Project Report

on

AUDIO SENTIMENT ANALYSIS

Submitted in partial fulfillment for the award of

Post Graduate Diploma in Big Data Analytics

from **C-DAC ACTS (Pune)**

Guided by:

Mr. Prakash Sinha

Presented by:

Mr. Akshay Singh **PRN: 200940183005**

Ms. Khyati Modi **PRN: 200940183021**

Mr. Rahul Khatkar **PRN: 200940183038**

Ms. Sneha Swami **PRN: 200940183052**

Centre of Development of Advanced Computing (C-DAC), Pune

CERTIFICATE

TO WHOMSOEVER IT MAY CONCERN

This is to certify that

Mr. Akshay Singh

Ms. Khyati Modi

Mr. Rahul Khatkar

Ms. Sneha Swami

have successfully completed their project on

Audio Sentiment Analysis

under the guidance of Mr. Prakash Sinha

Project Guide **Project Supervisor**

HOD ACTS

Ms. Mita Karaja

ACKNOWLEDGEMENT

This project "Audio Sentiment Analysis" was a great learning experience for us and we are submitting this work to Advanced Computing Training School (CDAC ACTS).

We all are very glad to mention the name of *Mr.Prakash* for his valuable guidance to work on this project. His guidance and support helped us to overcome various obstacles and intricacies during the course of project work.

We are highly grateful to Ms. Risha P.R. (Manager (ACTS training Centre), C-DAC),for her guidance and support whenever necessary while doing this course Post Graduate Diploma in *Big Data Analytics(PGDBDA)*

through C-DAC ACTS, Pune.

Our most heartfelt thanks goes to *Ms. Seema Sajeevan* (Course Coordinator, *PGDBDA*) who gave all the required support and kind coordination to provide all the necessities like required hardware, internet facility and extra Lab hours to complete the project and throughout the course up to the last day here in C-DAC ACTS, Pune.

From:

Mr.Akshay Singh(200940183005)

Ms.Khyati Modi(200940183021)

Mr. Rahul Khatkar(200940183038)

Ms. Sneha Swami(200940183052)

TABLE OF CONTENT

1. Abstract

Through all the available senses, humans can sense the emotional state of their communication partner. This emotional detection is natural for humans, but it is very difficult task for computers; although they can easily understand content based information, accessing the depth behind content is difficult and that's what speech emotion recognition sets out to do. It is a system through which various audio speech files are classified into different emotions such as happy, sad, anger and neutral by computers. Speech emotion recognition can be used in areas such as the medical field or customer call centers. The foundation of modeling began with feature selection. After extracting MFCCs, Chroma, and Mel spectrograms from the audio files we began assembling models readily available from Sci-kit Learn and other Python packages. The RAVDESS is a validated multimodal database of emotional speech. The database is gender balanced consisting of 24 professional actors. Speech includes calm, happy, sad, angry, fearful, surprise, and disgust expressions contain calm, happy, sad, angry, and fearful emotions.

2. Introduction and Overview of Project

The study of emotion has advanced rapidly over the last decade, driven by low-cost smart technologies and broad interest from researchers in neuroscience, psychology, psychiatry, audiology, and computer science. Integral to these studies is the availability of validated and reliable expressions of emotion. To meet these needs, a growing number of emotion stimulus sets have become available. Most sets contain either static facial expressions or voice recordings. Clinically, there is growing recognition for the role of singing in understanding neurological disorders and facilitating rehabilitation. Yet there are few validated sets of sung emotional expression. To address these needs, we developed the RAVDESS, a large validated set of audiovisual speech and song in North American English. This paper describes the creation of the RAVDESS, and reports validity and reliability data based on ratings from healthy, adult participants.

2.1 Objectives of the project:

i. Analyzing the features of audio files.

ii. Predicting the emotion of the audio files taken as input from test set & microphone.

2.2 Models Used For Prediction:

1. MLP Classifier
2. Sequential NN

2.3 Neural Networks:

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria. The concept of neural networks, which has its roots in artificial intelligent, is swiftly gaining popularity in the development of trending system.

Neural networks are multi-layer networks of neurons (the blue and magenta nodes in the chart below) that we use to classify things, make predictions, etc.

The arrows that connect the dots shows how all the neurons are interconnected and how data travels from the input layer all the way through to the output layer.

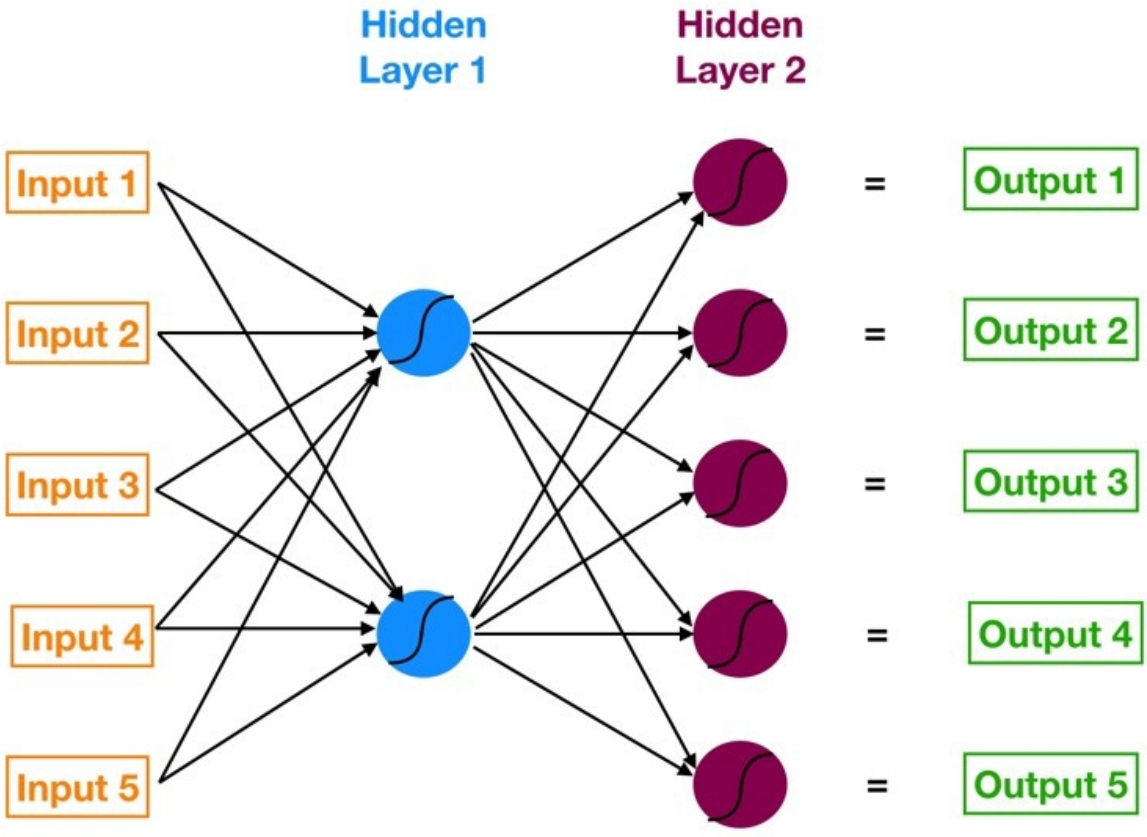
Advantages of Neural Network:

- Neural Networks have the ability to learn by themselves and produce the output that is not limited to the input provided to them.
- The input is stored in its own networks instead of a data base, hence the loss of data does not affect its working.
- These networks can learn from examples and apply them when a similar event arises, making them able to work through real-time events.
- Even if a neuron is not responding or a piece of information is missing, the network can detect the fault and still produce the output.
- They can perform multiple tasks in parallel without affecting the system performance.

1. MLP Classifier

Neural networks are multi-layer networks of neurons (the blue and magenta nodes in the chart below) that we use to classify things, make predictions, etc.

MLP Classifier stands for Multi-layer Perceptron classifier which in the name itself connects to a Neural Network. Unlike other classification algorithms such as Support Vectors or Naive Bayes Classifier, MLP Classifier relies on an underlying Neural Network to perform the task of classification. It helps to convert the input into a more useful output. Sigmoid activation function creates an output with values between 0 and 1. There can be other activation functions like Tanh, softmax and RELU



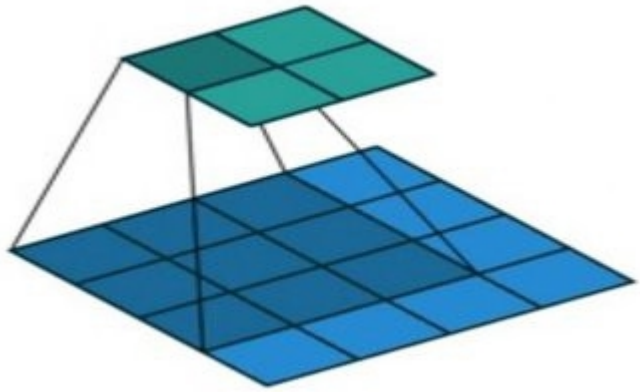
Starting from the left, we have:

1. The input layer of our model in orange.
2. Our first hidden layer of neurons in blue.
3. Our second hidden layer of neurons in magenta.
4. The output layer (a.k.a. the prediction) of our model in green.

The arrows that connect the dots shows how all the neurons are interconnected and how data travels from the input layer all the way through to the output layer.

2. Sequential CNN

A great way to use deep learning to classify images is to build a convolutional neural network (CNN). The Keras library in Python makes it pretty simple to build a CNN.A convolution multiplies a matrix of pixels with a filter matrix or 'kernel' and sums up the multiplication values. Then the convolution slides over to the next pixel and repeats the same process until all the image pixels have been covered. This process is visualized below.



CNN process

Advantages of CNN:

- CNN learns the filters automatically without mentioning it explicitly. These filters help in extracting the right and relevant features from the input data.
- CNN captures the spatial features from an image. Spatial features refer to the arrangement of pixels and the relationship between them in an image. They help us in identifying the object accurately, the location of an object, as well as its relation with other objects in an image.
- CNN also follows the concept of parameter sharing. A single filter is applied across different parts of an input to produce a feature map.

3. Data Description

Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) This RAVDESS dataset contains 1440 files: 60 trials per actor x 24 actors = 1440. The RAVDESS contains 24 professional actors (12 female, 12 male), vocalizing two lexically-matched statements in a neutral North American accent. Speech emotions include calm, happy, sad, angry, fearful, surprise, and disgust expressions. Each expression is produced at two levels of emotional intensity (normal, strong), with an additional neutral expression.

Emotion	Speech Count	Song Count	Total Count
Neutral	96	92	188
Calm	192	184	376
Happy	192	184	376
Sad	192	184	376
Angry	192	184	376
Fearful	192	184	376
Disgust	192	0	192
Surprised	192	0	192
Total	1440	1012	2452

File naming convention:

Each of the 1440 files has a unique filename. The filename consists of a 7-part numerical identifier (e.g., 03-01-06-01-02-01-12.wav). These identifiers define the stimulus characteristics:

Filename identifiers

- Modality (01 = full-AV, 02 = video-only, 03 = audio-only).
- Vocal channel (01 = speech, 02 = song).
- Emotion (01 = neutral, 02 = calm, 03 = happy, 04 = sad, 05 = angry, 06 = fearful, 07 = disgust, 08 = surprised).
- Emotional intensity (01 = normal, 02 = strong). NOTE: There is no strong intensity for the 'neutral' emotion.
- Statement (01 = "Kids are talking by the door", 02 = "Dogs are sitting by the door").

- Repetition (01 = 1st repetition, 02 = 2nd repetition).
- Actor (01 to 24. Odd numbered actors are male, even numbered actors are female).

Filename example: ***03-01-06-01-02-01-12.wav***

1. Audio-only (03)
2. Speech (01)
3. Fearful (06)
4. Normal intensity (01)
5. Statement "dogs" (02)
6. 1st Repetition (01)
7. 12th Actor (12)

Female, as the actor ID number is even.

The files are in the WAV raw audio file format and all have a 16 bit Bitrate and a 48 kHz sample rate. The files are all uncompressed, lossless audio, meaning that the audio files in the dataset have not lost any information/data or been modified from the original recording.

As mentioned before, to process/manipulate these files we used the libROSA python package. This package was originally created for music and audio analysis, making it the perfect selection for dealing with our dataset.

After importing libROSA, we read in one WAV file at a time. An audio time series in the form of a 1-dimensional array for mono or 2-dimensional array for stereo, along with time sampling rate (which defines the length of the array), where the elements within each of the arrays represent the amplitude of the sound waves is returned by libROSA's "load" function.

4. Evaluation Metrics

4.1 Accuracy:

Accuracy is one metric for evaluating classification models. Informally, **accuracy** is the fraction of predictions our model got right. Formally, accuracy has the following definition:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

For binary classification, accuracy can also be calculated in terms of positives and negatives as follows:

$$\text{Accuracy}=\text{TP}+\text{TNTP}+\text{TN}+\text{FP}+\text{FN}$$

Where *TP* = True Positives, *TN* = True Negatives, *FP* = False Positives, and *FN* = False Negatives.

4.2 Precision and Recall

Precision is a good measure to determine, when the costs of False Positive is high. For instance, email spam detection. In email spam detection, a false positive means that an email that is non-spam (actual negative) has been identified as spam (predicted spam). The email user might lose important emails if the precision is not high for the spam detection model.

In the field of information retrieval precision is the fraction of retrieved documents that are relevant to the query:

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$= \frac{True\ Positive}{Total\ Predicted\ Positive}$$

Recall

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

$$= \frac{True\ Positive}{Total\ Actual\ Positive}$$

Recall calculates how many of the Actual Positives our model capture through labeling it as Positive (True Positive). Applying the same understanding, we know that Recall shall be the model metric we use to select our best model when there is a high cost associated with False Negative.

4.3 F1 Score

F1 is a function of Precision and Recall.

$$F1 = 2 \times \frac{Precision * Recall}{Precision + Recall}$$

F1 Score might be a better measure to use if we need to seek a balance between Precision and Recall AND there is an uneven class distribution (large number of Actual Negatives).

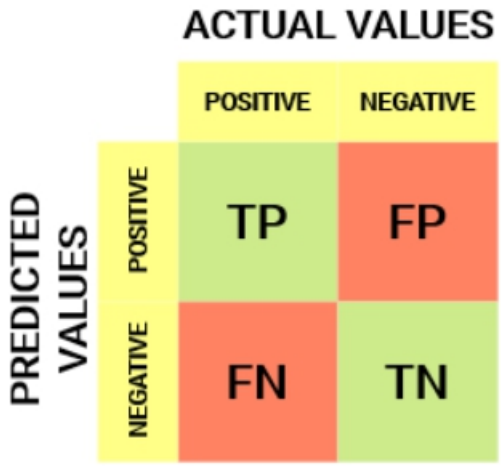
4.4 Support

Support is the number of actual occurrences of the class in the specified data set.Imbalanced support in the training data may indicate structural weaknesses in the reported scores of the classifier and could indicate the need for stratified sampling or re-balancing.

4.5 Confusion matrix

A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. This gives us a holistic view of how well our classification model is performing and what kinds of errors it is making.

For a binary classification problem, we would have a 2 x 2 matrix as shown below with 4 values:



Let's decipher the matrix:

- The target variable has two values: **Positive** or **Negative**
- The **columns** represent the **actual values** of the target variable
- The **rows** represent the **predicted values** of the target variable

5. Data Preprocessing

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

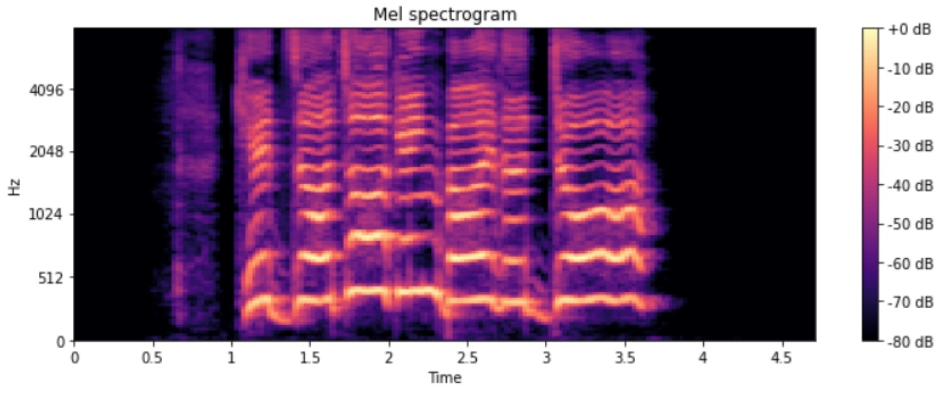
Before going into pre-processing and data exploration we will explain some of the concepts that allowed us to select our features.

5.1 Feature Extraction

5.1.1 Mel scale :- deals with human perception of frequency, it is a scale of pitches judged by listeners to be equal distance from each other

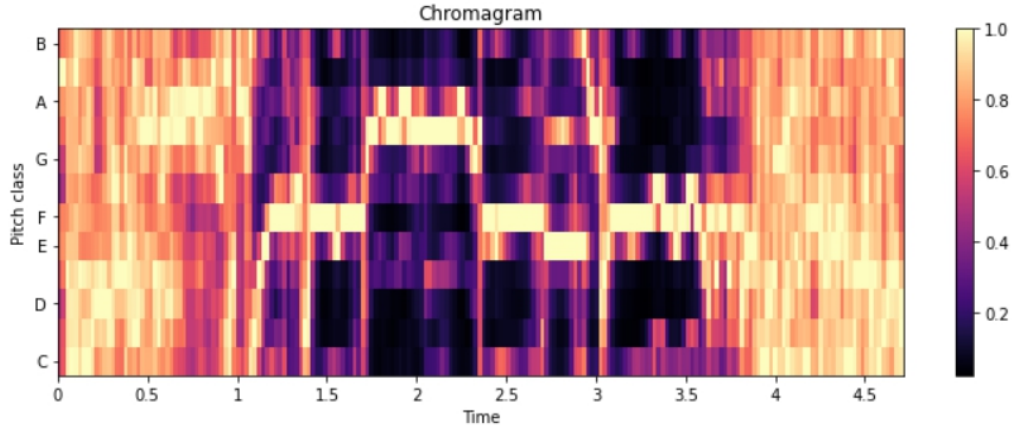
magnitude spectrogram computed then mapped onto mel scale x-axis is time, y-axis is frequency

```
In [375]: S = librosa.feature.melspectrogram(y=x, sr=sr, n_mels=128,fmax=8000)
          plt.figure(figsize=(10, 4))
          librosa.display.specshow(librosa.power_to_db(S,ref=np.max),y_axis='mel', fmax=8000,x_axis='time')
          plt.colorbar(format='%+2.0f dB')
          plt.title('Mel spectrogram')
          plt.tight_layout()
```



5.1.2 Chroma :-Representation for audio where spectrum is projected onto 12 bins representing the 12 distinct semitones (or chroma). Computed by summing the log frequency magnitude spectrum across octaves.

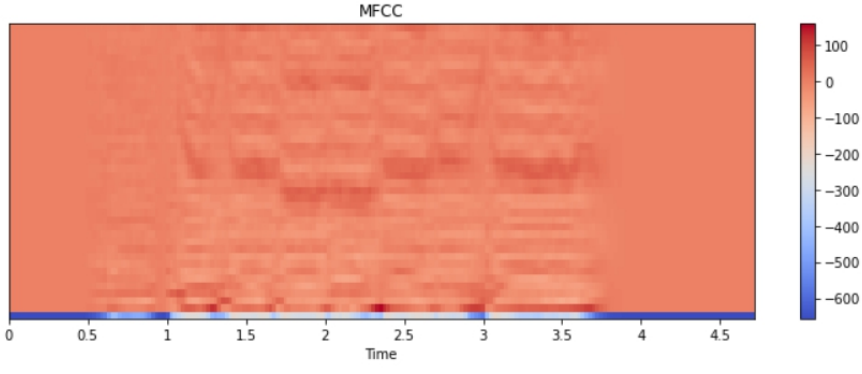
In [378]:
```
x,sr=librosa.load(Ravdess_df.Path[0])
S = np.abs(librosa.stft(x))
chroma = librosa.feature.chroma_stft(S=S, sr=sr)
plt.figure(figsize=(10, 4))
librosa.display.specshow(chroma, y_axis='chroma', x_axis='time')
plt.colorbar()
plt.title('Chromagram')
plt.tight_layout()
```



5.1.3 Mfcc :-Mel frequency Cepstral coefficients algorithm is a technique which takes voice sample as inputs. After processing, it calculates coefficients unique to a particular sample

- Voice is dependent on the shape of vocal tract including tongue, teeth, etc.
- Representation of short-time power spectrum of sound, essentially a representation of the vocal tract

In [377]:
```
mfccs = librosa.feature.mfcc(y=x, sr=sr, n_mfcc=40)
plt.figure(figsize=(10, 4))
librosa.display.specshow(mfccs, x_axis='time')
plt.colorbar()
plt.title('MFCC')
plt.tight_layout()
```



5.2 Exploratory Data Analysis :- refers to the critical process of performing initial investigations on data so as to discover patterns, to spot nomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.

5.2.1 Imports and Read In Data

In [364]:
```
import warnings
warnings.filterwarnings("ignore")
import os
import numpy as np
import pandas as pd


import matplotlib.pyplot as plt
import seaborn as sns

import librosa as lr

import librosa
import speech_recognition as sr

import IPython.display as ipd
import librosa.display
```

5.2.2 loading dataset

In [365]:
```
# loading audio files of 24 actors and their respective path from Ravdess Dataset with path
import os
os.listdir(path='Sonic_Sark/Dataset')

def getListOfFiles(dirName):
    listOfFile=os.listdir(dirName)
    allFiles=list()
    for entry in listOfFile:
        fullPath=os.path.join(dirName, entry)
        if os.path.isdir(fullPath):
            allFiles=allFiles + getListOfFiles(fullPath)
        else:
            allFiles.append(fullPath)
    return allFiles

dirName = 'Sonic_Sark/Dataset'
listOfFiles = getListOfFiles(dirName)
print("Total Number of Audio Files is ",len(listOfFiles))
```

```
Total Number of Audio Files is  2452
```

5.2.3 Separating & Labelling Emotions from Dataset

```
In [368]: import pandas as pd
          file_emotion = []
          file_path = []

          for i in listOfFiles:
                  part = i.split('.')[0]
                  part = part.split('-')
                  file_emotion.append(int(part[2]))
                  file_path.append(i)

          # Storing 3rd part of every file into emotion_df dataframe
          emotion_df = pd.DataFrame(file_emotion, columns=['Emotions'])

          # Concatenating file_emotion and file_path into Ravdess_df
          path_df = pd.DataFrame(file_path, columns=['Path'])
          Ravdess_df = pd.concat([emotion_df, path_df], axis=1)

          # Labeling integers into their respective classes
          Ravdess_df.Emotions.replace({1:'neutral', 2:'calm', 3:'happy', 4:'sad', 5:'angry', 6:'fear', 7:'disgust', 8:'surprise'}, inplace=
```

5.3 EDA

It is a way of visualizing, summarizing and interpreting the information that is hidden in rows and column format. EDA is one of the crucial step in data science that allows us to achieve certain insights and statistical measure that is essential for the business continuity, stockholders and data scientists. It performs to define and refine our important features.

1. Handle Missing value
2. Removing duplicates
3. Outlier Treatment
4. Normalizing and Scaling(Numerical Variables)
5. Encoding Categorical variables(Dummy Variables)
6. Bivariate Analysis

5.3.1 Variable identification and data types

The very first step in exploratory data analysis is to identify the type of variables in the dataset. Variables are of two types Numerical and Categorical. dtypes method to identify the data type of the variables in the dataset .

```
In [386]: df.dtypes

Out[386]: 0       object
          0.1    float64
          1      float64
          2      float64
          3      float64
                  ...
          175    float64
          176    float64
          177    float64
          178    float64
          179    float64
          Length: 181, dtype: object
```

5.3.2 Size of the dataset

We can get the size of the dataset using the shape method.

```
In [388]: df.shape
Out[388]: (2457, 181)
```

5.3.3 To get the list and number of unique values

the unique() function of pandas returns the list of unique values in the dataset.

```
In [387]: df['0'].unique(), df['0'].nunique()
Out[387]: (array(['angry', 'fear', 'happy', 'sad', 'calm', 'neutral', 'surprise',
                  'disgust'], dtype=object),
           8)
```

5.3.4 Finding null values

When we import our dataset from a CSV file, many blank columns are imported as null values into the Data Frame, which can later create problems while operating that data frame. Pandas isnull() method is used to check and manage NULL values in a data frame.

```
In [392]: df.apply(lambda x: sum(x.isnull()),axis=0)
Out[392]: 0      0
          0.1    0
          1      0
          2      0
          3      0
                ..
          175    0
          176    0
          177    0
          178    0
          179    0
          Length: 181, dtype: int64
```

5.3.5 describe the dataset

Describe() function to get various summary statistics that exclude NaN values.this function returns the count, mean, standard deviation, minimum and maximum values and the quantiles of the data.

```
In [389]: df.describe()
```

Out[389]:

	0.1	1	2	3	4	5	6	7	8	9	...	170
count	2457.000000	2457.000000	2457.000000	2457.000000	2457.000000	2457.000000	2457.000000	2457.000000	2457.000000	2457.000000	...	2.457000e+03
mean	0.619268	0.595147	0.587892	0.610196	0.628515	0.658210	0.615478	0.601259	0.617101	0.616681	...	8.211475e-04
std	0.087860	0.109573	0.120182	0.111091	0.086690	0.085072	0.077030	0.102370	0.098979	0.094794	...	2.079504e-03
min	0.306251	0.223043	0.215276	0.253819	0.362172	0.364283	0.284953	0.241488	0.255482	0.306462	...	7.245407e-08
25%	0.564828	0.518404	0.499749	0.528754	0.569516	0.613475	0.564903	0.531092	0.549741	0.550432	...	3.206290e-05
50%	0.627198	0.597895	0.590556	0.614679	0.640792	0.670442	0.624019	0.605853	0.622478	0.622195	...	1.664589e-04
75%	0.682820	0.678445	0.682986	0.697490	0.692547	0.719197	0.671030	0.681293	0.695971	0.692210	...	6.814696e-04
max	0.836730	0.860695	0.859838	0.866144	0.845397	0.846784	0.827082	0.840626	0.848023	0.836951	...	2.890956e-02

8 rows × 180 columns

5.4 Data preprocessing

5.4.1 Seperating target variable and feature variable

After execution of this code, the independent variable *X* and dependent variable *Y* will transform into the following.

```
In [39]: X = df.iloc[: ,1:].values
         y=df["0"].values
```

5.4.2 Splitting X and y into train and test set

Any machine learning algorithm needs to be tested for accuracy. In order to do that, we divide our data set into two parts: **training set** and **testing set.** As the name itself suggests, we use the training set to make the algorithm learn the behaviours present in the data and check the correctness of the algorithm by testing on testing set.

```
In [40]: from sklearn.model_selection import train_test_split
         x_train, x_test, y_train, y_test = train_test_split(X, y, random_state=9,test_size=0.10)# shuffle=True
```

6. Model Building

The modeling process was divided into two main parts: traditional machine learning models and deep neural networks. Simpler models were to be used as a baseline for the convolutional neural network and recurrent neural network.

1.Multi Layer Perceptron Classifier (MLP)

we will build the Multi-layer Perceptron classifier

```
In [43]: # Initializing the Multi Layer Perceptron Classifier
         from sklearn.neural_network import MLPClassifier
         model=MLPClassifier(alpha=0.01, batch_size=16, epsilon=1e-08, hidden_layer_sizes=(500,), learning_rate='adaptive', max_iter=500)
```

```
In [44]: # Train the model
         model.fit(x_train,y_train)

Out[44]: MLPClassifier(alpha=0.01, batch_size=16, hidden_layer_sizes=(500,),
                       learning_rate='adaptive', max_iter=500)
```

- hidden_layer_sizes : This parameter allows us to set the number of layers and the number of nodes we wish to have in the Neural Network Classifier. Each element in the tuple represents the number of nodes at the ith position where i is the index of the tuple. Thus the length of tuple denotes the total number of hidden layers in the network.
- max_iter: It denotes the number of epochs.
- alpha: by default 0.001,L2 penalty (regularization term) parameter.
- Learning_rate:This model optimizes the log-loss function using LBFGS or stochastic gradient descent.
- Epsilon:This makes sure that the loss function is not heavily influenced by the outliers while not completely ignoring their effect.

After initializing we can now give the data to train the Neural Network.

Using the trained network to predict

```
In [45]: # Predict for the test set
         y_pred=model.predict(x_test)
```

Calculating the accuracy of predictions

In [15]:
```
# Calculate the accuracy of our model
from sklearn.metrics import accuracy_score
accuracy=accuracy_score(y_true=y_test, y_pred=y_pred)

# Print the accuracy
print("Accuracy: {:.2f}%".format(accuracy*100))
```

```
Accuracy: 72.56%
```

Classification report

In [47]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

       angry       0.92      0.80      0.86        41
        calm       0.94      0.83      0.88        41
     disgust       0.73      0.62      0.67        13
        fear       0.77      0.90      0.83        30
       happy       0.89      0.93      0.91        44
     neutral       0.86      0.83      0.84        29
         sad       0.76      0.78      0.77        36
    surprise       0.65      0.92      0.76        12

    accuracy                           0.84       246
   macro avg       0.81      0.83      0.81       246
weighted avg       0.85      0.84      0.84       246
```

Confusion matrix for emotions prediction on RAVDESS with Accuracy(72.56%) and each row indicated the confusion of each emotion with ground truth and predictions.

Python pickle module is used for serializing and de-serializing python object structures. The process to converts any kind of python objects (list, dict, etc.) into byte streams (0s and 1s) is called pickling or serialization or flattening or marshalling

In [50]:
```
# SAVING THE MODEL
# Saving the Model to file in the current working directory

import pickle
Pkl_Filename = "new_mlp_74.pkl"
with open(Pkl_Filename, 'wb') as file:
    pickle.dump(model, file)
```

On Microphone data

In [237]:
```
import librosa
import speech_recognition as sr

# obtain audio from the microphone
r = sr.Recognizer()
with sr.Microphone() as source:
    print("Hiii SARK's Say something!")
    audio = r.listen(source,timeout=1,phrase_time_limit=4)

# write audio to a WAV file
with open("output1.wav", "wb") as f:
    f.write(audio.get_wav_data())
```

```
Hiii SARK's Say something!
```

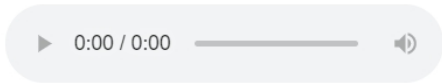
In [238]:
```
#Audio to text

txt=sr.AudioFile("output1.wav")

with txt as source:
    audio = r.record(source)
try:
    s = r.recognize_google(audio)
    print("You Said  : "+s)
except Exception as e:
    print("Exception: "+str(e))

ipd.display(ipd.Audio('output1.wav'))
```

```
You Said  : hi Siri where are you I am fine I will miss you
```

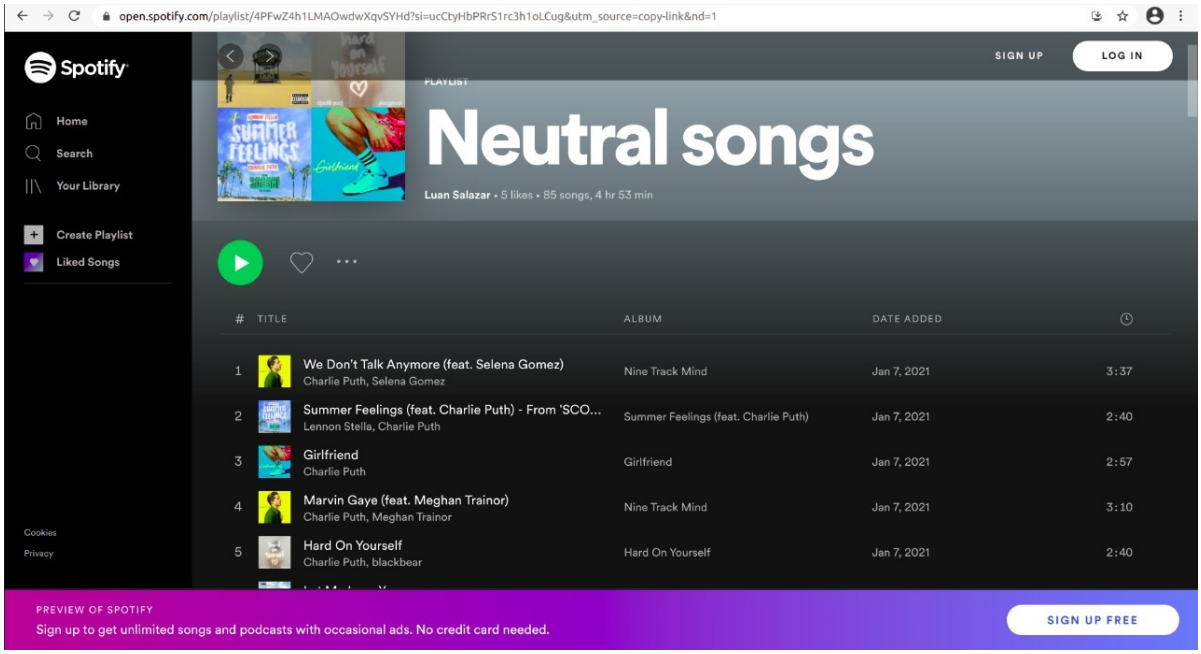


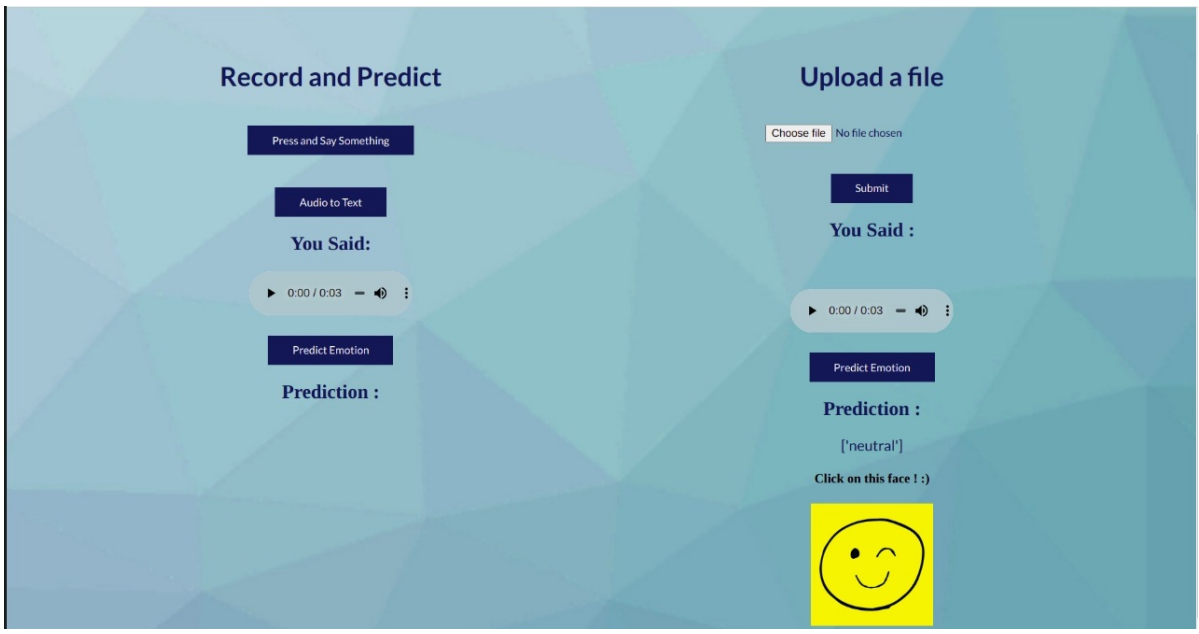
Result on Microphone data

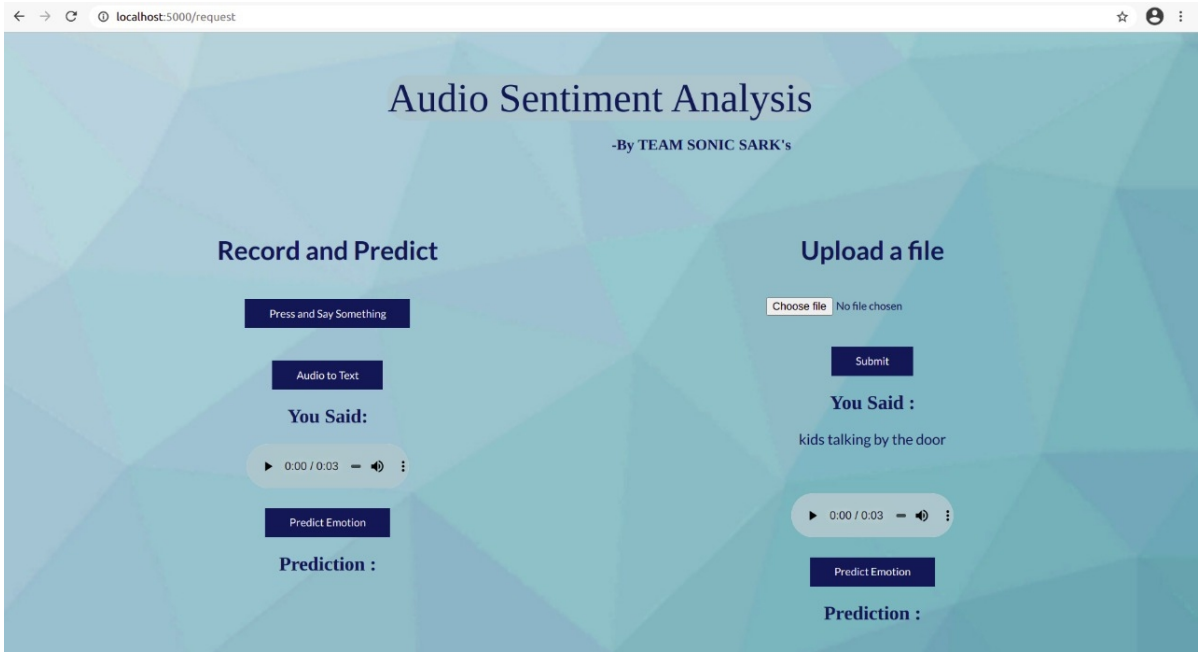
In [244]:
```
model.predict(x)
```

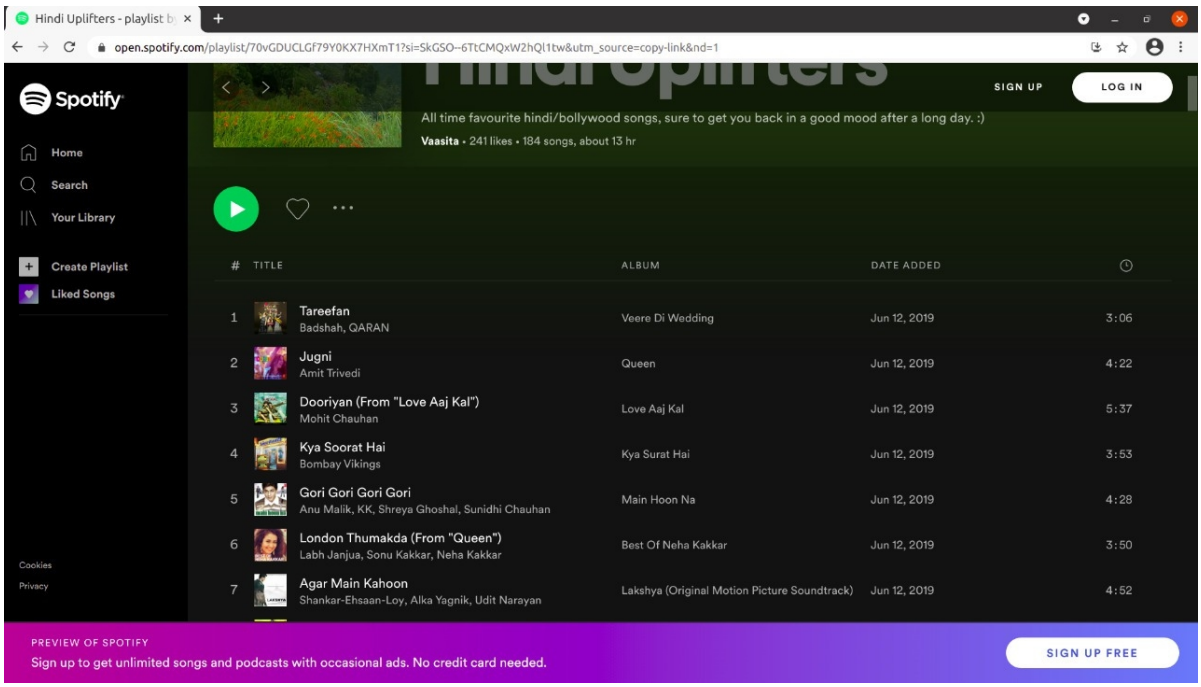
Out[244]:
```
array(['calm'], dtype='<U8')
```

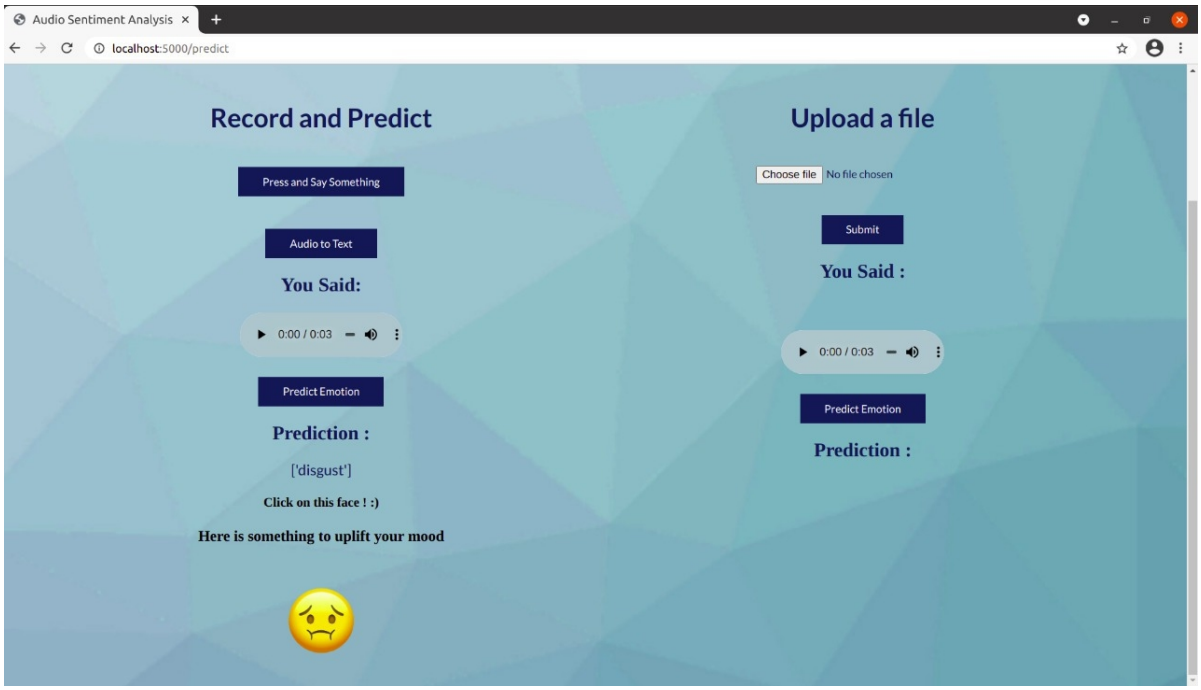
7. Prediction

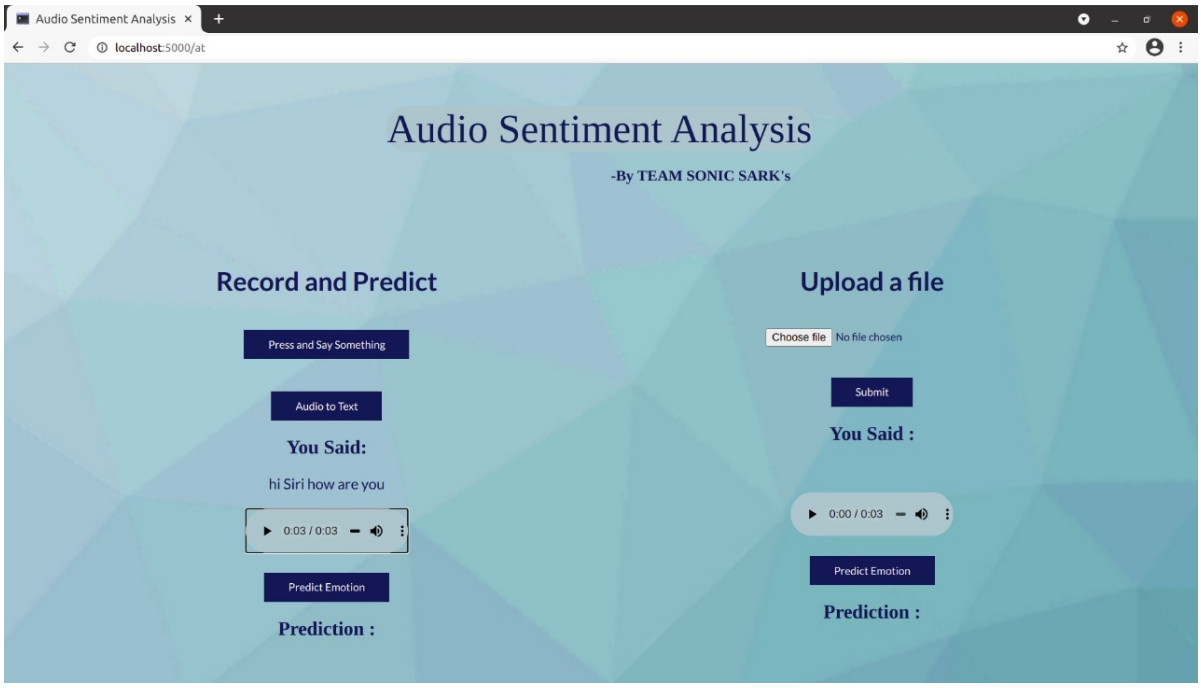





localhost:5000/request
Audio Sentiment Analysis
-By TEAM SONIC SARK's
Record and Predict
Press and Say Something
Audio to Text
You Said:
0:00 / 0:03
Predict Emotion
Prediction :
Upload a file
Choose file
No file chosen
Submit
You Said :
kids talking by the door
0:00 / 0:03
Predict Emotion
Prediction :


Hindi Uplifters - playlist
open.spotify.com/playlist/70vGDUCLGf79Y0KX7HXmT1?si=SkGSO--6TtCMQxW2hQl1tw&utm_source=copy-link&nd=1
Spotify
Home
Search
Your Library
Create Playlist
Liked Songs
Cookies
Privacy
SIGN UP
LOG IN
All time favourite hindi/bollywood songs, sure to get you back in a good mood after a long day. :)
Vaasita • 241 likes • 184 songs, about 13 hr
TITLE
ALBUM
DATE ADDED
1 Tareefan Badshah, QARAN Veere Di Wedding Jun 12, 2019 3:06
2 Jugni Amit Trivedi Queen Jun 12, 2019 4:22
3 Dooriyan (From "Love Aaj Kal") Mohit Chauhan Love Aaj Kal Jun 12, 2019 5:37
4 Kya Soorat Hai Bombay Vikings Kya Surat Hai Jun 12, 2019 3:53
5 Gori Gori Gori Gori Anu Malik, KK, Shreya Ghoshal, Sunidhi Chauhan Main Hoon Na Jun 12, 2019 4:28
6 London Thumakda (From "Queen") Labh Janjua, Sonu Kakkar, Neha Kakkar Best Of Neha Kakkar Jun 12, 2019 3:50
7 Agar Main Kahoon Shankar-Ehsaan-Loy, Alka Yagnik, Udit Narayan Lakshya (Original Motion Picture Soundtrack) Jun 12, 2019 4:52
PREVIEW OF SPOTIFY
Sign up to get unlimited songs and podcasts with occasional ads. No credit card needed.
SIGN UP FREE


Audio Sentiment Analysis
localhost:5000/predict
Record and Predict
Press and Say Something
Audio to Text
You Said:
0:00 / 0:03
Predict Emotion
Prediction :
['disgust']
Click on this face ! :)
Here is something to uplift your mood
Upload a file
Choose file
No file chosen
Submit
You Said :
0:00 / 0:03
Predict Emotion
Prediction :


Audio Sentiment Analysis
localhost:5000/at
Audio Sentiment Analysis
-By TEAM SONIC SARK's
Record and Predict
Press and Say Something
Audio to Text
You Said:
hi Siri how are you
0:03 / 0:03
Predict Emotion
Prediction :
Upload a file
Choose file
No file chosen
Submit
You Said :
0:00 / 0:03
Predict Emotion
Prediction :

8.Conclusion

Machine learning (ML) methods has recently contributed very well in the advancement of the prediction models used for energy consumption. Such models highly improve the accuracy, robustness, and precision and the generalization ability of the conventional time series forecasting tools. This project demonstrates how we can leverage the Neural Networks to obtain the underlying emotion from speech audio data and some insights on the human expression of emotion through voice. This system can be employed in a variety of setups like Call Centre for complaints or marketing, in voice-based virtual assistants or chatbots, in linguistic research, etc.

This project analysis the speech audio data using MLP classifier for emotions prediction on RAVDESS with Accuracy(72.56%) and each row indicated the confusion of each emotion with ground truth and predictions.The accuracy of the model can be increased by including more audio files for training.

9.Future scope

- An accurate implementation of the pace of the speaking can be explored to check if it can resolve some of the deficiencies of the model.
- Exploring other acoustic features of sound data to check their applicability in the domain of speech emotion recognition. These features could simply be some proposed extensions of MFCC like RAS-MFCC or they could be other features entirely like LPCC, PLP or Harmonic cepstrum.
- Figuring out a way to clear random silence from the audio clip.
- An alternate approach that could be explored for this problem is splitting the classifying task into two distinct problems. A separate model could be used to classify gender and then separate models for each gender to classify emotion could be utilized. This could possibly lead to a performance improvement by segregating the task of emotion classification by gender.
- Adding more data volume either by other augmentation techniques like time-shifting or speeding up/slowing down the audio or simply finding more annotated audio clips

10. Bibliography

1. Dataset

https://www.kaggle.com/uwrfkaggler/ravdess-emotional-speech-audio

2. Librosa Documentation

https://librosa.org/doc/latest/index.html

3. Resources

https://towardsdatascience.com/nns-aynk-c34efe37f15a

https://towardsdatascience.com/speech-emotion-recognition-with-convolution-neural-network-1e6bb7130ce3

4. Speech recognition

https://pypi.org/project/SpeechRecognition/

5. Flask tutorial

https://www.tutorialspoint.com/flask/index.html