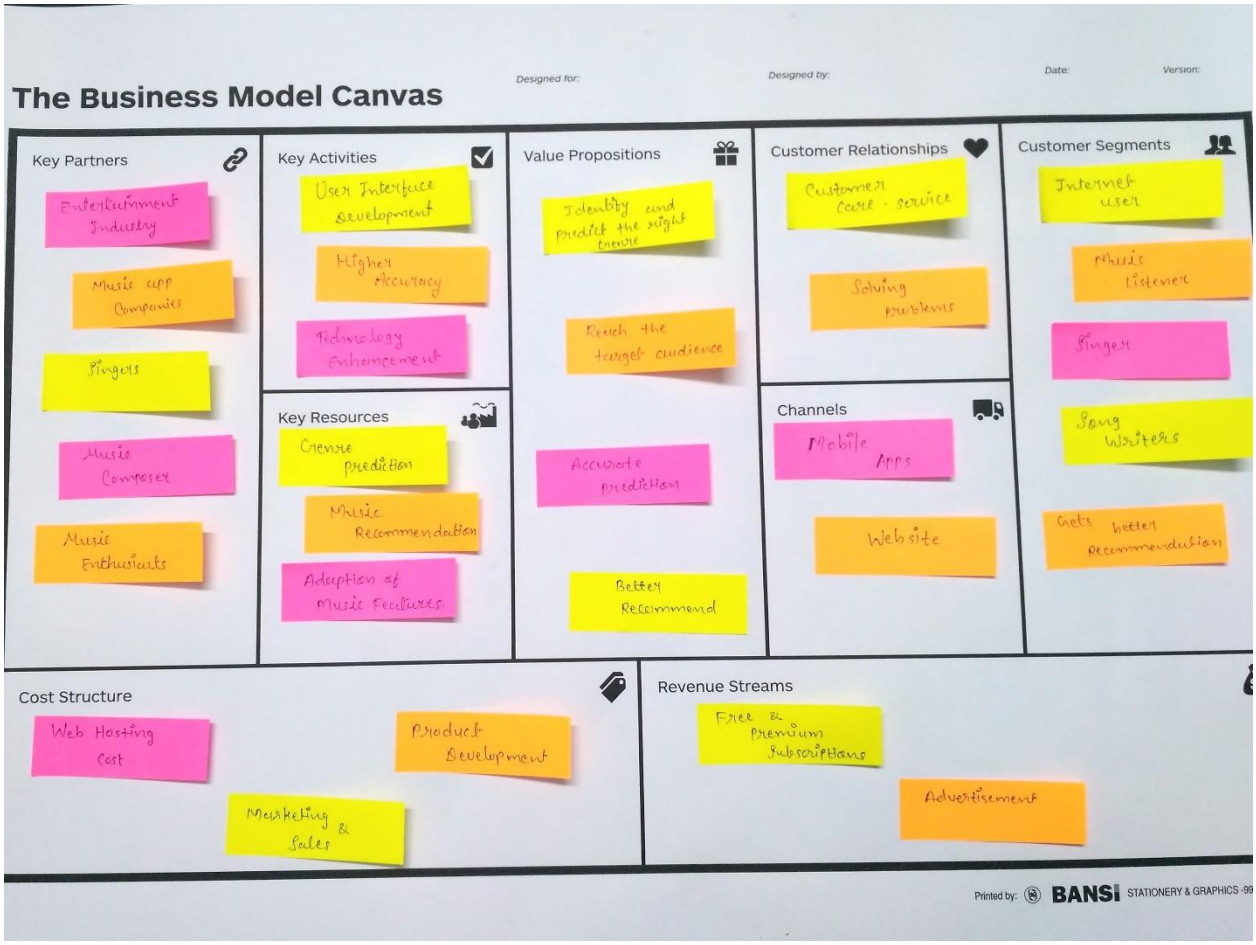
**5. Product Feature**

1. Classifying genre of music
2. Recommendation
3. Accurate prediction
4. Good result
5. Customer satisfaction

**6. Components**

1. Taring | Cross validation | Testing dataset
2. Classification algorithm
3. Python libraries
4. Feature extraction
5. Checking accuracy
6. Visualization

## 2.5 Business Model Canvas




1. **KEY PARTNERS**
    1. Entertainment Industry
    2. Music Apps Companies
    3. Singers
    4. Music Enthusiastic
    5. Music Composers

2. **KEY ACTIVITIES**
    1. Platform Development
    2. Technology Enhancement
    3. Features Extraction from songs
    4. Problem solving
    5. Increase Accuracy

3. **KEY RESOURCES**
    1. Features of different genre
    2. Dataset Required for prediction
    3. Music Website & App

4. **VALUE PROPOSITIONS**
    1. Identify and Reach the right genre of song.
    2. Automatic Genre prediction
    3. Reach the target audience.

5. **CUSTOMER RELATIONSHIP**
    1. Same side network effects.
    2. If more songs will be added the good and accurate model can be and better experience for customers.
    3. A prediction can work efficiently so, indirectly it is useful and work full.

6. **CHANNELS**
    1. We made a website and mobile app can be added as a channel.

7. **COSTOMERS SEGMENTS**
    1. Internet Users
    2. Music Listeners
    3. Singers
    4. Song Writer
    5. People

8. **COST STRUCTURE**
    1. Web Hosting Costs
    2. Marketing & Sales
    3. Product Development

9. **REVENUE STREAMS**
    1. Free offering & premium subscriptions
    2. Advertising

# 3. Implementation steps

1. Pre-process the data set/ songs
    a. Format Normalization (any format of song to .wav format)

```
def main():
    print("Creating new folder...")
    try:
        os.mkdir(OUTPUT_DIR)
        print("Converting files to wav format...")
        for eachfile in os.listdir(INPUT_DIR):
            song = AudioSegment.from_file(INPUT_DIR + "/" + eachfile, FILE_TYPE)
            song.export(OUTPUT_DIR + eachfile[:-3] + "_wav.wav", format="wav")
```

2. Feature extraction using librosa- the ML python library
    - Feature extraction was most important part of our project.
    - Here we included snapshots and brief introduction of the main features.

```
#genres = 'alternative blues electronic folkcountry funksoulrnb jazz pop raphiphop
genres = 'classical blues country disco hiphop jazz pop metal reggae rock'.split()
row=0
for g in genres:
    for filename in os.listdir(f'/path/{g}'):

        row+=1 #for dataframe row entry

        songname = f'/path/{g}/{filename}'

        y, sr = librosa.load(songname, mono=True, duration=30)

        #features
        rmse=librosa.feature.rmse(y=y)
        chroma_stft = librosa.feature.chroma_stft(y=y, sr=sr)
        spec_cent = librosa.feature.spectral_centroid(y=y, sr=sr)
        spec_bw = librosa.feature.spectral_bandwidth(y=y, sr=sr)
        rolloff = librosa.feature.spectral_rolloff(y=y, sr=sr)
        zcr = librosa.feature.zero_crossing_rate(y)
        mfcc = librosa.feature.mfcc(y=y, sr=sr)
        chroma_cqt = librosa.feature.chroma_cqt(y=y, sr=sr)
        chroma_cens = librosa.feature.chroma_cens(y=y, sr=sr)
        melspectrogram = librosa.feature.melspectrogram(y=y, sr=sr)

        S = np.abs(librosa.stft(y))
        contrast = librosa.feature.spectral_contrast(S=S, sr=sr)
        flatness = librosa.feature.spectral_flatness(y=y)
```

1. **MFCC:**
   - The main point to understand about speech is that the sounds generated by a human are filtered by the shape of the vocal tract including tongue, teeth etc. This shape determines what sound comes out. If we can determine the shape accurately, this should give us an accurate representation of the phoneme being produced. The shape of the vocal tract manifests itself in the envelope of the short time power spectrum, and the job of MFCCs is to accurately represent this envelope.
   - We will now go a little more slowly through the steps and explain why each of the steps is necessary.
   - An audio signal is constantly changing, so to simplify things we assume that on short time scales the audio signal doesn't change much (when we say it doesn't change, we mean statistically i.e. statistically stationary, obviously the samples are constantly changing on even short time scales). This is why we frame the signal into 20-40ms frames. If the frame is much shorter we don't have enough samples to get a reliable spectral estimate, if it is longer the signal changes too much throughout the frame.
   - The next step is to calculate the power spectrum of each frame. This is motivated by the human cochlea (an organ in the ear) which vibrates at different spots depending on the frequency of the incoming sounds. Depending on the location in the cochlea that vibrates (which wobbles small hairs), different nerves fire informing the brain that certain frequencies are present. Our periodogram estimate performs a similar job for us, identifying which frequencies are present in the frame.
   - The periodogram spectral estimate still contains a lot of information not required for Automatic Speech Recognition (ASR). In particular the cochlea can not discern the difference between two closely spaced frequencies. This effect becomes more pronounced as the frequencies increase. For this reason we take clumps of periodogram bins and sum them up to get an idea of how much energy exists in various frequency regions. This is performed by our Mel filterbank: the first filter is very narrow and gives an indication of how much energy exists near 0 Hertz. As the frequencies get higher our filters get wider as we become less concerned about variations. We are only interested in roughly how much energy occurs at each spot. The Mel scale tells us exactly how to space our filterbanks and how wide to make them. See below for how to calculate the spacing.
   - Once we have the filterbank energies, we take the logarithm of them. This is also motivated by human hearing: we don't hear loudness on a linear scale. Generally to double the percieved volume of a sound we need to put 8 times as much energy into it. This means that large variations in energy may not sound all that different if the sound is loud to begin with. This compression operation makes our features match more closely what humans actually hear. Why the logarithm and not a cube root? The logarithm allows us to use cepstral mean subtraction, which is a channel normalisation technique.
   - The final step is to compute the DCT of the log filterbank energies. There are 2 main reasons this is performed. Because our filterbanks are all overlapping, the filterbank energies are quite correlated with each other. The DCT decorrelates the energies

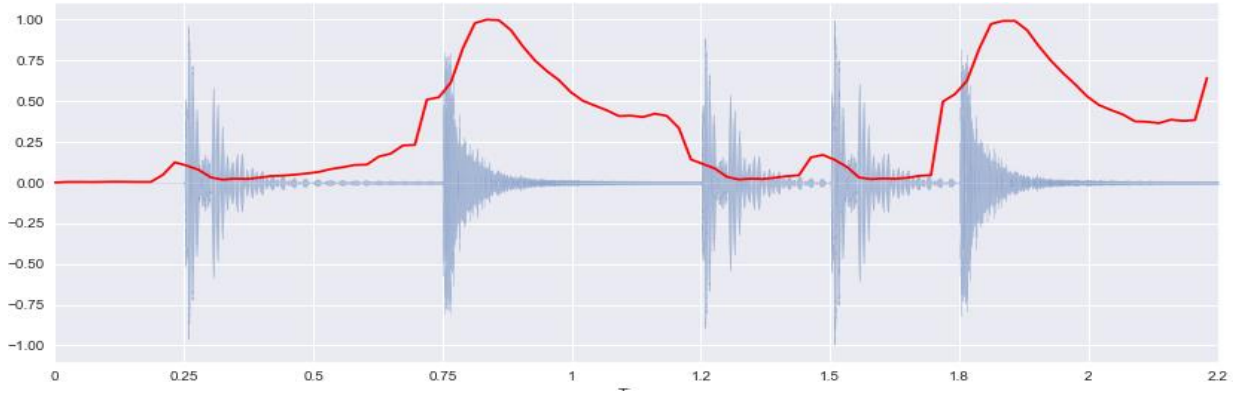
which means diagonal covariance matrices can be used to model the features in e.g. a HMM classifier. But notice that only 12 of the 26 DCT coefficients are kept. This is because the higher DCT coefficients represent fast changes in the filterbank energies and it turns out that these fast changes actually degrade ASR performance, so we get a small improvement by dropping them.

2. **Zero Crossing Rate:**
   - The zero-crossing rate is the rate of sign-changes along a signal, i.e., the rate at which the signal changes.
   - from positive to zero to negative or from negative to zero to positive.
   - This feature has been used heavily in both speech recognition and music information retrieval, being a key feature to classify percussive sounds.

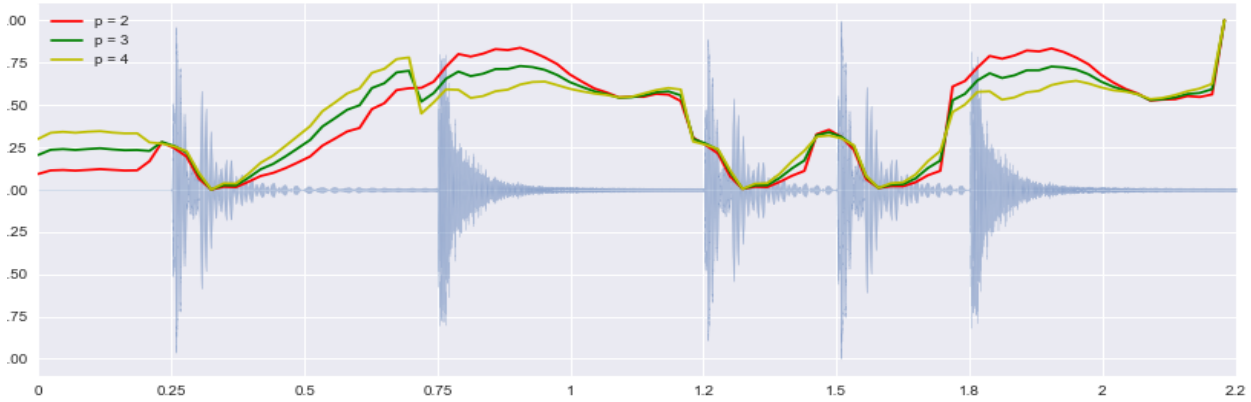
3. **Chroma Frequencies:**
   - In music, the term chroma feature or chromagram closely relates to the twelve different pitch classes. Chroma-based features, which are also referred to as pitch class profiles, are a powerful tool for analyzing music whose pitches can be meaningfully categorized (often into twelve categories) and whose tuning approximates to the equal-tempered scale. One main property of chroma features is that they capture harmonic and melodic characteristics of music, while being robust to changes in timbre and instrumentation.

4. **Roll Off:**
   - Spectral Rolloff Point. A feature extractor that extracts the Spectral Rolloff Point. This is a measure measure of the amount of the right-skewedness of the power spectrum. The spectral rolloff point is the fraction of bins in the power spectrum at which 85% of the power is at lower frequencies.
   -

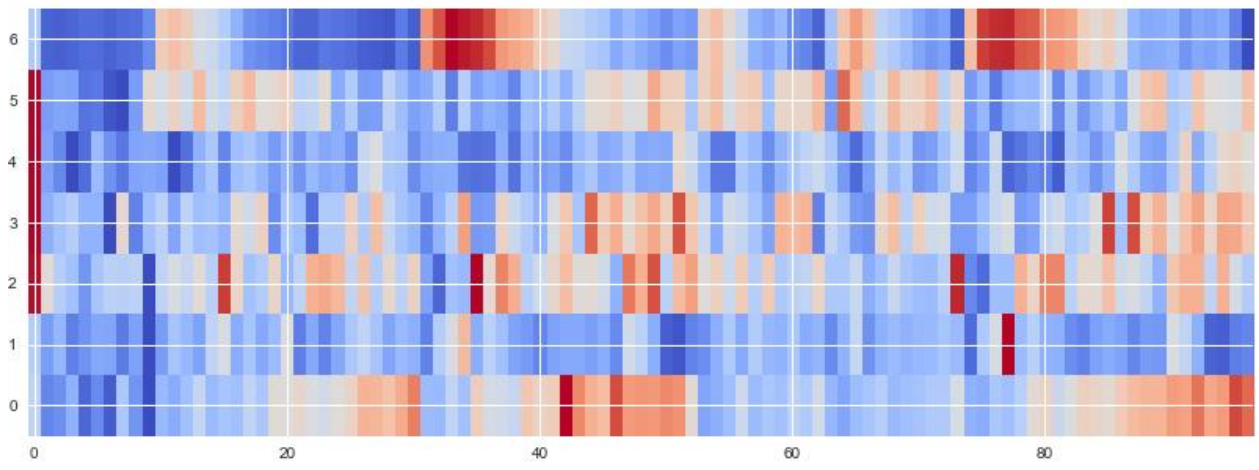
5. **Spectral Centroid:**
   - Spectral centroid is a measure used in digital signal processing to characterise a spectrum. It indicates where the "center of mass" of the spectrum is located. Perceptually, it has a robust connection with the impression of "brightness" of a sound.



6. **Spectral BandWidth:**
    - Spectral Bandwidth. It is the Wavelength interval in which a radiated spectral quantity is not less than half its maximum value. It is a measure of the extent of the spectrum For a Light Source typical spectral widths are 20 to 60 nm for a LED and 2 to 5 nm for a Laser diode.




7. **Spectral Contrast:**
    - Spectral Bandwidth. It is the Wavelength interval in which a radiated spectral quantity is not less than half its maximum value. It is a measure of the extent of the spectrum For a Light Source typical spectral widths are 20 to 60 nm for a LED and 2 to 5 nm for a Laser diode.



- Now we have a dataset….

| | filename | rmse | chroma_stft | rolloff | mfcc1 | mfcc2 | mfcc3 | mfcc4 |
|---|---|---|---|---|---|---|---|---|
| 0 | classical.00027.au | 0.028419 | 0.240705 | 1456.666192 | -366.220203 | 181.600639 | -5.524955 | 27.001510 |
| 1 | classical.00037.au | 0.041551 | 0.284173 | 2838.224504 | -275.346266 | 125.311286 | -41.043671 | 24.590540 |
| 2 | classical.00038.au | 0.044319 | 0.294410 | 2376.660629 | -276.043356 | 146.371620 | -36.571275 | 24.187743 |
| 3 | classical.00030.au | 0.029736 | 0.278349 | 2353.127438 | -315.649646 | 149.671703 | -30.598152 | 23.223976 |
| 4 | classical.00096.au | 0.057000 | 0.293142 | 3573.061728 | -203.587158 | 111.198330 | -51.526940 | 39.475952 |

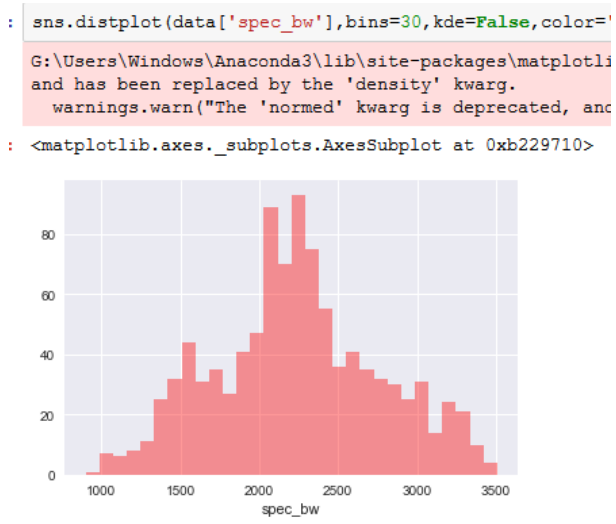
5 rows × 83 columns

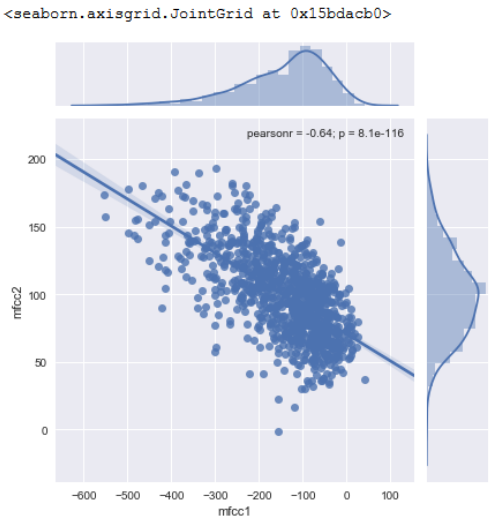
```
data.shape
```

```
(1000, 83)
```

3. Data visualization
    a. Seaborn –python library
    b. Matplotlib

- Thease are some visualization snapshots of the dataset.

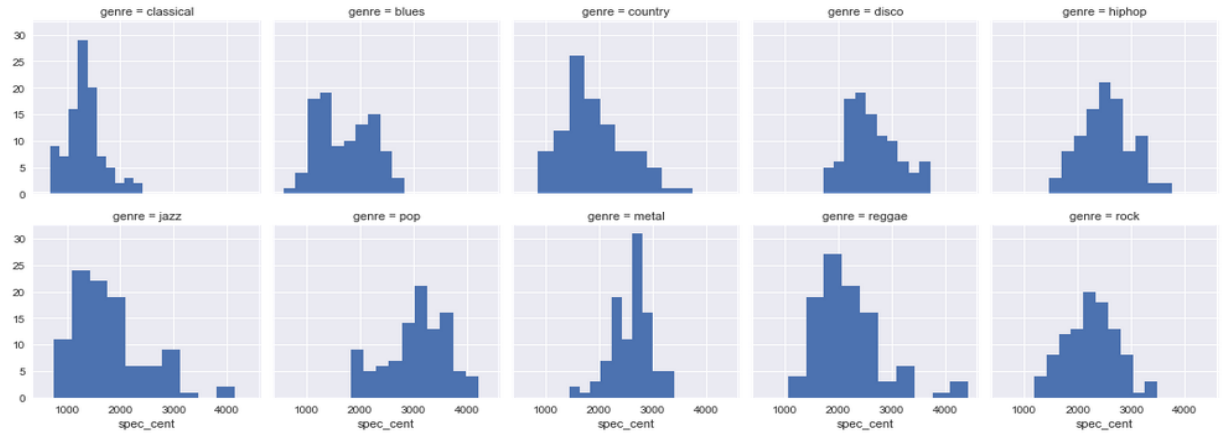



[Spectral Bandwidth]

```
g = sns.FacetGrid(data=data,col='genre',col_wrap=5)
g.map(plt.hist,'spec_cent')
```

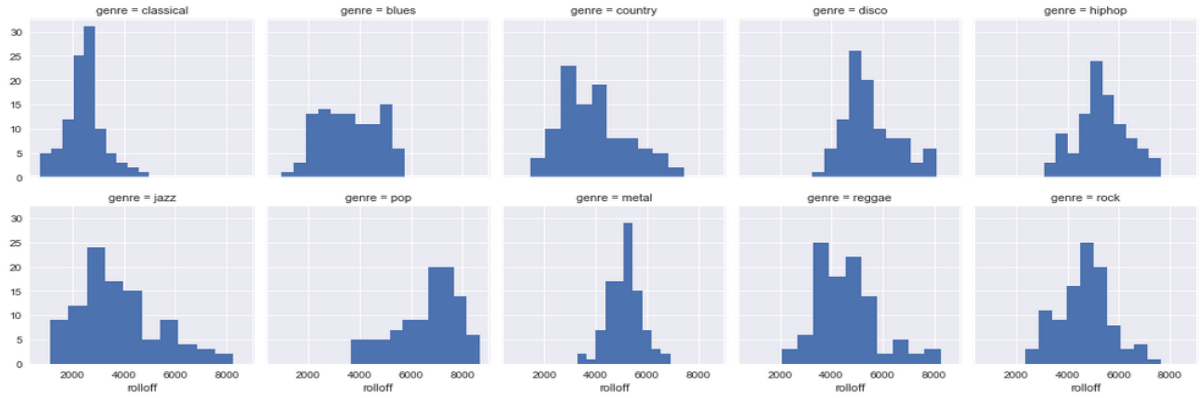
```
<seaborn.axisgrid.FacetGrid at 0x13ef7790>
```




[Spectral Centroid]

```
g = sns.FacetGrid(data=data,col='genre',col_wrap=5)
g.map(plt.hist,'rolloff')
```

```
<seaborn.axisgrid.FacetGrid at 0x14a026b0>
```



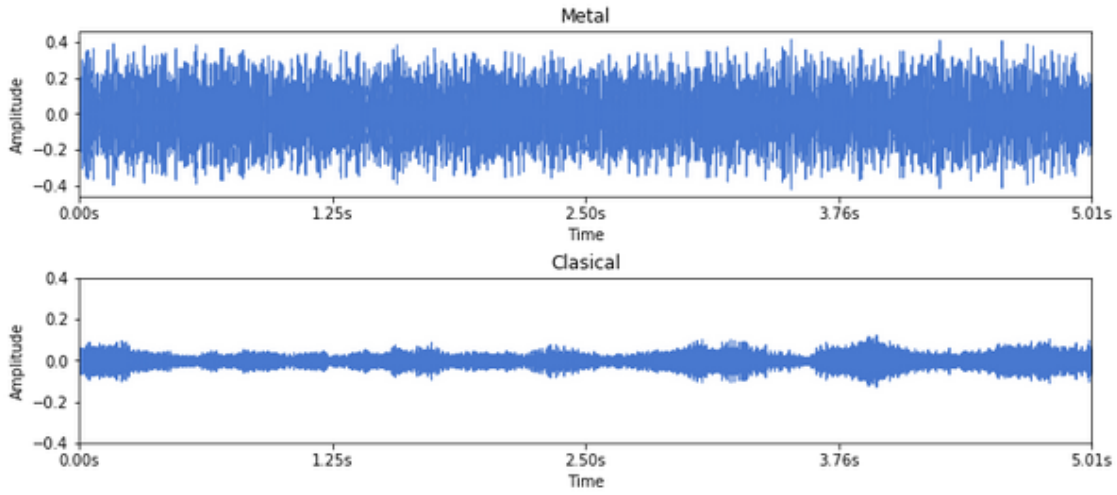

[Spectral Roll off]

```
fig=plt.figure(figsize=(12,5))
fig.subplots_adjust(hspace=.5)

plt.subplot(2, 1, 1)
plt.xlabel("Time")
plt.ylabel("Amplitude")
librosa.display.waveplot(x_metal,sampling_rate)
plt.title('Metal')

plt.subplot(2, 1, 2)
plt.ylim(-0.4,0.4)
plt.xlabel("Time")
plt.ylabel("Amplitude")
librosa.display.waveplot(x_classical,sampling_rate)
plt.title('Clasical')
```

```
<matplotlib.text.Text at 0x7f6679281b70>
```




[Time/ Amplitude graph]

4. Checking correlation to find disjoint set of features

$$r = \frac{\sum XY - \frac{\sum X \sum Y}{N}}{\sqrt{\left(\sum X^2 - \frac{(\sum X)^2}{N}\right)\left(\sum Y^2 - \frac{(\sum Y)^2}{N}\right)}}$$

Applying above formula on feature vectors to decide which features are correlated, this helps in selecting optimum set of features and algorithms like SVM.

5. Applying data mining methodology that is finding best polynomial features for linear or logistic models

In this step , we apply different combinations of polynomial features generated by applying power function to each features.

This helps in finding a better set of features that are combinations of different features.

6. Training model using dataset(training | cross validation | test data set)
    a. Dividing dataset into 3 sets
        i. Training data set
        ii. Cross validation set
        iii. Test data set

    This way of dividing dataset helps improve accuracy and checks whether model is not stuck in high-variance or high-bias

    b. ML models
        i. Linear regression
        ii. Logistic regression
        iii. SVM
        iv. KNN
7. Prediction
    a. Get a new song
    b. Pre-process it
    c. Use trained model to classify the new song
    d. Get output in terms of genre

- Snapshots of trained model and their classification reports:

## svm

```
20]: clf = svm.SVC(gamma=0.01, C=10)

21]: clf.fit(X_train,y_train)

21]: SVC(C=10, cache_size=200, class_weight=None, coef0=0.0,
       decision_function_shape=None, degree=3, gamma=0.01, kernel='rbf',
       max_iter=-1, probability=False, random_state=None, shrinking=True,
       tol=0.001, verbose=False)

22]: from sklearn.metrics import classification_report,confusion_matrix

23]: predictions=clf.predict(X_test)

24]: print(classification_report(y_test,predictions))
```

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| blues | 0.65 | 0.76 | 0.70 | 17 |
| classical | 0.71 | 0.89 | 0.79 | 19 |
| country | 0.67 | 0.70 | 0.68 | 20 |
| disco | 0.46 | 0.29 | 0.35 | 21 |
| hiphop | 0.64 | 0.62 | 0.63 | 26 |
| jazz | 0.75 | 0.69 | 0.72 | 26 |
| metal | 0.77 | 1.00 | 0.87 | 20 |
| pop | 0.77 | 0.62 | 0.69 | 16 |
| reggae | 0.65 | 0.62 | 0.63 | 21 |
| rock | 0.43 | 0.43 | 0.43 | 14 |
| avg / total | 0.66 | 0.67 | 0.65 | 200 |

- We applied gridsearch approach to find best parameters for svm.

```
from sklearn.model_selection import GridSearchCV
```

```
param_grid = {'C': [0.1,1, 10, 100], 'gamma': [1,0.1,0.01,0.001]}
```

```
grid = GridSearchCV(svm.SVC(),param_grid,refit=True,verbose=2)
grid.fit(X_train,y_train)
```

```
Fitting 3 folds for each of 16 candidates, totalling 48 fits
[CV] C=0.1, gamma=1 ..................................................
[CV] ..................................... C=0.1, gamma=1, total=   0.2s
[CV] C=0.1, gamma=1 ..................................................
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.3s remaining:    0.0s
[CV] ..................................... C=0.1, gamma=1, total=   0.2s
[CV] C=0.1, gamma=1 ..................................................
[CV] ..................................... C=0.1, gamma=1, total=   0.2s
[CV] C=0.1, gamma=0.1 ................................................
[CV] ................................. C=0.1, gamma=0.1, total=   0.1s
[CV] C=0.1, gamma=0.1 ................................................
[CV] ................................. C=0.1, gamma=0.1, total=   0.1s
[CV] C=0.1, gamma=0.1 ................................................
[CV] ................................. C=0.1, gamma=0.1, total=   0.1s
[CV] C=0.1, gamma=0.01 ...............................................
[CV] ................................ C=0.1, gamma=0.01, total=   0.1s
[CV] C=0.1, gamma=0.01 ...............................................
[CV] ................................ C=0.1, gamma=0.01, total=   0.1s
```

```
grid.best_params_
```

```
{'C': 10, 'gamma': 0.01}
```

[Grid Search]

- We got different accuracy for models..

## logistic regression ¶

```
from sklearn.linear_model import LogisticRegression
clf = LogisticRegression(random_state=0, solver='lbfgs',multi_class='multinomial')
clf.fit(X_train,y_train)
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='multinomial',
          n_jobs=1, penalty='l2', random_state=0, solver='lbfgs',
          tol=0.0001, verbose=0, warm_start=False)
```

```
output=clf.predict(X_test)
k=0
for i in range(len(output)):
    #print(i,genreDict[y_test[i]],'\t',genreDict[output[i]])
    if genreDict[y_test[i]]==genreDict[output[i]]:
        k+=1
print("Accuracy",k/len(output))
```

```
Accuracy 0.595
```

```
clf = LogisticRegression(random_state=0, solver='lbfgs',multi_class='multinomial')
scores = cross_val_score(clf,X,y, cv=100)
print(scores.mean())
```

```
0.6439999999999999
```

## Decision tree

```
clf = tree.DecisionTreeClassifier()
clf.fit(X_train,y_train)
```

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

```
output=clf.predict(X_test)
k=0
for i in range(len(output)):
    #print(i,genreDict[y_test[i]],'\t',genreDict[output[i]])
    if genreDict[y_test[i]]==genreDict[output[i]]:
        k+=1
print("Accuracy",k/len(output))
```

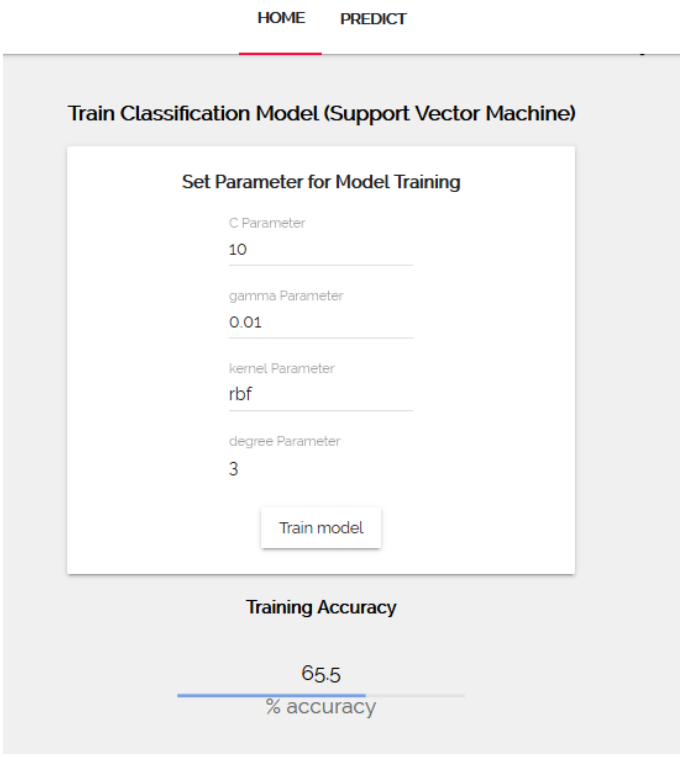
```
Accuracy 0.47
```

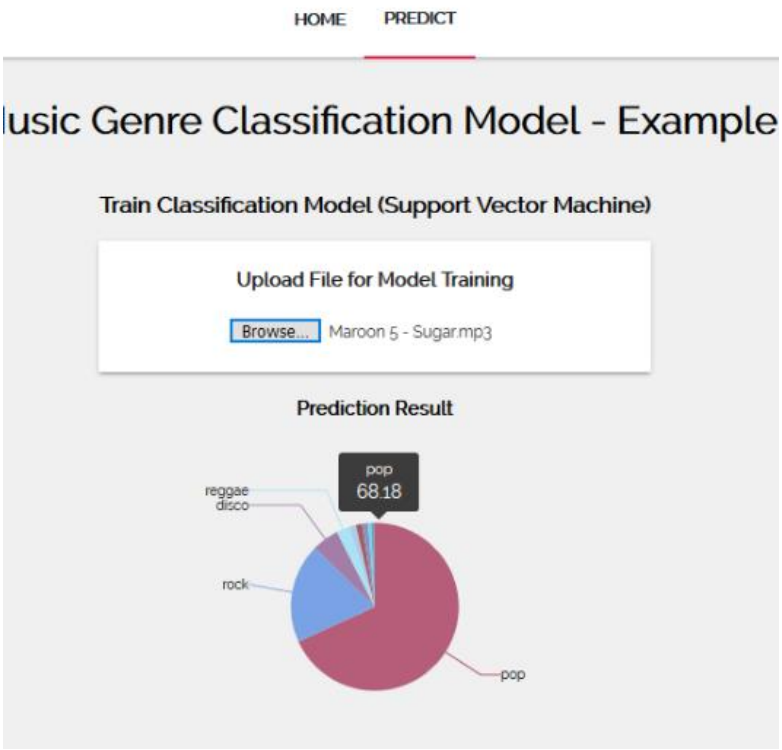
```
clf = tree.DecisionTreeClassifier()

scores = cross_val_score(clf,X,y, cv=100)
print(scores.mean())
```

```
0.44299999999999995
```

- We finally decided to go with SVM and make a web-application that give prediction results based on the uploaded song and user can also train the model by giving model parameters.



- As we have used SVM in training and predicting the result here is a brief introduction of the algorithm.
- A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (*supervised learning*), the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimentional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side.
- Tuning parameters: Kernel, Regularization, Gamma and Margin.

    1. Kernel:
        - Polynomial and exponential kernels calculates separation line in higher dimension. This is called **kernel trick.**

    2. Regularization:
        - The Regularization parameter (often termed as C parameter in python's sklearn library) tells the SVM optimization how much you want to avoid misclassifying each training example.
        - For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points.

    3. Gamma:
        - The gamma parameter defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'. In other words, with low gamma, points far away from plausible seperation line are considered in calculation for the seperation line. Where as high gamma means the points close to plausible line are considered in calculation.

    4. Margin:
        - A ***good margin*** is one where this separation is larger for both the classes. Images below gives to visual example of good and bad margin. A good margin allows the points to be in their respective classes without crossing to other class.

## 4. Summary:

- **Usefulness with respect to existing solution**

Existing solution for music genre classification is to take help of an expert and singer to determine genre of a song which is very hectic if you have a big data set of songs. Now this is no longer a problem with Automated ML approach to classify genre. In addition to this, Accuracy is also enhanced and it takes less time to do so.

- **Advantages**

Automatic classification of music based on its waves and intrinsic property
No need of expertise required
No need to append labels regarding its genre to each song from database

- **Future work**

1. Music recommendation system to help user get his favourite genre by suggesting them some songs.

   In order to recommend song to user we need to follow below steps.

   a. Keep a watch on what kind of genre are in favourite list of user or he listens most.
   b. Generate a dataset based on user likings and preferences.
   c. Train a model based on gathered information about user from above steps and his profile, country, history.
   d. This will incorporate text mining.
   e. Get a list of new songs.
   f. Apply model to find a song for recommendation.
   g. Recommend it.
2. We can also apply unsupervised learning technique for classification of the genre. Unsupervised learning techniques are leveraged when we don't have information about labels of songs available in database..
   a. k means.
   b. Hierarchical clustering.
   c. Probabilistic clustering.
   d. Association rule mining.
3. Semi supervised learning can be used to add new songs to database and train the model for better accuracy.
   a. Predict the genre of new song.
   b. Add new song to respective genre list in database.
   c. Train the model again.
   d. New updated model is ready for further use.

- **Unique Feature**
    1. Automatic classification of song based on genre.
    2. Better and easy-to-use user interface to access the facility.
    3. PyChart- a visual aid to see the percentage of genres in a song has in it.
    4. Better performance because of faster execution and better accuracy.

## 5. References:


[1] Unsupervised Classification of Music Genre Using Hidden Markov Model Xi Shao'*, Changsheng Xu', Mohan S Kankanhalli *2004 IEEE(ICME)* https://pdfs.semanticscholar.org/8830/c2e847c56028f3460542808028fca322ce1b.pdf

[2] AMGC of audio signals with Machine Learning approach George V. Karkavitsas and George A. Tsihrintzis 2013 GSTF www.scielo.br/scielo.php?script=sci_arttext&pid=S0104

[3] Automatic Music Genre Classification by low rank semantic mapping: Yannis Panagakis and Constantine Kotropoulos 2013 springer https://www.eurasip.org/Proceedings/Eusipco/Eusipco2011/papers/1569427041.pdf

[4] AMGC based on musical instrument track separation Aldona Rosner 1 · Bozena Kostek 12 May 2017 springer https://link.springer.com/article/10.1007/s10844-017-0464-5

[5] Music Genre Classification by Indian Institute of Technology, Kanpur Archit Rathore 12152, Margaux Dorido March 16, 2015 https://cse.iitk.ac.in/users/cs365/2015/_submissions/archit/report.pdf

[6] Marsyas "Data Sets". http://marsyasweb.appspot.com/ download/data_sets/

[7] Practical Cryptography "Mel Frequency Cepstral Coefficient (MFCC) tutorial". http://practicalcryptography.com/ miscellaneous/machine-learning/guidemel-frequency-cepstral- coefficients-mfccs/