



Vel Tech
Rangarajan Dr. Sagunthala
R&D Institute of Science and Technology
(Deemed to be University Estd. u/s 3 of UGC Act, 1956)



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING
10214CS602 MINOR PROJECT -II
WINTER SEMESTER(2023-2024)
REVIEW-II**

“AI AUDITOR: CRAFTING YOUR PERFECT PLAYLIST WITH MACHINE LEARNING”

PRESENTED BY

- 1.MANDHALA MANOJ BABU (VTU19600)(21UECM0149)
- 2.CHAKALI VIJAYMANI (VTU19315)(21UECM0041)
- 3.RAPURU GAUTAM (VTU19434)(21UECM0198)

SUPERVISED BY

Mrs.S.Thylasri Assistant professor

OVERVIEW

- ❑ ABSTRACT
- ❑ OBJECTIVE
- ❑ INTRODUCTION
- ❑ LITERATURE REVIEW
- ❑ DESIGN AND METHODOLOGIES
- ❑ IMPLEMENTATION
- ❑ TESTING
- ❑ INPUT AND OUTPUT
- ❑ INCLUDE DEMO VIDEO-1 (Till REVIEW-1)
- ❑ INCLUDE DEMO VIDEO-2(Complete Implementation of Project)
- ❑ CONCLUSION
- ❑ WEB REFERENCES LINK
- ❑ PLAGIARISM REPORT OF PPT
- ❑ REFERENCES

ABSTRACT

- The music industry has undergone major changes from its conventional existence and also in the form of music created in last few years. The ever-growing customer base has also increased the market for different music styles.
- A music genre is a conventional category that identifies some pieces of music as belonging to a shared tradition or set of conventions.
- Music can be divided into different genres in many Different ways. The popular music genres are Pop, Hip-Hop, Rock, Jazz, Blues, Country and Metal.
- Categorizing music files According to their genre is a challenging task in the area of music information retrieval (MIR).
- Automatic music genre Classification is important to obtain music from a large collection. It finds applications in the real world in various fields like Automatic tagging of unknown piece of music (useful for apps like Saavan, Wynk etc.).

OBJECTIVES

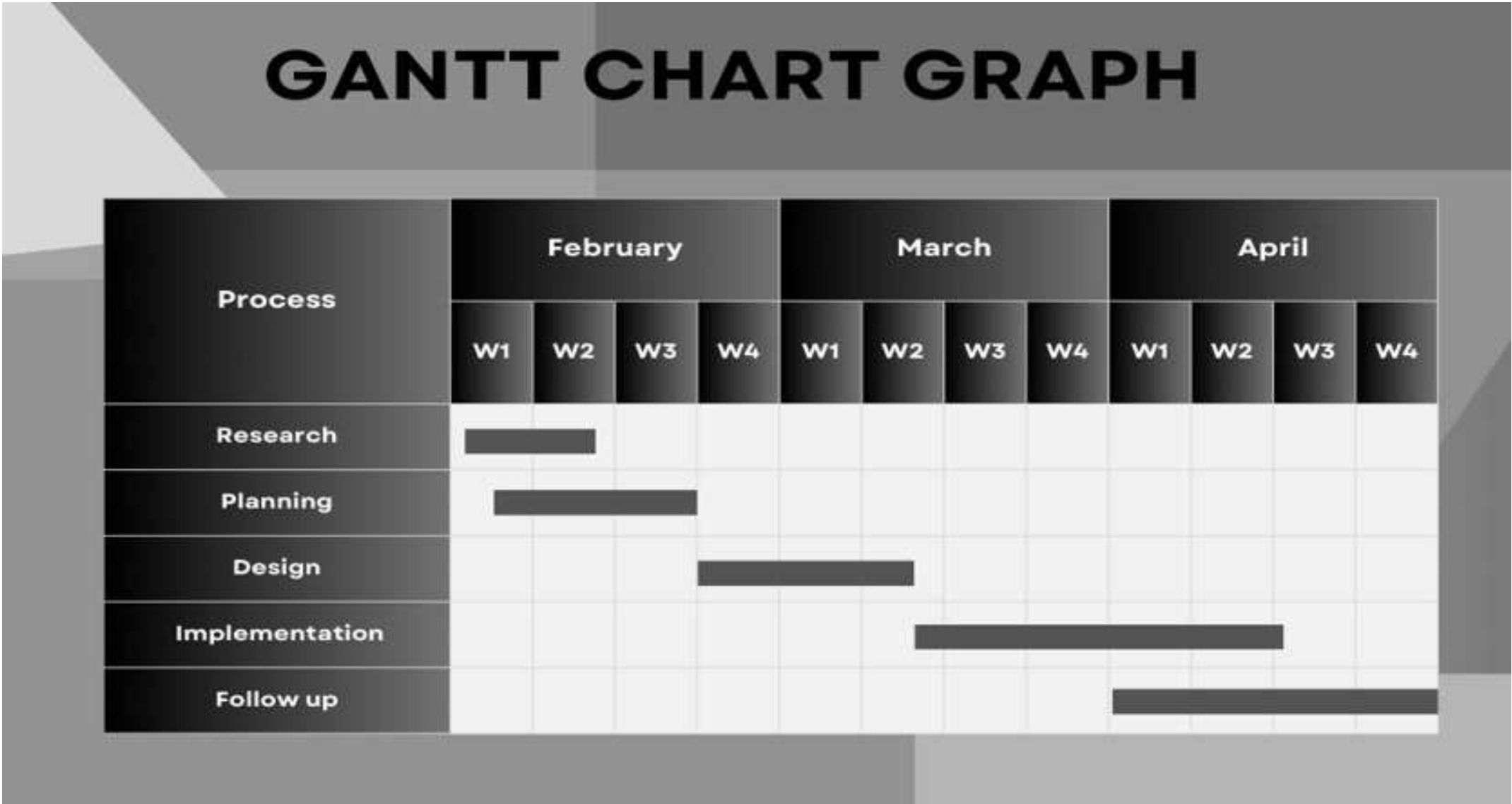
- **Aim of the Project:**

Audio processing is one of the most complex tasks in data science as compared to image processing and other classification techniques. One such application is music genre classification which aims to classify the audio files in certain categories of sound to which they belong. The application is very important and requires automation to reduce the manual error and time because if we have to classify the music manually then one has to listen out each file for the complete duration.

- **Scope of the Project:**

The main scope of the our project " **AI AUDITOR: CRAFTING YOUR PERFECT PLAYLIST WITH MACHINE LEARNING** " to give multiple audio files, and the task is to categorize each audio file in a certain category like audio belongs to Disco, hip-hop, etc.

TIMELINE OF THE PROJECT



INTRODUCTION

- Audio classification is an Application of machine learning where different sound is categorized in certain categories.
- The music genre classification can be built using different approaches in which the top 4 approaches that are mostly used are listed below.
 1. Multiclass support vector machine
 2. K-Nearest Neighbors
 3. K-means clustering algorithm
 4. Convolutional neural network.
- We will use K-Nearest Neighbors algorithm because various researches prove it is one of the best algorithms to give good performance and till time along with optimized models organizations uses this algorithm in recommendation systems as support.
- K-Nearest Neighbour (KNN) is a machine learning algorithm used for regression, and classification. It is also known as the lazy learner algorithm. It simply uses a distance-based method to find the K number of similar Neighbours to new data and the class in which the majority of Neighbours lies, it results in that class as an output.

LITERATURE REVIEW

Sl.No	Author's Name	Paper name and publication details	Year of publication	Main content of the paper
1	Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu	IEEE/ACM Transactions on audio, speech, and language processing 22(10):1533–1545.	2020	Convolutional neural networks for speech recognition
2	Lam Hoang	ACM Symposium on Neural Gaze Detection, Woodstock, NY . ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/ 1122445.112245 .	2018	Literature Review about Music Genre Classification. In Woodstock '18
3	Hareesh Bahuleyan	CoRR, abs/1804.01149	2018	Music genre classification using machine learning techniques
4	Prajwal R*1, Shubham Sharma*2, Prasanna Naik*3, Mrs. Sugna Mk*4	IN Volume:03/Issue:07/July-2021	2021	AI AUDITOR: CRAFTING YOUR PERFECT PLAYLIST WITH MACHINE LEARNING

DESIGN AND METHODOLOGIES

- **MODULE 1:-** Implementation of Music Genre Classification.
- Step 1: Import required libraries.
- Step 2: Processing of data.
- Step 3: Apply Machine Learning Algorithms(KNN).
- Step 4: Identify the nearest neighbors.

MODULE 1: Mention name of the module 1(eg: decision tree)

Step:1 :- Import required libraries.

```
In [3]: #import libraries
import numpy as np
import pandas as pd
import scipy.io.wavfile as wav
from python_speech_features import mfcc
from tempfile import TemporaryFile
import os
import math
import pickle
import random
import operator
```

Step 2: Processing of data.

blues (100 files)

blues.00000.wav
1.32 MB

blues.00001.wav
1.32 MB

blues.00002.wav
1.32 MB

blues.00003.wav
1.32 MB

blues.00004.wav
1.32 MB

blues.00005.wav
1.32 MB

blues.00006.wav
1.32 MB

blues.00007.wav
1.32 MB

blues.00008.wav
1.32 MB

blues.00009.wav
1.32 MB

blues.00010.wav
1.32 MB

blues.00011.wav
1.32 MB

blues.00012.wav
1.32 MB

blues.00013.wav
1.32 MB

blues.00014.wav
1.32 MB

Data Explorer

Version 1 (1.41 GB)

Data

genres_original

blues

blues.00000.wav
blues.00001.wav
blues.00002.wav
blues.00003.wav
blues.00004.wav
blues.00005.wav
blues.00006.wav
blues.00007.wav
blues.00008.wav
blues.00009.wav
blues.00010.wav
blues.00011.wav
blues.00012.wav
blues.00013.wav
blues.00014.wav
blues.00015.wav
blues.00016.wav
blues.00017.wav

country (100 files)

country.00000.wav
1.33 MB

country.00001.wav
1.32 MB

country.00002.wav
1.34 MB

country.00003.wav
1.32 MB

country.00004.wav
1.32 MB

country.00005.wav
1.33 MB

country.00006.wav
1.34 MB

country.00007.wav
1.32 MB

country.00008.wav
1.33 MB

country.00009.wav
1.33 MB

country.00010.wav
1.32 MB

country.00011.wav
1.32 MB

country.00012.wav
1.32 MB

country.00013.wav
1.32 MB

country.00014.wav
1.32 MB

Version 1 (1.41 GB)

Data

genres_original

blues

classical

country

country.00000.v
country.00001.w
country.00002.v
country.00003.v
country.00004.v
country.00005.v
country.00006.v
country.00007.w
country.00008.v
country.00009.v
country.00010.w
country.00011.w
country.00012.w
country.00013.v
country.00014.w
country.00015.w
country.00016.w

Step 3: Using algorithm's name algorithm

K-Nearest Neighbour Algorithm (KNN)

One of them, K-Nearest Neighbour (KNN), is a technique that has been reportedly successful in categorizing music into different genres. Let us find out how.

A supervised machine learning algorithm, the K-Nearest Neighbour technique is used to find solutions for classification and regression problems. Relying on labeled input data to process unlabeled data in the future, this ML technique is used in music genre classification.

Step 1: Select the value of K neighbors(say $k=5$)

Step 3 : Find the K(5) nearest data point for our new data point based on Euclidean distance(which we discuss later)

Step 3: Among these K data points count the data points in each category

Step 4: Assign the new data point to the category that has the most neighbors of the new datapoint

Step 4: Identify the nearest neighbors

In [6]:

```
#function to identify the nearest neighbors
def nearestclass(neighbors):
    classVote = {}

    for x in range(len(neighbors)):
        response = neighbors[x]
        if response in classVote:
            classVote[response] += 1
        else:
            classVote[response] = 1

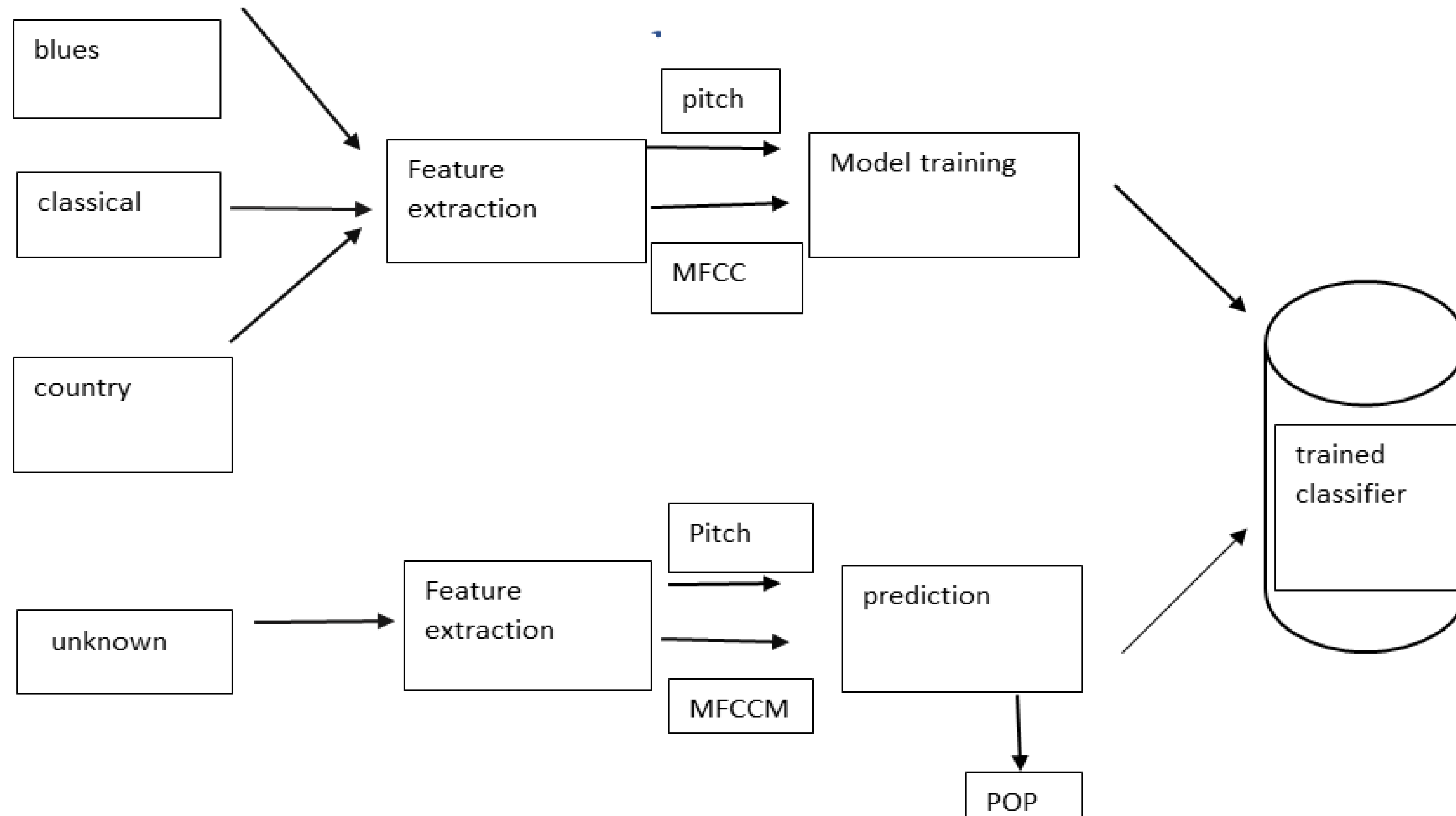
    sorter = sorted(classVote.items(), key=operator.itemgetter(1), reverse=True)
    return sorter[0][0]
```



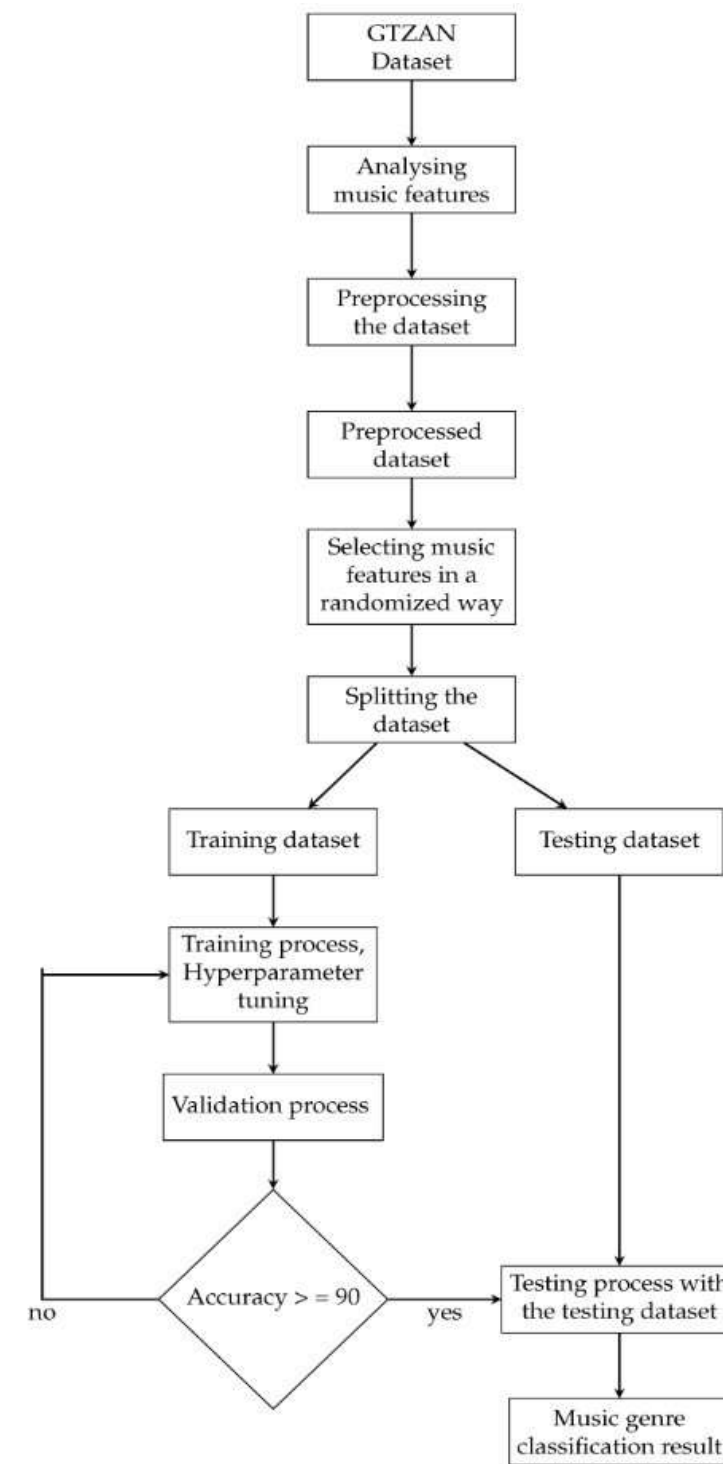
IMPLEMENTATION

- Architecture Diagram
- Data –Flow Diagram
- Use Case Diagram
- Class Diagram
- Activity Diagram

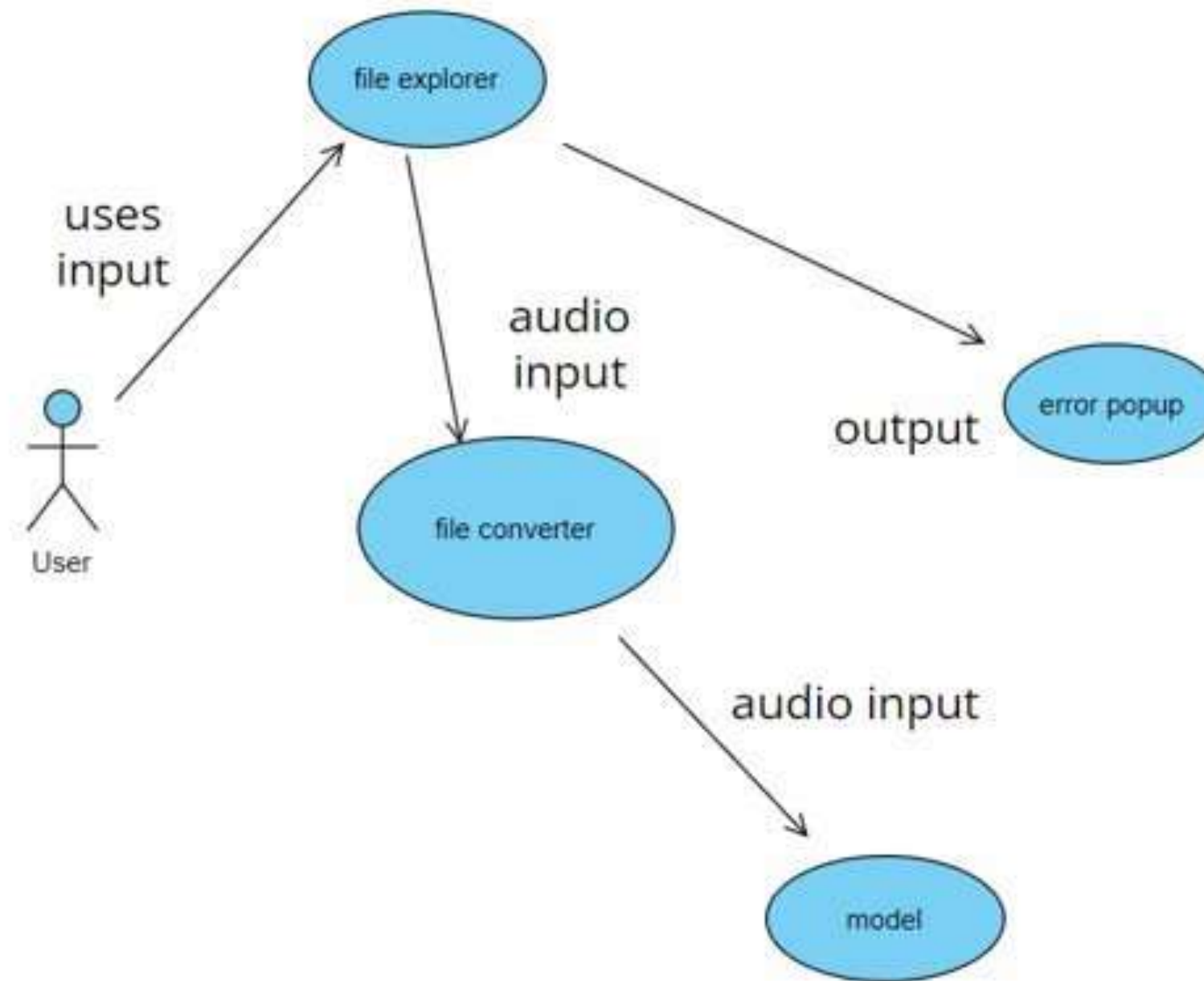
Architecture Diagram



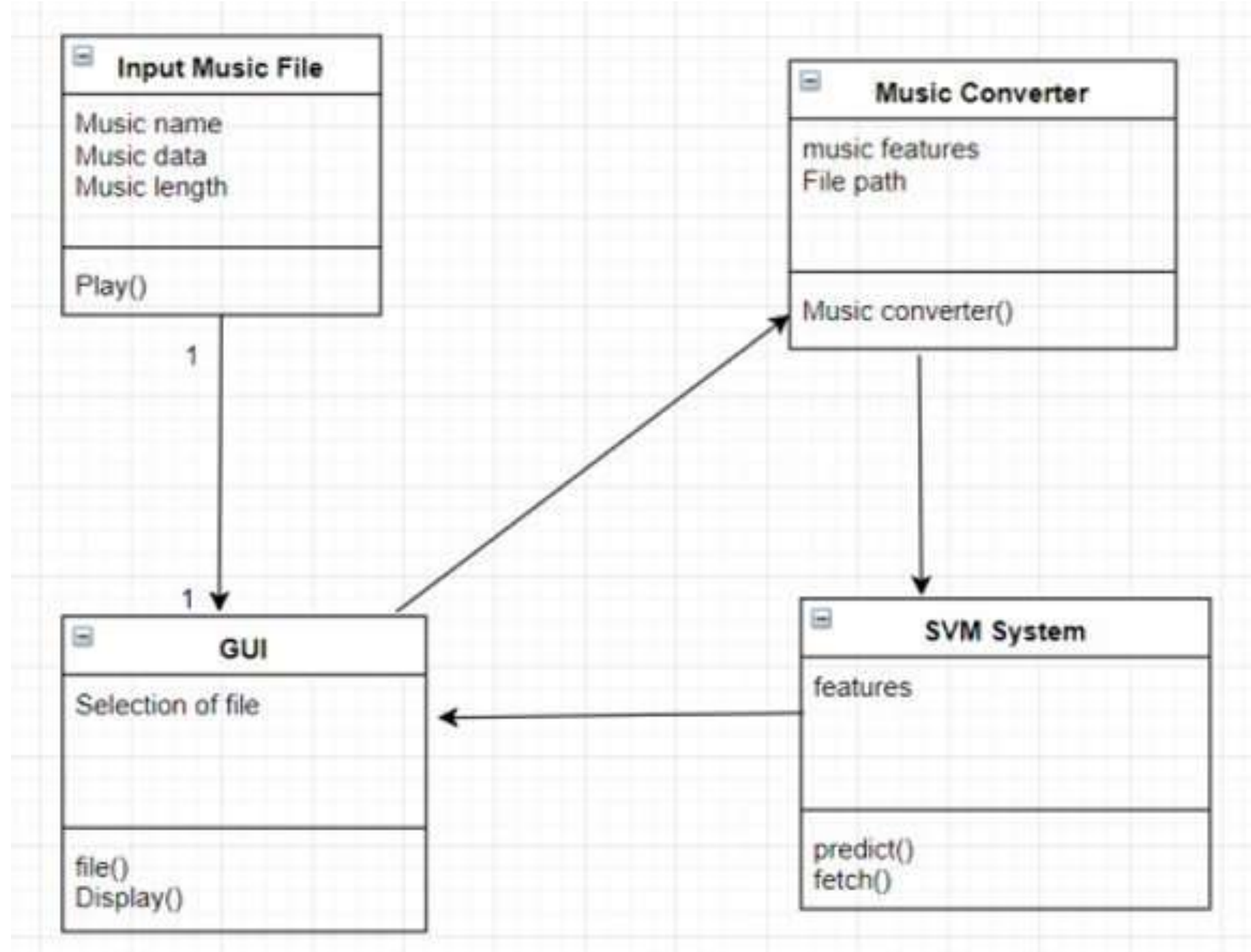
Data –Flow Diagram



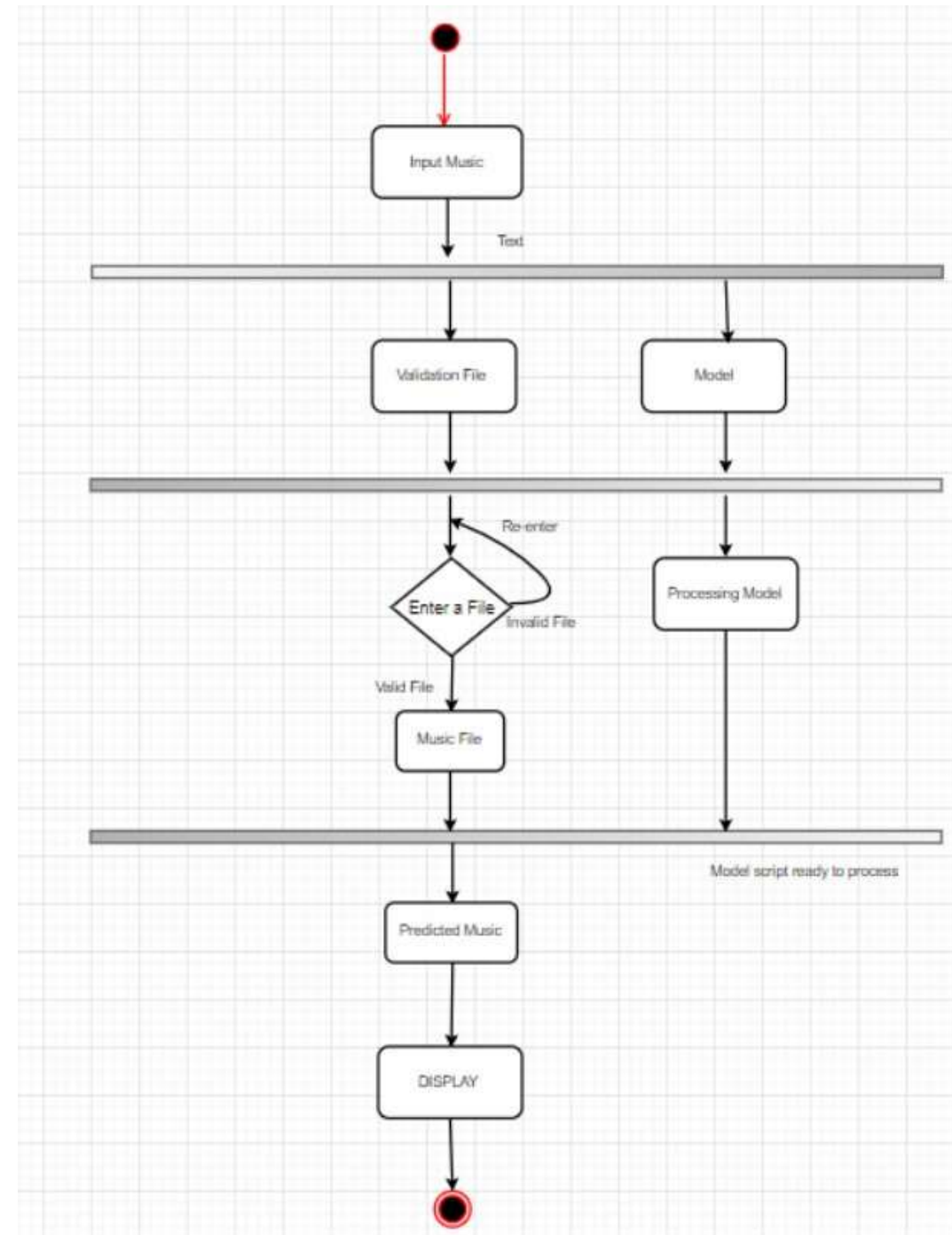
Use Case Diagram



Class Diagram



Activity Diagram

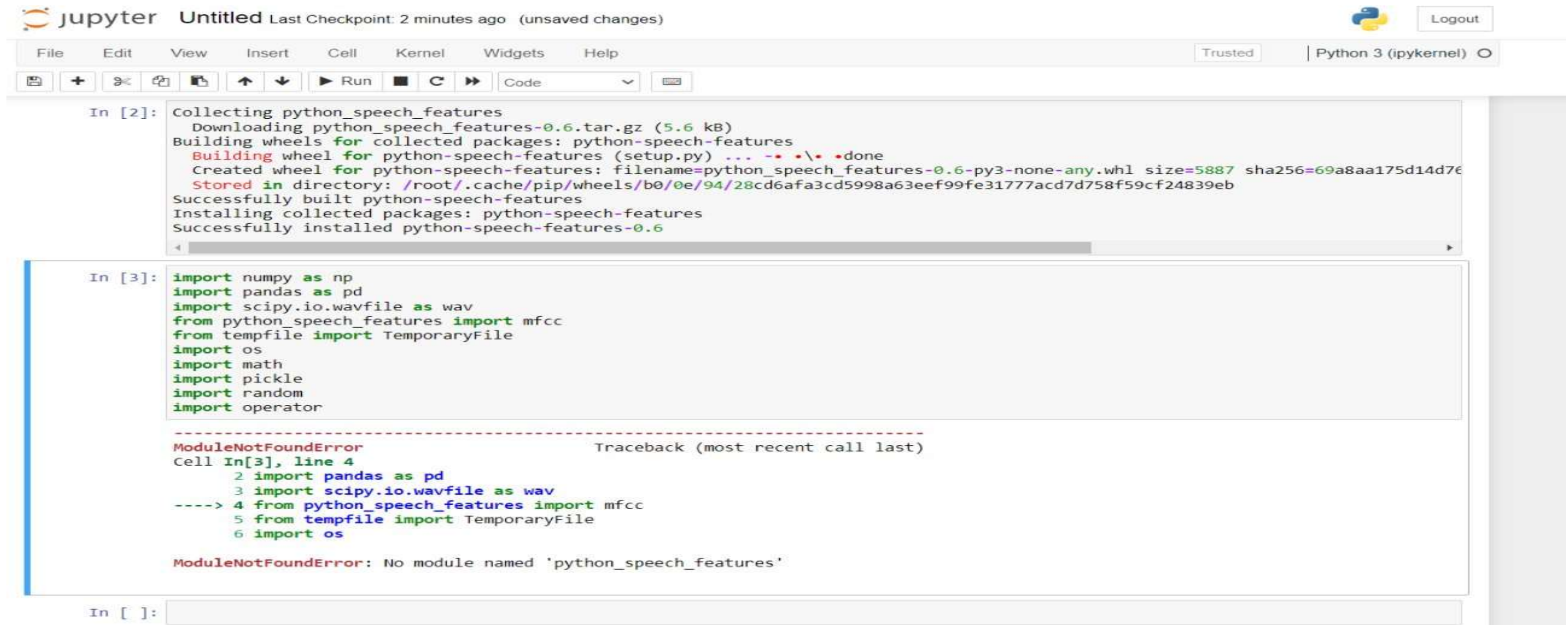


TESTING

- UNIT TESTING
- INTEGRATION TESTING
- FUNCTIONAL TESTING

UNIT TESTING

The first level of testing software is unit testing: they allow us to test the smallest unit of code that makes logical sense to isolate within a system. The cool thing about unit tests is they can be anything you want them to be although it's usually a function, class, or line of code in most programming languages.



```
jupyter Untitled Last Checkpoint: 2 minutes ago (unsaved changes) Python 3 (ipykernel)

File Edit View Insert Cell Kernel Widgets Help

In [2]: Collecting python_speech_features
        Downloading python_speech_features-0.6.tar.gz (5.6 kB)
        Building wheels for collected packages: python-speech-features
        Building wheel for python-speech-features (setup.py) ... - * - * \ * - * done
        Created wheel for python-speech-features: filename=python_speech_features-0.6-py3-none-any.whl size=5887 sha256=69a8aa175d14d76
        Stored in directory: /root/.cache/pip/wheels/b0/0e/94/28cd6afa3cd5998a63eef99fe31777acd7d758f59cf24839eb
        Successfully built python-speech-features
        Installing collected packages: python-speech-features
        Successfully installed python-speech-features-0.6

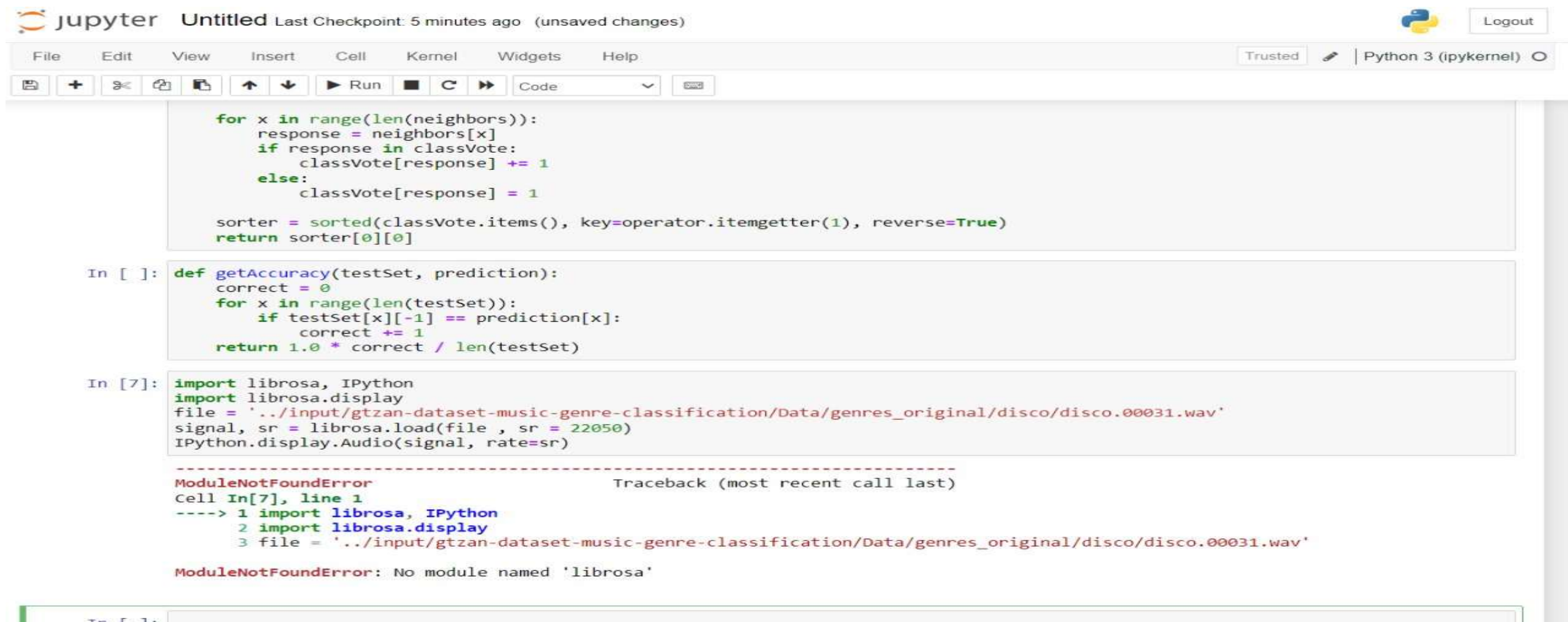
In [3]: import numpy as np
import pandas as pd
import scipy.io.wavfile as wav
from python_speech_features import mfcc
from tempfile import TemporaryFile
import os
import math
import pickle
import random
import operator

-----
ModuleNotFoundError                                Traceback (most recent call last)
Cell In[3], line 4
      2 import pandas as pd
      3 import scipy.io.wavfile as wav
----> 4 from python_speech_features import mfcc
      5 from tempfile import TemporaryFile
      6 import os

ModuleNotFoundError: No module named 'python_speech_features'
```


INTEGRATION TESTING

- **Integration tests:** come after unit tests. The main purpose of integration tests is to find out any irregularity between the interactions of different components of the software.
- After unit tests, it's useful to test how components work together. For that, we use integration testing. Integration testing doesn't necessarily mean testing the whole ML project altogether but one logical part of the project as a single unit.



```
jupyter Untitled Last Checkpoint: 5 minutes ago (unsaved changes) Python 3 (ipykernel) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

for x in range(len(neighbors)):
    response = neighbors[x]
    if response in classVote:
        classVote[response] += 1
    else:
        classVote[response] = 1

sorter = sorted(classVote.items(), key=operator.itemgetter(1), reverse=True)
return sorter[0][0]

In [ ]: def getAccuracy(testSet, prediction):
        correct = 0
        for x in range(len(testSet)):
            if testSet[x][-1] == prediction[x]:
                correct += 1
        return 1.0 * correct / len(testSet)

In [7]: import librosa, IPython
import librosa.display
file = '../input/gtzan-dataset-music-genre-classification/Data/genres_original/disco/disco.00031.wav'
signal, sr = librosa.load(file, sr = 22050)
IPython.display.Audio(signal, rate=sr)

-----
ModuleNotFoundError                                Traceback (most recent call last)
Cell In[7], line 1
----> 1 import librosa, IPython
      2 import librosa.display
      3 file = '../input/gtzan-dataset-music-genre-classification/Data/genres_original/disco/disco.00031.wav'

ModuleNotFoundError: No module named 'librosa'
```

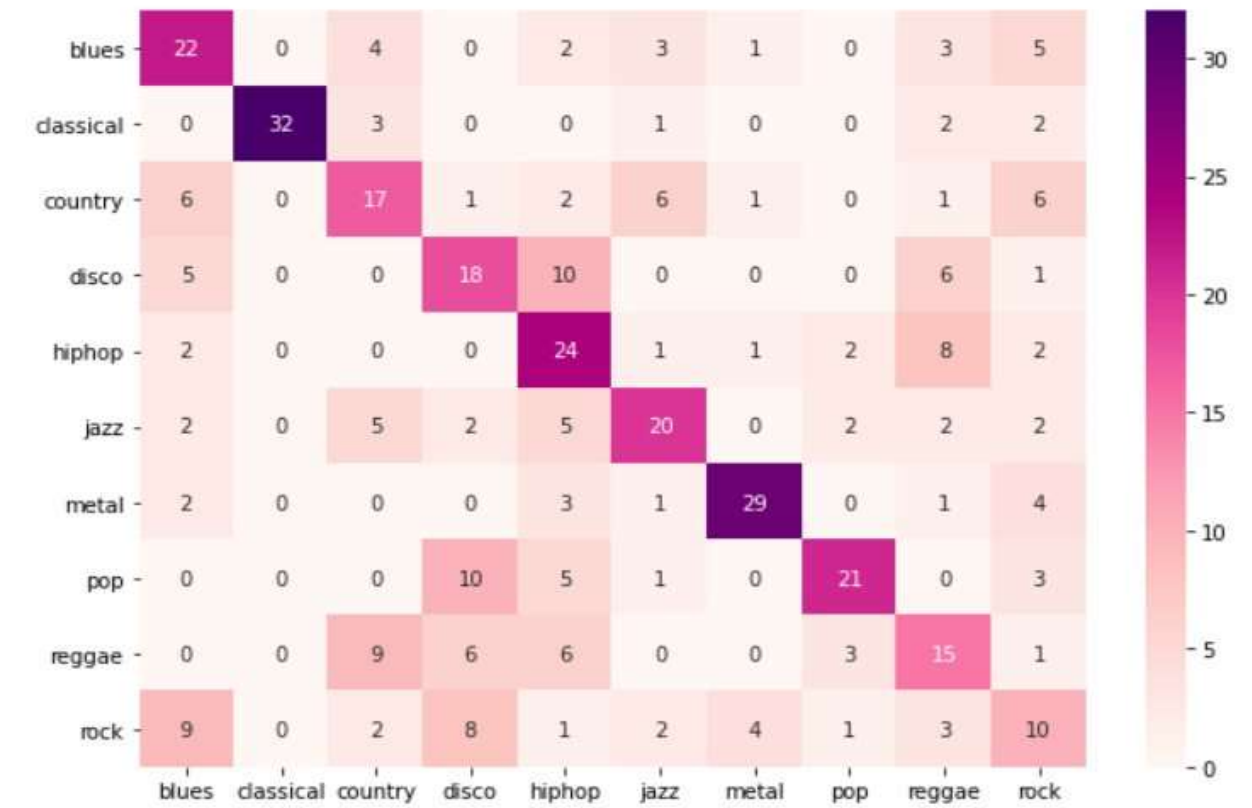
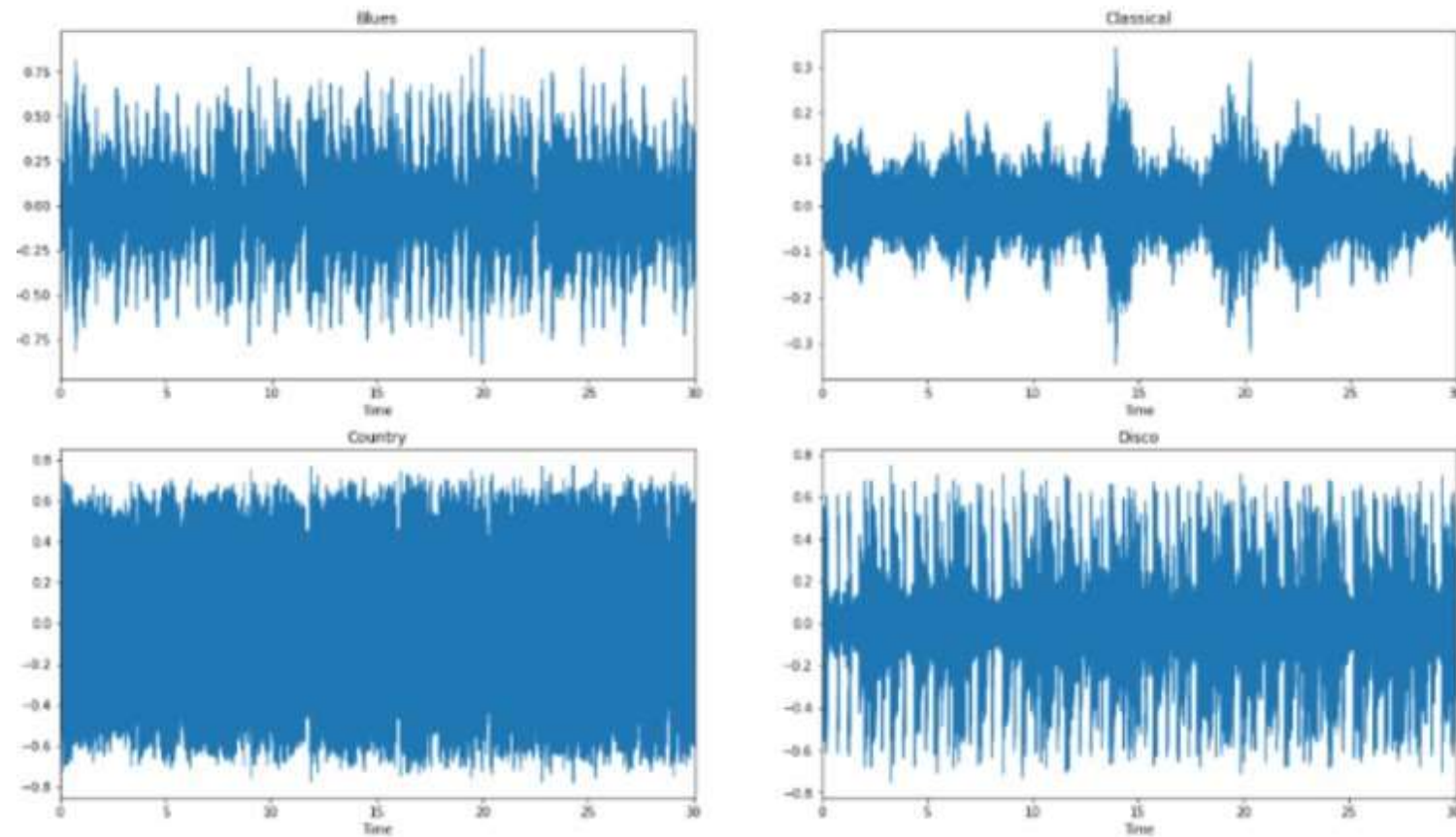
INPUT

features 30 seconds: a csv with a mean and average value of different extracted features from the audio files

filename	length	chroma_stft_mean	chroma_stft_var	rms_mean	rms_var
1000 unique values					
blues.00000.wav	661794	0.35008811950683594	0.08875656872987747	0.1382279233932495	0.00003
blues.00001.wav	661794	0.3409135937690735	0.09498025476932526	0.09594780951738358	0.00003
blues.00002.wav	661794	0.36363717913627625	0.08527519553899765	0.17557041347026825	0.00006
blues.00003.wav	661794	0.4047847092151642	0.09399903565645218	0.14109300076961517	0.00003
blues.00004.wav	661794	0.30852603912353516	0.0878409817814827	0.09152871370315552	0.00003
blues.00005.wav	661794	0.3024562895298004	0.08753237873315811	0.10349363833665848	0.00006
blues.00006.wav	661794	0.29132798314094543	0.09398132562637329	0.14187414944171906	0.00003
blues.00007.wav	661794	0.3079547882080078	0.09290207643671036	0.13182210372021808	0.00003
blues.00008.wav	661794	0.40887922048568726	0.0865124762058258	0.14241649210453033	0.00007



OUTPUT



CONCLUSION

The proposed KNN model has shown improved performance in terms of music genre classification, music similarity and music recommendation, compared to previous studies. In particular, adding some dropout layers into CNN have provided much better results. Owing to the fact that the KNN has only three layers, it is an appropriate model to employ in the music streaming applications for music similarity and recommendation. When the performance results are examined, some similar music genres can lead to mis-classification and mis-recommendation such as Jazz and Classic. In future studies, in order to improve current results, we plan to design more comprehensive deep neural network models and to add extra data models as an input in addition to using only spectrogram. Big data processing techniques and tools can also be utilized for feature extraction and model creation in music genre recommendation systems.

Plagiarism Report of PPT

Curiginal

Document Information

Analyzed document	REVIEW 2 MINOR PROJECT PPT Template WS 20-21 (2).pdf (D138284327)
Submitted	2022-05-28T09:09:00.0000000
Submitted by	Umanandhini
Submitter email	drumanandhini@veltech.edu.in
Similarity	10%
Analysis address	drumanandhini.veltec@analysis.orkund.com

Sources included in the report

W	URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3712136 Fetched: 2022-05-28T09:09:42.5530000		4
SA	Rport 11.docx Document Rport 11.docx (D132187989)		2
SA	Patel Shivani Niravbhai Ph.D Computer Science and Application Thesis.pdf Document Patel Shivani Niravbhai Ph.D Computer Science and Application Thesis.pdf (D123346275)		1

Web references/video links

§ <https://www.kaggle.com/andradaolteanu/gtzandataset-music-genre-classification>

§ <https://www.analyticsvidhya.com/blog/2022/03/music-genre-classification-project-using-machine-learning-techniques/>

§ <https://www.clairvoyant.ai/blog/music-genre-classification-using-cnn>

§ <https://www.geeksforgeeks.org/music-genre-classifier-using-machine-learning/>

§ <https://www.analyticssteps.com/blogs/music-genre-classification-using-machine-learning>

REFERENCES

- A. Kayatana and O. Yildiz, “Music genre classification with machine learning techniques,” in Signal Processing and Communications Applications Conference (SIU), 2017 25th, IEEE, 2019, pp. 1–4.
- A. Tsantakis, G. and Cook, P. “Musical genre classification of audio signal”, IEEE Transactions on Speech and Audio Processing, Vol. 10, No. 3, pp. 293-302, July 2020.
- Holzapfel, A. and Stylianou Y. “Musical genre classification using nonnegative matrix factorization-based features”, IEEE Transactions on Audio, Speech, and Language Processing, Vol. 16, No. 2, pp. 424-434, 2019.
- Sturm, B. L. (2020). The GTZAN dataset: Its contents, its faults, their effects on evaluation, and its future use. Arxiv preprint arXiv:1306.1461.
- Gemmate, J. F., Ellis, D. P., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., ... & Ritter, M. (2017, Mar) IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 776-780) IEEE. Audio set: An ontology and human-labelled dataset for audio events. In 2020

THANK YOU



Vel Tech
Rangarajan Dr. Sagunthala
R&D Institute of Science and Technology
(Deemed to be University Estd. u/s 3 of UGC Act, 1956)

