import librosa

import soundfile

import os, glob, pickle

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.neural\_network import MLPClassifier

from sklearn.metrics import accuracy\_score

#DataFlair - Extract features (mfcc, chroma, mel) from a sound file

def extract\_feature(file\_name, mfcc, chroma, mel):

with soundfile.SoundFile(file\_name) as sound\_file:

X = sound\_file.read(dtype="float32")

sample\_rate=sound\_file.samplerate

if chroma:

stft=np.abs(librosa.stft(X))

result=np.array([])

if mfcc:

mfccs=np.mean(librosa.feature.mfcc(y=X, sr=sample\_rate, n\_mfcc=40).T, axis=0)

result=np.hstack((result, mfccs))

if chroma:

chroma=np.mean(librosa.feature.chroma\_stft(S=stft, sr=sample\_rate).T,axis=0)

result=np.hstack((result, chroma))

if mel:

mel=np.mean(librosa.feature.melspectrogram(X, sr=sample\_rate).T,axis=0)

result=np.hstack((result, mel))

return result

#DataFlair - Emotions in the RAVDESS dataset

emotions={

'01':'neutral',

'02':'calm',

'03':'happy',

'04':'sad',

'05':'angry',

'06':'fearful',

'07':'disgust',

'08':'surprised'

}

#DataFlair - Emotions to observe

observed\_emotions=['calm', 'happy', 'fearful', 'disgust']

#DataFlair - Load the data and extract features for each sound file

def load\_data(test\_size=0.2):

x,y=[],[]

for file in glob.glob("D:\\DataFlair\\ravdess data\\Actor\_\*\\\*.wav"):

file\_name=os.path.basename(file)

emotion=emotions[file\_name.split("-")[2]]

if emotion not in observed\_emotions:

continue

feature=extract\_feature(file, mfcc=True, chroma=True, mel=True)

x.append(feature)

y.append(emotion)

return train\_test\_split(np.array(x), y, test\_size=test\_size, random\_state=9)

#DataFlair - Split the dataset

x\_train,x\_test,y\_train,y\_test=load\_data(test\_size=0.25)

#DataFlair - Get the shape of the training and testing datasets

print((x\_train.shape[0], x\_test.shape[0]))

#DataFlair - Get the number of features extracted

print(f'Features extracted: {x\_train.shape[1]}')

#DataFlair - Initialize the Multi Layer Perceptron Classifier

model=MLPClassifier(alpha=0.01, batch\_size=256, epsilon=1e-08, hidden\_layer\_sizes=(300,), learning\_rate='adaptive', max\_iter=500)

#DataFlair - Train the model

model.fit(x\_train,y\_train)

#DataFlair - Predict for the test set

y\_pred=model.predict(x\_test)

#DataFlair - Calculate the accuracy of our model

accuracy=accuracy\_score(y\_true=y\_test, y\_pred=y\_pred)

#DataFlair - Print the accuracy

print("Accuracy: {:.2f}%".format(accuracy\*100))

from sklearn.metrics import classification\_report

print(classification\_report(y\_test,y\_pred))

#conda install -c conda-forge jupyterlab

#pip install librosa soundfile numpy sklearn pyaudio

#pipwin install pyaudio

#pip install pipwin