# %% [markdown]

# # Speech Emotion and Gender Recognition

# %% [markdown]

# # Description -

# %% [markdown]

# Speech constitues one of the most popular and significant means for humman to communicate, express their emotions, cognitive states, and intentions to each other.

# %% [markdown]

# Gender recognition is a technique which is utilized to determine the gender category of speaker by processing speech signals.

# %% [markdown]

# # Goal -

# %% [markdown]

# To recognize the human emotions and gender from audio.

# %% [markdown]

# # Step 1 :- Importing the Libraries

# %%

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

import pandas as pd

import numpy as np

import os

import random

import sys

import glob

import librosa

import librosa.display

from tqdm import tqdm

import matplotlib.pyplot as plt

import seaborn as sns

import IPython.display as ipd

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

from sklearn.metrics import accuracy\_score

from sklearn.metrics import classification\_report

from sklearn.metrics import confusion\_matrix

from sklearn import metrics

import matplotlib.pyplot as plt

from sklearn.preprocessing import LabelEncoder

import warnings

warnings.filterwarnings('ignore')

# %% [markdown]

# # Step 2 - Listing Dataset

# %%

os.listdir(path="Ravdass Dataset")

def getListOfFiles(dirName):

listOfFile=os.listdir(dirName)

allFiles=list()

for File in listOfFile:

fullPath=os.path.join(dirName,File)

if os.path.isdir(fullPath):

allFiles=allFiles + getListOfFiles(fullPath)

else:

allFiles.append(fullPath)

return allFiles

dirName = "Ravdass Dataset"

listOfFiles = getListOfFiles(dirName)

val=len(listOfFiles)

#print(listOfFiles)

print(f"The number of audio data files :{val}")

df=pd.DataFrame(listOfFiles, columns=['File'])

print(df.head())

# # Step 4 - Feature Extraction

# %%

def extract\_feature(file\_name):

"""Function Extracts Features from WAV file"""

X, sample\_rate = librosa.load(file\_name)

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

result=np.array([])

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

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

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

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

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)

# %% [markdown]

#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)

# %%

# %% [markdown]

# # Step 3.1 - Data Visualization of Features (Male)

# %%

import IPython.display as ipd

fname= r'Ravdass Dataset\Actor\_01\03-01-01-01-01-01-01.wav';

X, sample\_rate=librosa.load(fname)

plt.figure(figsize=(20,5))

librosa.display.waveshow(X, sr=sample\_rate)

ipd.Audio(fname)

# %%

mfccs = librosa.feature.mfcc(X, sr=sample\_rate, n\_mfcc=40)

print(mfccs.shape)

plt.figure(figsize=(8, 4))

print(f'Sampling rate is {sample\_rate}')

librosa.display.specshow(mfccs,sr=sample\_rate)

plt.colorbar(format='%+2.0f dB')

plt.title('MFCC')

plt.tight\_layout()

plt.xlabel('Time')

plt.ylabel('Hz')

plt.show()

# %%

chroma\_1 = librosa.feature.chroma\_stft(X, sr=sample\_rate)

print(chroma\_1.shape)

plt.figure(figsize=(8, 4))

librosa.display.specshow(chroma\_1, sr=sample\_rate, x\_axis='time')

plt.colorbar(format='%+2.0f dB')

plt.title('Chroma')

plt.tight\_layout()

# %%

mel\_1= librosa.feature.melspectrogram(X, sr=sample\_rate)

print(mel\_1.shape)

plt.figure(figsize=(8, 4))

librosa.display.specshow(mel\_1, sr=sample\_rate, x\_axis='time')

plt.colorbar(format='%+2.0f dB')

plt.title('Mel spectrogram')

plt.tight\_layout()

# %% [markdown]

# # Step 3.2 - Data Visualization of Features (Female)

# %%

import IPython.display as ipd

fname= r'Ravdass Dataset\Actor\_04\03-01-02-02-02-01-04.wav';

X, sample\_rate=librosa.load(fname)

plt.figure(figsize=(20,5))

print(f'Sampling rate is {sample\_rate}')

librosa.display.waveshow(X, sr=sample\_rate)

ipd.Audio(fname)

# %%

mfccs = librosa.feature.mfcc(X, sr=sample\_rate, n\_mfcc=40)

print(mfccs.shape)

plt.figure(figsize=(8, 4))

print(f'Sampling rate is {sample\_rate}')

librosa.display.specshow(mfccs,sr=sample\_rate)

plt.colorbar(format='%+2.0f dB')

plt.title('MFCC')

plt.tight\_layout()

plt.xlabel('Time')

plt.ylabel('Hz')

plt.show()

# %%

chroma\_1 = librosa.feature.chroma\_stft(X, sr=sample\_rate)

print(chroma\_1.shape)

plt.figure(figsize=(8, 4))

librosa.display.specshow(chroma\_1, sr=sample\_rate, x\_axis='time')

plt.colorbar(format='%+2.0f dB')

plt.title('Chroma')

plt.tight\_layout()

# %%

mel\_1= librosa.feature.melspectrogram(X, sr=sample\_rate)

print(mel\_1.shape)

plt.figure(figsize=(8, 4))

librosa.display.specshow(mel\_1, sr=sample\_rate, x\_axis='time')

plt.colorbar(format='%+2.0f dB')

plt.title('Mel spectrogram')

plt.tight\_layout()

# %% [markdown]

def gender(g):

"""Returns Gender Label"""

if int(g[0:2]) % 2 == 0:

return 'female'

else:

return 'male'

# %%

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

"""Loads Data from directory containing WAV files."""

x,y=[],[]

for file in tqdm(glob.glob(r"Ravdass Dataset\Actor\_\\.wav")):

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

male\_female=gender(file\_name.split("-")[-1])

feature=extract\_feature(file)

x.append(feature)

y.append(male\_female)

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

#Split the dataset

x\_train,x\_test,y\_train,y\_test=load\_data(test\_size=0.2)

# %%

#Shape of the training and testing datasets

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

# %%

#Number of features extracted

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

# %% [markdown]

# # Step 6 - Training and Testing the models

# %% [markdown]

# # Model - SVM Model

# %%

from sklearn.svm import SVC

model2 = SVC(C=1,kernel='linear')

model2 = model2.fit(x\_train,y\_train)

model2\_train\_pred = model2.predict(x\_train)

model2\_test\_pred=model2.predict(x\_test)

print("Accuracy score on training data :" ,metrics.accuracy\_score(y\_train,model2\_train\_pred),'\n')

print("Accuracy Score on test data :",metrics.accuracy\_score(y\_test,model2\_test\_pred),'\n')

print('Classfication Report : \n ',classification\_report(y\_test,model2\_test\_pred),'\n')

accuracy=accuracy\_score(y\_true=y\_test, y\_pred=model2\_test\_pred)

print("Accuracy: {:.5f}%".format(accuracy\*100))

# Confusion Matrix

conf\_mat = confusion\_matrix(y\_test,model2\_test\_pred)

x\_axis\_labels=['Female','Male']

y\_axis\_labels=['Female','Male']

sns.heatmap(conf\_mat,annot=True, fmt='',cmap='Blues', xticklabels=x\_axis\_labels , yticklabels=y\_axis\_labels)

plt.xlabel('Actual')

plt.ylabel('Predict')

plt.show()

# %%