Artificial Intelligence and Machine Learning (CSHO631AIP)

Mini Project Report
on

"MUSIC GENRE CLASSIFICATION"

Submitted in partial fulfillment of Continuous Internal Assessment of
Internet and Web Programming CS541

Computer Science and Engineering
by

K Bhargav Sai Reddy 19603471
Meghashyam Malur 1960359

Under the Guidance of
Dr. Rekha V,
Assistant Professor

Department of Computer Science and Engineering,
School of Engineering and Technology,
CHRIST (Deemed to be University),
Kumbalgodu, Bangalore - 560 074

April - 2022

CHRIST
(DEEMED TO BE UNIVERSITY)
BANGALORE · INDIA

School of Engineering and Technology
Department of Computer Science and Engineering

CERTIFICATE

This is to certify that **Kothapalli Bhargav Sai Reddy - 1960347, Meghashyam Malur -1960359**, have successfully completed the project work entitled in partial fulfillment for submission Internal Assessment of Mini Project in Artificial Intelligence and Machine Learning (CSHO631AIP) 2021-2022.

Dr Rekha V	Dr K Balachandran
Subject Incharge	Head of the Department,
	Computer Science and Engineering

CHRIST
(DEEMED TO BE UNIVERSITY)
BANGALORE · INDIA

Acknowledgment

We would like to thank CHRIST (Deemed to be University) Vice-Chancellor, Dr. Rev. Fr Abraham V M, Pro-Vice-Chancellor, Dr. Rev. Fr Joseph C C, Director of School of Engineering and Technology, Dr. Rev. Fr Benny Thomas, and the Dean Dr. Iven Jose for their kind patronage.

We would like to express our sincere gratitude and appreciation to the Head of the Department of Computer Science and Engineering, School of Engineering and Technology, Dr. K Balachandran, for giving us this opportunity to take up this project.

We are also extremely grateful to our Staff Incharge, Dr. Rekha V, Assistant professor, Department of CSE who has supported and helped to carry out the project. Her constant monitoring and encouragement helped us keep up with the project schedule.

We also thank all the faculty members for their continuous support and constructive inputs that have helped shape this project. This project would not be complete if it were not for the variety of ideas put forward by our classmates and peers. Last, but not least we thank our parents for being supportive in all challenges we faced during this project.

Abstract

Music recommendation and classification systems are an area of interest in digital signal processing and digital music processing.

In this study by using digital signal processing techniques and autoencoders, music features are extracted, and then with these features music classification and clustering have been done, and with the results, music recommendation has been made.

Obtained results are compared with each other. In the study, the GTZAN dataset has been used. The purpose of this study is to compare the result feature extraction with autoencoders and digital signal processing techniques.

The Primary objective of this project is to successfully train an ML Algorithm to be able to classify any sound byte of any piece of music as one of the common genres in music.

The inherent nature of music to have recurring patterns, set tempos, and other aspects of music theory make AI/ML algorithms perfect for the job of understanding and classifying music. the music files themselves can be considered data as they carry all technical and categorically discernable parameters of the music are self-contained in the file thereby only needing for them to be of set sample lengths/sizes and set file formats.

The data set used in this project contains music files of 30 seconds in length, in .WAV format and in the 16 Bit PCM WAVE format in 0000x01 Encoding making it suitable for python based scipy.io.wave library.

Introduction

With the growth of online music databases and easy access to music content, people find it increasing hard to manage the songs that they listen to.

One way to categorize and organize songs is based on the genre, which is identified by some characteristics of the music such as rhythmic structure, harmonic content, and instrumentation.

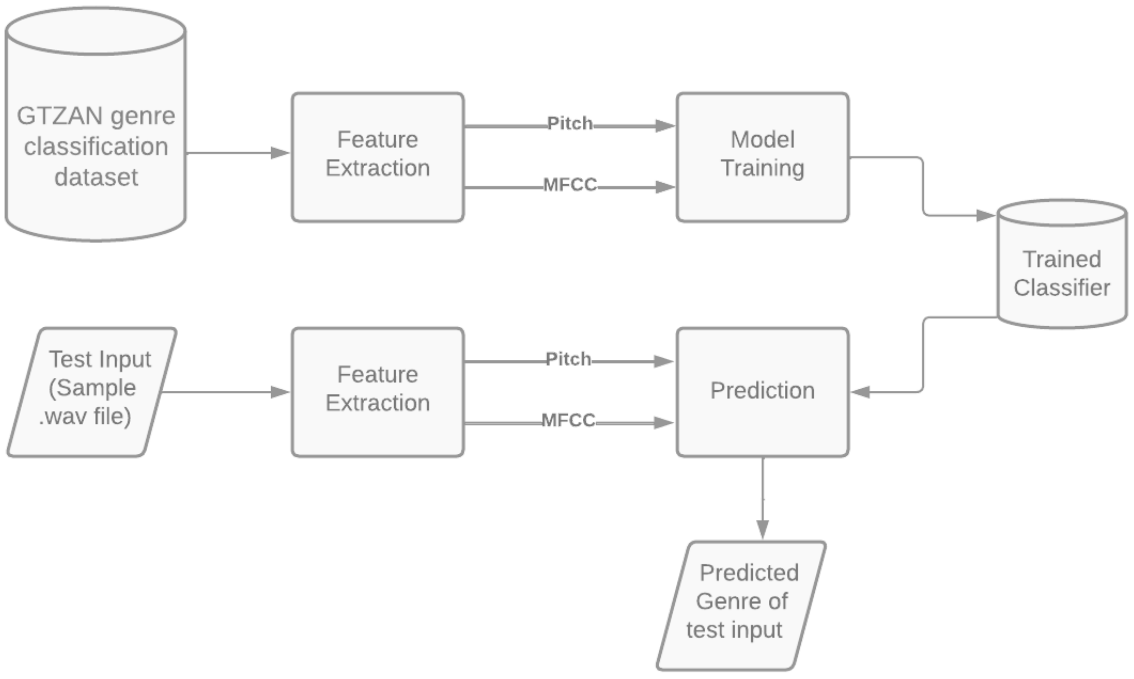
Being able to automatically classify and provide tags to the music present in a user's library, based on genre, would be beneficial for audio streaming services such as Spotify and iTunes.This study explores the application of machine learning (ML)
algorithms to identify and classify the genre of a given1audio file.

The GTZAN dataset is the most-used public dataset for evaluation in machine listening research for music genre recognition (MGR). The files were collected in 2000-2001 from a variety of sources including personal CDs, radio, and microphone recordings, in order to represent a variety of recording conditions.

Design

This project can be divided into 3 major parts:

1. Feature Extraction
2. Model Training
3. Prediction



Feature Extraction:

The first step for this project would be to extract features and components from the audio files. It includes identifying the linguistic content and discarding noise.

Mel Frequency Cepstral Coefficients:

- These are state-of-the-art features used in automatic speech and speech recognition studies. There are a set of steps for the generation of these features:
- Since the audio signals are constantly changing, first we divide these signals into smaller frames. Each frame is around 20-40 ms long
- Then we try to identify different frequencies present in each frame
- Now, separate linguistic frequencies from the noise

- To discard the noise, it then takes discrete cosine transform (DCT) of these frequencies. Using DCT we keep only a specific sequence of frequencies that have a high probability of information

Model Training:

In this project, the K-nearest neighbors algorithm is being used because various prior research have proved that this algorithm gives the best results for this problem of classification of music

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on the Supervised Learning technique.

it assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.

K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.

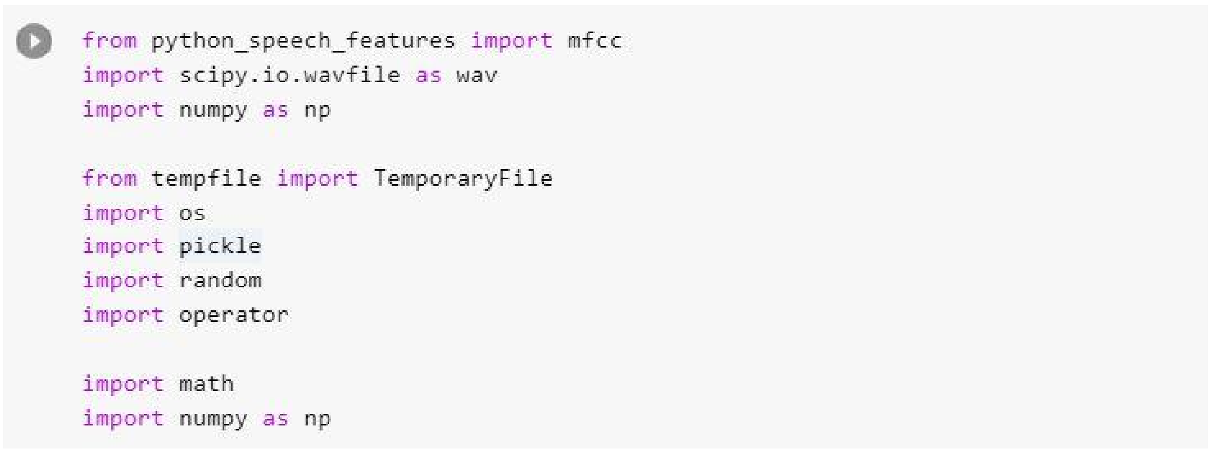
Prediction:

This step involves using test data to try and predict the genre of the sample music file against the trained model and gain accuracy parameters.

Implementation

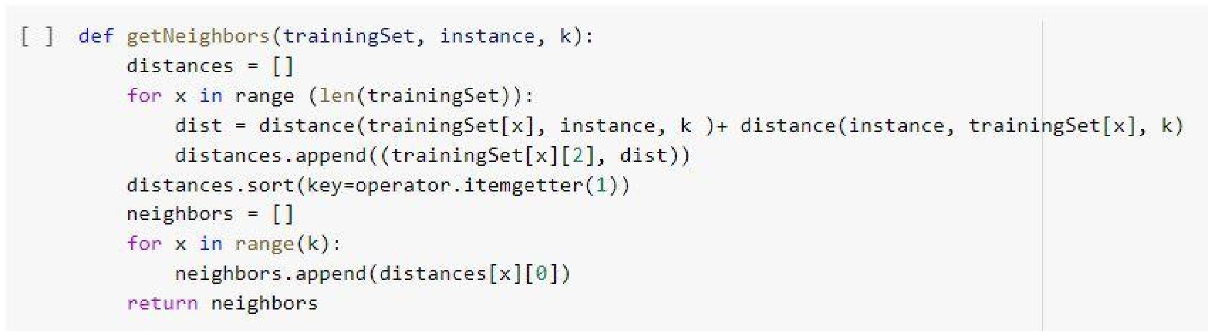
Implementation involves the following steps:

1. Imports:



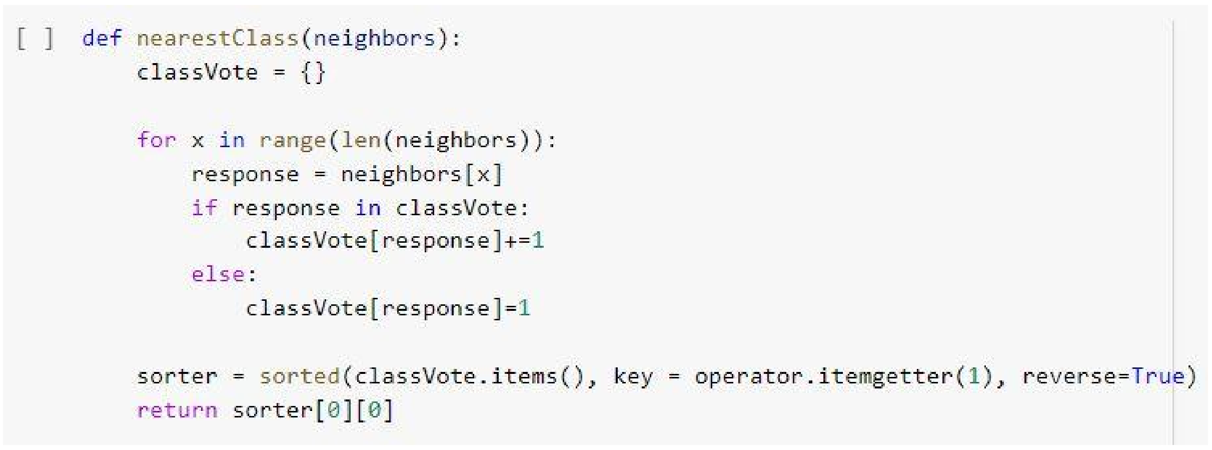
```python
from python_speech_features import mfcc
import scipy.io.wavfile as wav
import numpy as np

from tempfile import TemporaryFile
import os
import pickle
import random
import operator

import math
import numpy as np
```

2. Defining a function to calculate distance between feature vectors and find neighbors:



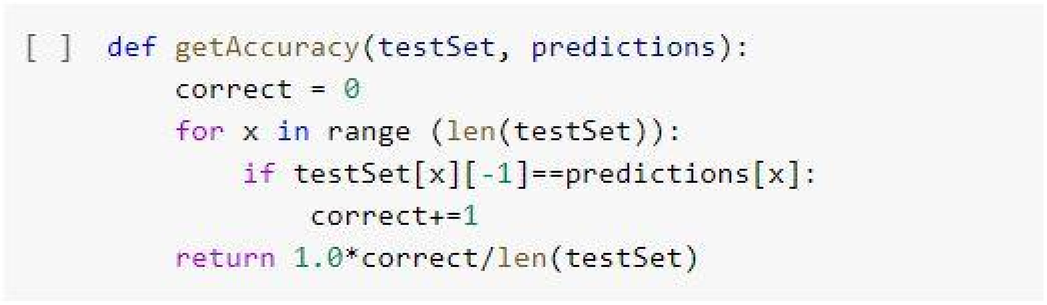
```python
def getNeighbors(trainingSet, instance, k):
    distances = []
    for x in range (len(trainingSet)):
        dist = distance(trainingSet[x], instance, k )+ distance(instance, trainingSet[x], k)
        distances.append((trainingSet[x][2], dist))
    distances.sort(key=operator.itemgetter(1))
    neighbors = []
    for x in range(k):
        neighbors.append(distances[x][0])
    return neighbors
```

3. Identifying the nearest neighbors:



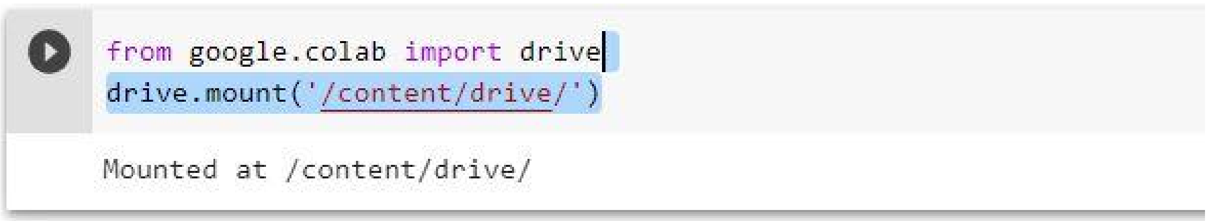
```python
def nearestClass(neighbors):
    classVote = {}

    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]
```

4. Define a function for model evaluation and performance metrics:

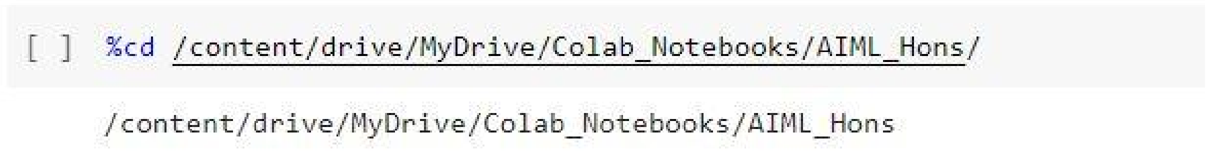


```
[ ]  def getAccuracy(testSet, predictions):
         correct = 0
         for x in range (len(testSet)):
             if testSet[x][-1]==predictions[x]:
                 correct+=1
         return 1.0*correct/len(testSet)
```

5. Deploying Dependencies i.e Dataset from google drive:

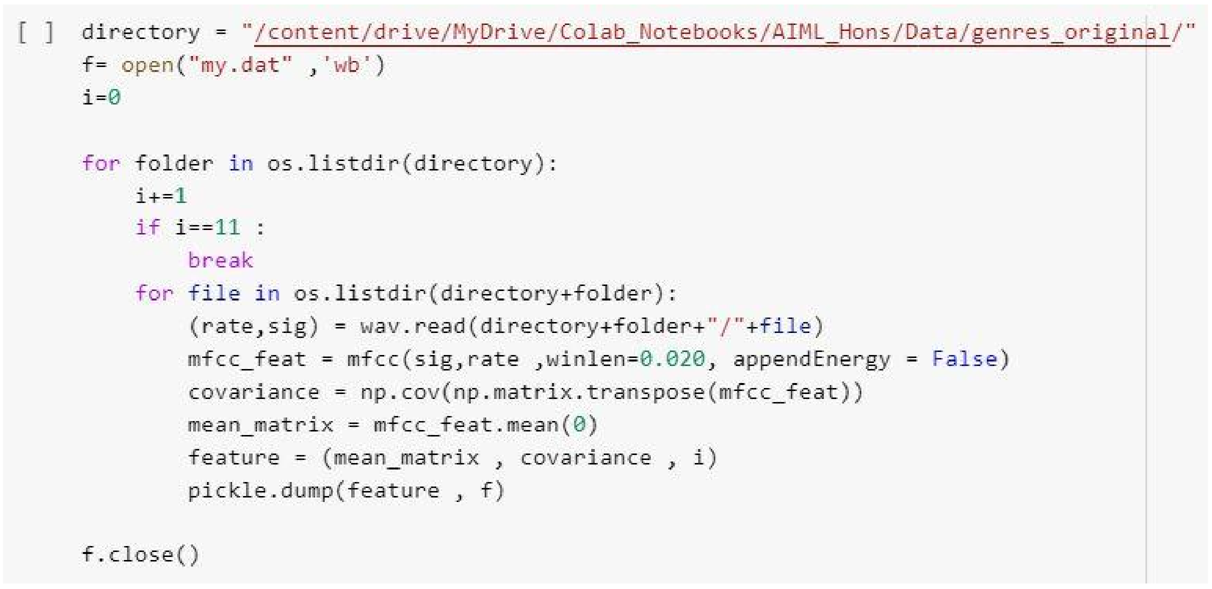


```
from google.colab import drive
drive.mount('/content/drive/')

Mounted at /content/drive/
```



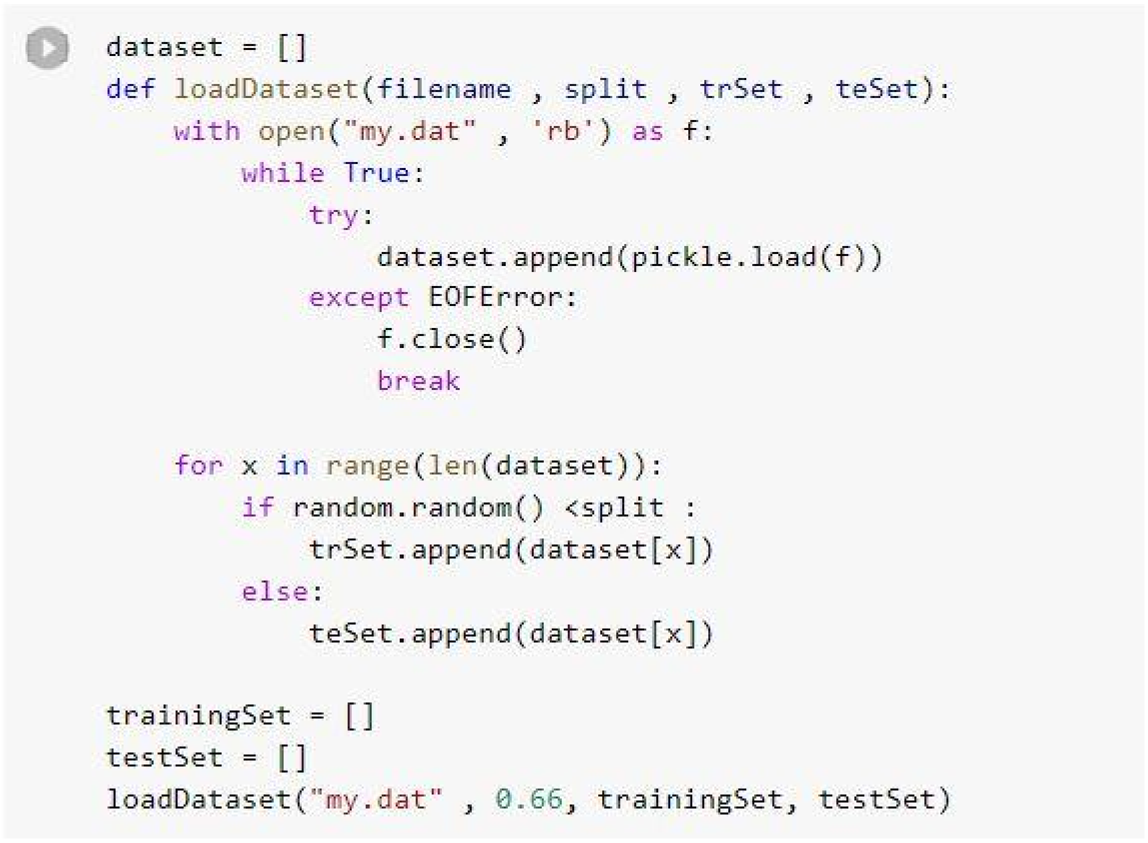
```
[ ]  %cd /content/drive/MyDrive/Colab_Notebooks/AIML_Hons/

     /content/drive/MyDrive/Colab_Notebooks/AIML_Hons
```

6. Extracting features from the dataset and exporting the extracted features into a binary .dat file "my.dat":



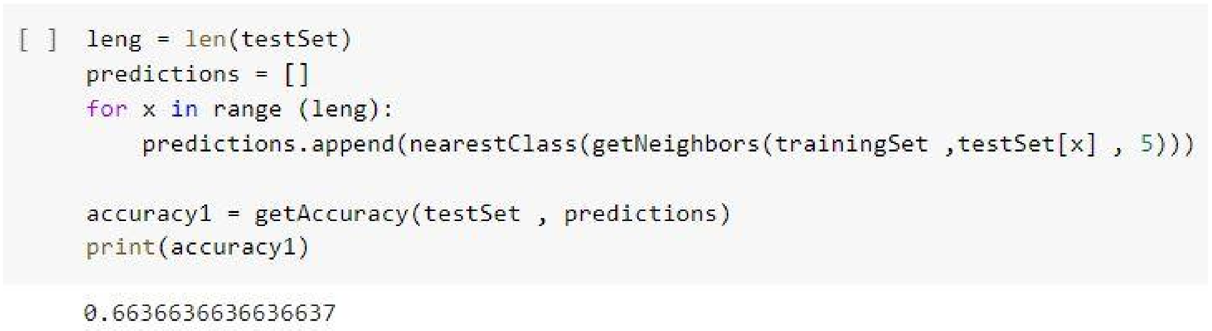
```
[ ]  directory = "/content/drive/MyDrive/Colab_Notebooks/AIML_Hons/Data/genres_original/"
     f= open("my.dat" ,'wb')
     i=0

     for folder in os.listdir(directory):
         i+=1
         if i==11 :
             break
         for file in os.listdir(directory+folder):
             (rate,sig) = wav.read(directory+folder+"/"+file)
             mfcc_feat = mfcc(sig,rate ,winlen=0.020, appendEnergy = False)
             covariance = np.cov(np.matrix.transpose(mfcc_feat))
             mean_matrix = mfcc_feat.mean(0)
             feature = (mean_matrix , covariance , i)
             pickle.dump(feature , f)

     f.close()
```

7. Train and test split on the dataset:



```
dataset = []
def loadDataset(filename , split , trSet , teSet):
    with open("my.dat" , 'rb') as f:
        while True:
            try:
                dataset.append(pickle.load(f))
            except EOFError:
                f.close()
                break

    for x in range(len(dataset)):
        if random.random() <split :
            trSet.append(dataset[x])
        else:
            teSet.append(dataset[x])

trainingSet = []
testSet = []
loadDataset("my.dat" , 0.66, trainingSet, testSet)
```

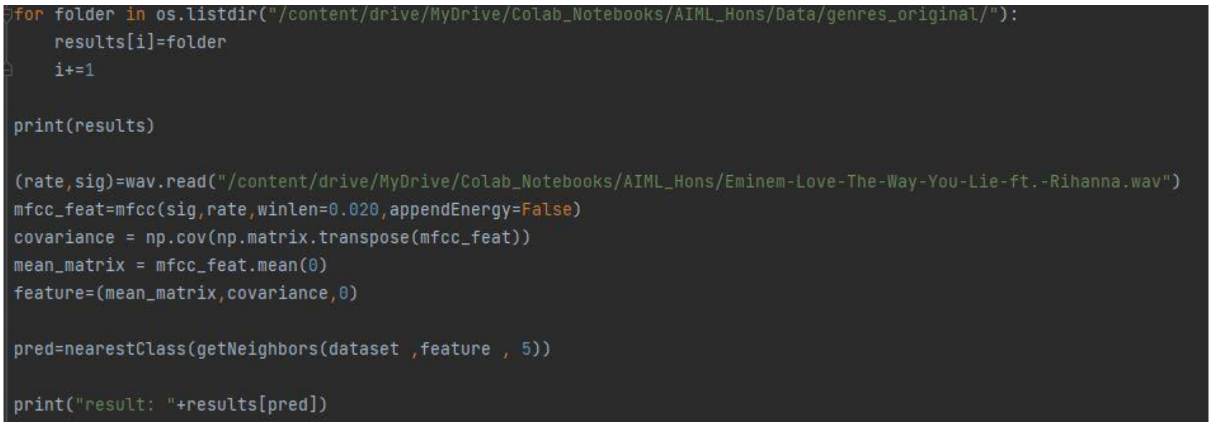
8. Make prediction using KNN and get the accuracy on test data:



```
leng = len(testSet)
predictions = []
for x in range (leng):
    predictions.append(nearestClass(getNeighbors(trainingSet ,testSet[x] , 5)))

accuracy1 = getAccuracy(testSet , predictions)
print(accuracy1)
```

```
0.6636636636636637
```

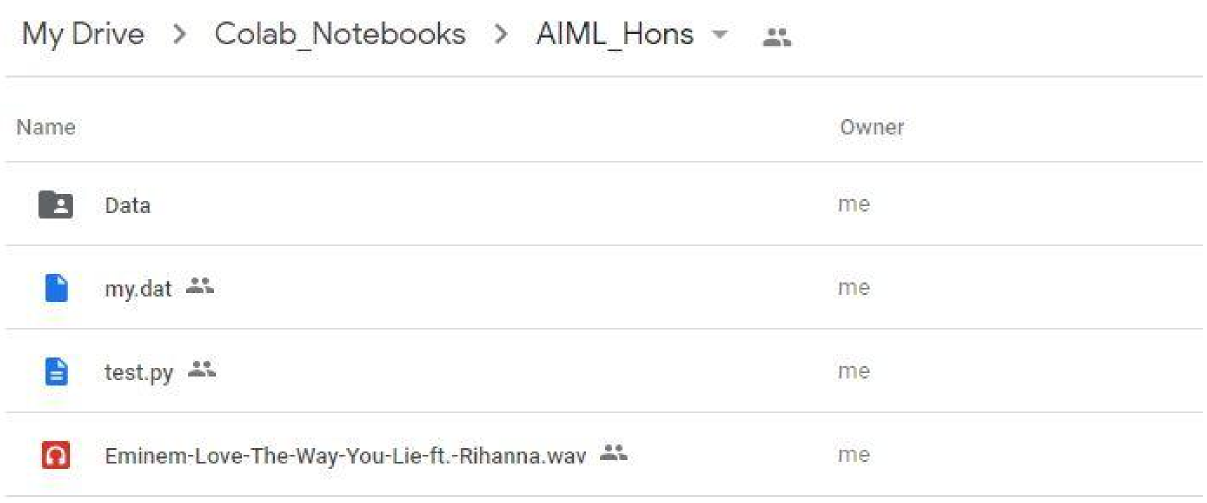
9. Testing the Trained Model against a test music file:



```
for folder in os.listdir("/content/drive/MyDrive/Colab_Notebooks/AIML_Hons/Data/genres_original/"):
    results[i]=folder
    i+=1

print(results)

(rate,sig)=wav.read("/content/drive/MyDrive/Colab_Notebooks/AIML_Hons/Eminem-Love-The-Way-You-Lie-ft.-Rihanna.wav")
mfcc_feat=mfcc(sig,rate,winlen=0.020,appendEnergy=False)
covariance = np.cov(np.matrix.transpose(mfcc_feat))
mean_matrix = mfcc_feat.mean(0)
feature=(mean_matrix,covariance,0)

pred=nearestClass(getNeighbors(dataset ,feature , 5))

print("result: "+results[pred])
```

Results and Discussion

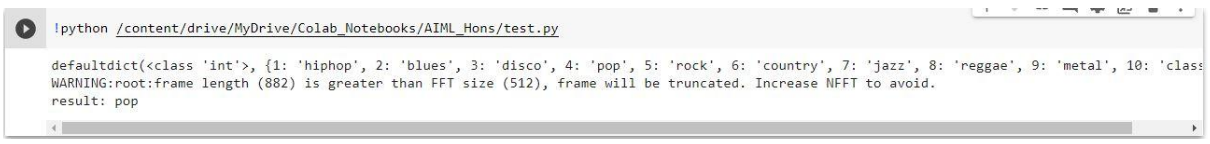
1.Dependencies staggered in required directories:



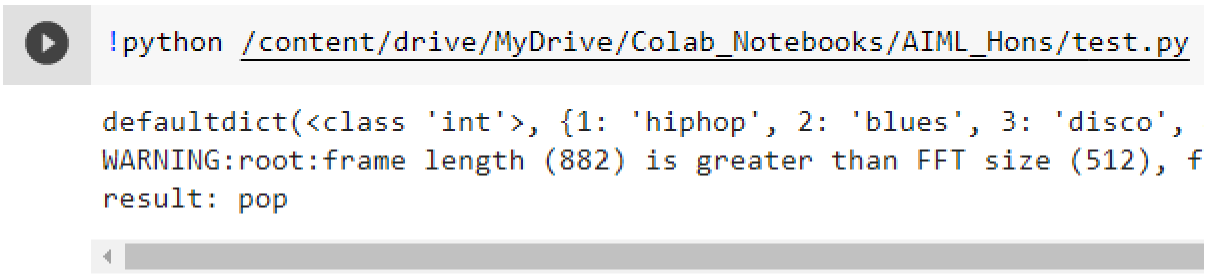
My Drive > Colab_Notebooks > AIML_Hons

Name	Owner
Data	me
my.dat	me
test.py	me
Eminem-Love-The-Way-You-Lie-ft.-Rihanna.wav	me

2. Running the aforementioned Test.py script:



```
!python /content/drive/MyDrive/Colab_Notebooks/AIML_Hons/test.py

defaultdict(<class 'int'>, {1: 'hiphop', 2: 'blues', 3: 'disco', 4: 'pop', 5: 'rock', 6: 'country', 7: 'jazz', 8: 'reggae', 9: 'metal', 10: 'class
WARNING:root:frame length (882) is greater than FFT size (512), frame will be truncated. Increase NFFT to avoid.
result: pop
```



```
!python /content/drive/MyDrive/Colab_Notebooks/AIML_Hons/test.py

defaultdict(<class 'int'>, {1: 'hiphop', 2: 'blues', 3: 'disco',
WARNING:root:frame length (882) is greater than FFT size (512), f
result: pop
```

The Test music file is fed to the Trained Model and the desired output of the genre (POP) of the music file has been successfully predicted.

Conclusion

This project entailed securing a standardized dataset, deciding on an efficient model, training the model against the dataset, and making predictions using the same. with all these steps completed.

We have successfully managed to implement a supervised learning algorithm to classify a given Test music file into its designated genre.

The model produces an accuracy metric of 68.35% and is a decent result for a KNN model with a dataset of this scale.

in conclusion the project has successfully achieved the objectives it had set out to achieve in Phase 1.

\

Scope for Future work

This Project was implemented with future expansion in mind throughout its lifecycle.

The primary objective was to create a system that can classify one of commercial music's primary discerning parameters that is the genre of the piece of music. With the primary differentiating parameter settled the next logical step would be to package this project as a portable plugin so that entities in the music industry can easily deploy and use the functionality of this project.

This project was also designed to be subsystem for a larger music intellectual property protection system that would help stakeholders prevent Intellectual Property issues that may rise up at later stages in music production and release.

Bibliography

Music Genre Classification using Machine Learning Techniques arXiv:1804.01149 [cs.SD]

MusicGenreClassification
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.375.204&rep=rep1&type=pdf

https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

Silla, C.N., Koerich, A.L. & Kaestner, C.A.A. A Machine Learning Approach to Automatic Music Genre Classification. J Braz Comp Soc 14, 7–18 (2008). https://doi.org/10.1007/BF03192561

A. Ghildiyal, K. Singh and S. Sharma, "Music Genre Classification using Machine Learning," 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA), 2020, pp. 1368-1372, doi: 10.1109/ICECA49313.2020.9297444.

http://marsyas.info/downloads/datasets.html (GTZAN Genre Collection Dataset)