MINI-PROJECT

ON

EMOTION DETECTION USING SPEECH

By

B.Yellareswari (N150055)

P.Sravani (N150207)

M.Sirisha (N150535)

M.Padmini (N151158)

Under the guidance of

Ms.Ch.Lakshimi Bala

Faculty, Dept of Computer Science

RGUKT- IIIT, Nuzvid

A Project Work submitted to the Department of CSE for Mini project of E3,Sem1.

Rajiv Gandhi University of Knowledge Technologies, A.P. IIIT

Nuzvid - 521201

Rajiv Gandhi University of Knowledge Technologies

APIIIT-Nuzvid

Department of CSE

CERTIFICATE OF PROJECT COMPLETION

This is to certify that the project work entitled "Emotion detection using speech " by N150055, N150535, N150207, and N151158 submitted for E3 sem1 mini project to the department of CSE under my guidance during the academic year 2019 -2020. This project in my opinion, is worthy of consideration for the awarding of marks in accordance with the Department and University regulations.

Date:

Place:

R.Upendar Rao
Head of the Department
Department of Computer Science and Engineering

Supervisor

Ms.Ch.Lakshimi Bala
Faculty
Department of CSE

November 2019

ABSTRACT :

Recognizing emotion from speech has become one the active research themes in speech processing and in applications based on human-computer interaction. This paper conducts an experimental study on recognizing emotions from human speech. The emotions considered for the experiments include neutral,calm,happy,fearful,disgust,surprised.
The distinuishability of emotional features in speech were studied first followed by emotion classification performed on a custom dataset.After performing the classification tests on a dataset formed from different subjects, it was found that for getting better accuracy, one should consider the data collected from one person rather than considering the data from a group of people.

Speech emotion recognition, the best ever python mini project. The best example of it can be seen at call centers. If you ever noticed, call centers employees never talk in the same manner, their way of pitching/talking to the customers changes with customers. Now, this does happen with common people too, but how is this relevant to call centers? Here is your answer, the employees recognize customers' emotions from speech, so they can improve their service and convert more people. In this way, they are using speech emotion recognition.

Speech Emotion Recognition, abbreviated as SER, is the act of attempting to recognize human emotion and affective states from speech. This is capitalizing on the fact that voice often reflects underlying emotion through tone and pitch. This is also the phenomenon that animals like dogs and horses employ to be able to understand human emotion.

Although emotion detection from speech is a relatively new field of research, it has many potential applications. In human-computer or human-human interaction systems, emotion recognition systems could provide users with improved services by being adaptive to their emotions. In virtual worlds, emotion recognition could help simulate more realistic avatar interaction.

The body of work on detecting emotion in speech is quite limited. Currently, researchers are still debating what features influence the recognition of emotion in speech. There is also considerable uncertainty as to the best algorithm for classifying emotion, and which emotions to class together.

In this project, we attempt to address these issues. We use MLP classifier to classify emotions. We saved the speech of different emotions in audio files .

There are a variety of temporal and spectral features that can be extracted from human speech. We use statistics relating to the pitch, Mel Frequency Cepstral Coefficients (MFCCs) and Formants of speech as inputs to classification algorithms. The emotion recognition accuracy of these experiments allow us to explain which features carry the most emotional information.

It also allows us to develop criteria to class emotions together. Using these techniques we are able to achieve high emotion recognition accuracy.

EXISTING AND PROPOSAL SYSTEMS :

→ EXISTING SYSTEM:

Earlier there are several projects on Emotion detection .The used to detect the Emotion based on images, facial expressions and speech also.
They used algorithms like SVM,K means clustering etc.

→ DRAWBACKS OF EXISTING SYSTEM:

- It doesn’t produce effective results.
- Time complexity for these systems is also high.
- Using SVM it is hard for larger datasets and if it has noise data it's not possible to get accurate results.

→PROPOSING SYSTEM:

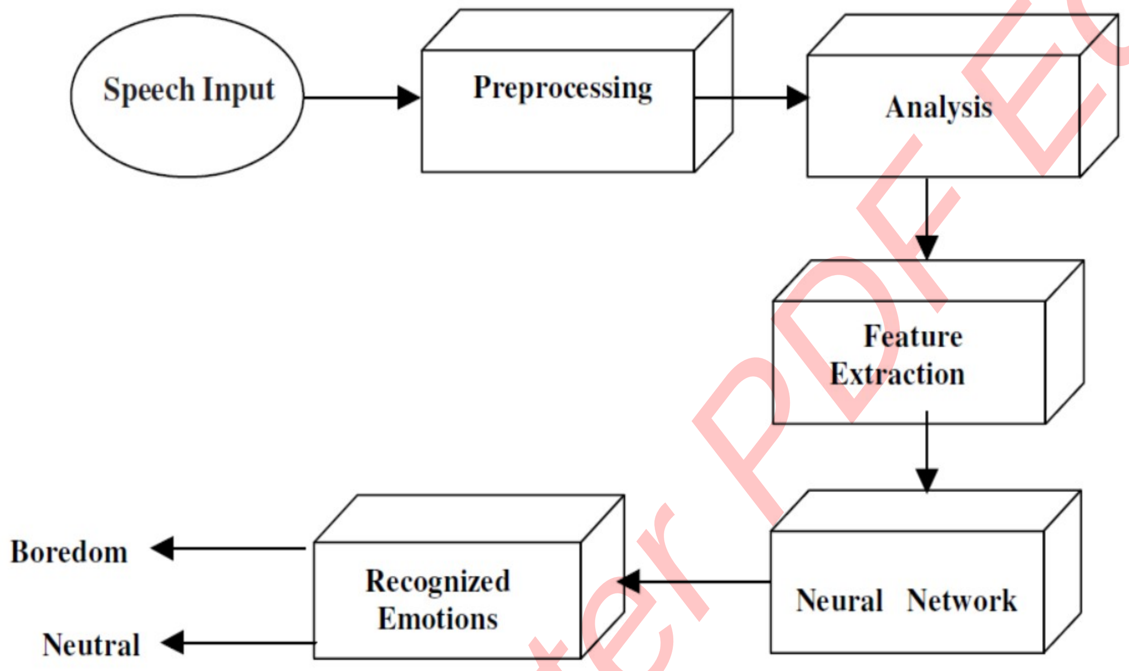


Fig. 1—Structure of emotion recognition system

→ADVANTAGES OF PROPOSING SYSTEM:

-Easily implemented.

- We used MLP classifier so that the classification of Emotions are easy. And the time complexity is less.

-With large number of dataset MLP classifier may be computationally faster than other approaches.

- MLP classifier may produce tighter clusters than hierarchical clustering.

REQUIREMENTS:

Software requirements:

1.Jupyter Lab

2.The datase having different emotions in audio format.

Hardware requirements:

1. PC with minimum4GB RAM and 64 bit processor.

MODULES :

Two of the major components of an emotional speech recognition system are feature extraction and classification.

3.1 Feature Extraction :

Features represent the characteristics of a human vocal tract and hearing system. As it is a complex system, efficient feature extraction is a challenging task in emotion recognition system. Extracting suitable features is one of the main aspects of the emotion recognition system. Linear prediction coefficients (LPCs) [6, 7] are one of the most used features for both speech and emotional recognition. Basic idea behind the LPC model is that given speech sample at time n, s(n) can be approximated as a linear combination of the past p speech samples. A LP model can be represented mathematically

$$e(n) = s(n) - \sum_{k=1}^{p} a_k s(n-k)$$

The error signal e(n) is the difference between the input speech and the estimated speech. The filter coefficients ak are called the LP (linear prediction) coefficients. One of the most widely used prosodic features for speech emotion is MFCC [8,9] which outperformed LPC in classification of speech and emotions due to use of Mel frequency which is linear at low frequency and logarithmic at high frequency to suit the human hearing system.

The Mel scale is represented by the following equation

$$\mathrm{Mel}(f) = 2595 \log_{10}\left(1 + \frac{f}{700}\right)$$

where f is the frequency of the signal.

Linear prediction cepstral coefficients (LPCC) [10] use all the steps of LPC.The LPC coefficients are converted into cepstral coefficients using the following algorithm.

$$\mathrm{LPCC} = \mathrm{LPC}_i + \sum_{k=1}^{i-1}\left(\frac{k-i}{i}\right)\mathrm{LPCC}_{i-k}\mathrm{LPC}_k$$

PLP [11] uses three concepts from the psychophysics of hearing to derive an estimate of the auditory spectrum: (1) the critical-bands spectral resolution, (2) the equal-loudness curve and (3) the intensity-loudness power law. The auditory spectrum is then approximated by an autoregressive all-pole model. A fifth-order all-pole model is effective in suppressing speaker-dependent details of the auditory spectrum. In comparison with conventional LP analysis, PLP analysis is more consistent with human hearing.

The spectrum P(ω) of the original emotional speech signal is wrapped along its frequency axis ω into the Bark frequency Ω by

$$\Omega(\omega) = 6 \ln \left[\frac{\omega}{1200\pi} + \left[\left(\frac{\omega}{1200\pi} \right)^2 + 1 \right]^{0.5} \right]$$

where ω is the angular frequency in rad/s.

3.2 Emotion Classification :

In this paper, multilayer perception (MLP) [12, 13] classifier is used and the results with different features are compared.

The structure of MLP for three layers is shown in Fig. 1. The three layers are input layer, hidden layer and output layer. Let each layer has its own index variable, 'k' for output nodes, 'j' for hidden nodes and 'i' for input nodes. The input vector is propagated through a weight layer V. The output of jth hidden node is given by,

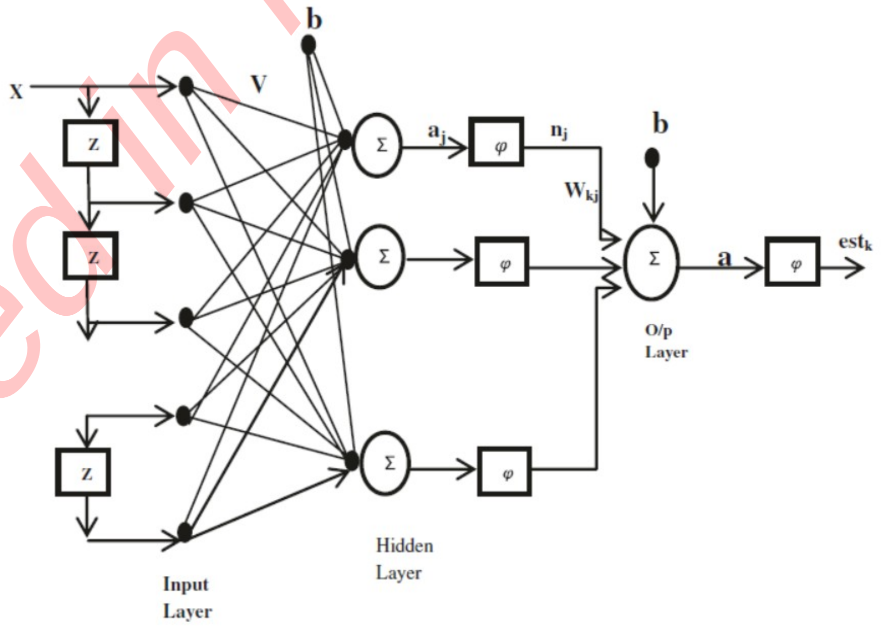


Fig. 1 Structure of MLP

$$n_j = \varphi\left(a_j(t)\right)$$

$$\text{where } a_j(t) = \sum_i x_i(t) v_{ji} + b_j$$

and aj is output of jth hidden node before activation. xi is the input value at ith node, bj is the bias for jth hidden node, and ϕ is the activation function.The output of the MLP network is determined by a set of output weights, W, and is computed as

$$\text{est}_k(t) = \varphi(a_k(t))$$

$$a_k(t) = \sum_j n_j(t) w_{kj} + b_k$$

where η is the learning rate parameter of the back-propagation algorithm.
where estk is the final estimated output of kth output node. The learning algorithm used in training the weights is back-propagation. In this algorithm, the correction to the synaptic weight is proportional to the negative gradient of the cost function with respect to that synaptic weight and is given as

$$\Delta W = -\eta \frac{\partial \xi}{\partial w}$$

where η is the learning rate parameter of the back-propagation algorithm.

Here we used MLP classifier and feature extractions as mel,mfcc,chroma for getting more accurate results.

SUPERVISED LEARNING :

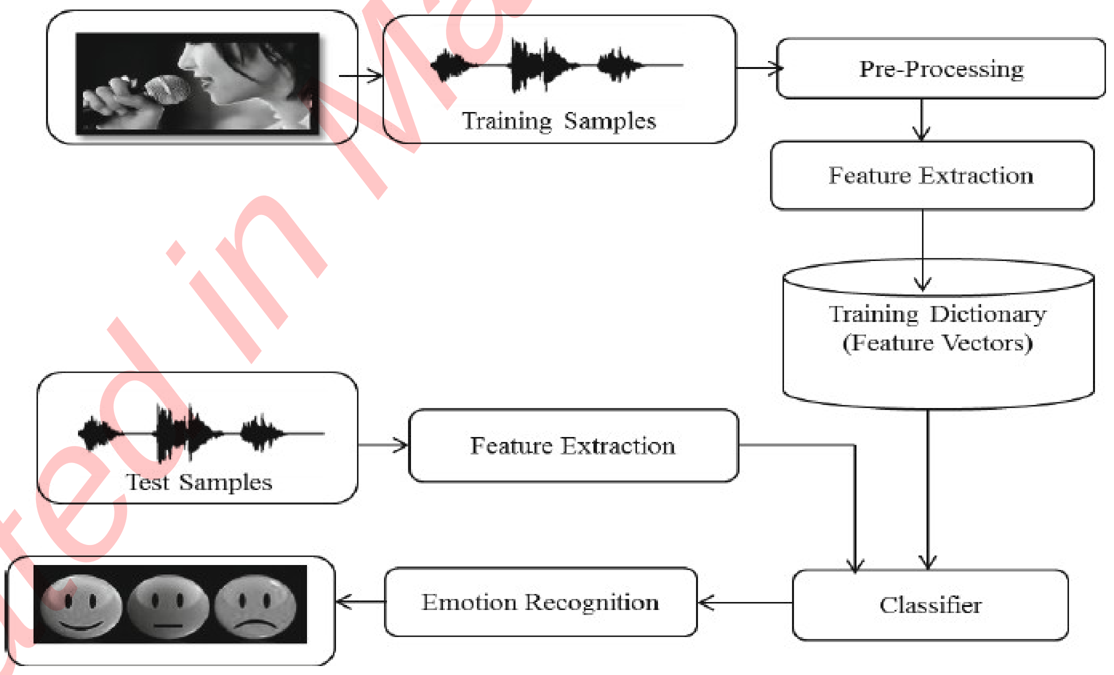
We are test the emotions using audio files. For this we used supervised learning.

In supervised learning a teacher or oracle is available which provides the desired action corresponding to a perception. A set of perception action pair provides what is called a training set.

Examples include an automated vehicle where a set of vision inputs and the corresponding steering actions are available to the learner.

DESIGN :

Usecase diagram



EXPLANATION AND TESTING:

To run code in the JupyterLab, you'll first need to run it with the command prompt:

```
C:\Users\Eswari>jupyter lab
```

This will open for you a new session in your browser. Create a new Console and start typing in your code. JupyterLab can execute multiple lines of code at once; pressing enter will not execute your code, you'll need to press Shift+Enter for the same.

The Dataset :

For this Python mini project, we'll use the RAVDESS dataset;
this is the Ryerson Audio-Visual Database of Emotional Speech and Song dataset, and is free to download. This dataset has 7356 files rated by 247 individuals 10 times on emotional validity, intensity, and genuineness. The entire dataset is 24.8GB from 24 actors, but we've lowered the sample rate on all the files, and you can

You'll need to install the following libraries with pip:

```
pip install librosa soundfile numpy sklearn pyaudio
```

1. Make the necessary imports:

```
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
```

2. Define a function extract_feature to extract the mfcc, chroma, and mel features from a sound file. This function takes 4 parameters- the file name and three Boolean parameters for the three features:

mfcc: Mel Frequency Cepstral Coefficient, represents the short-term power spectrum of a sound

chroma: Pertains to the 12 different pitch classes

mel: Mel Spectrogram Frequency

```
#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
```

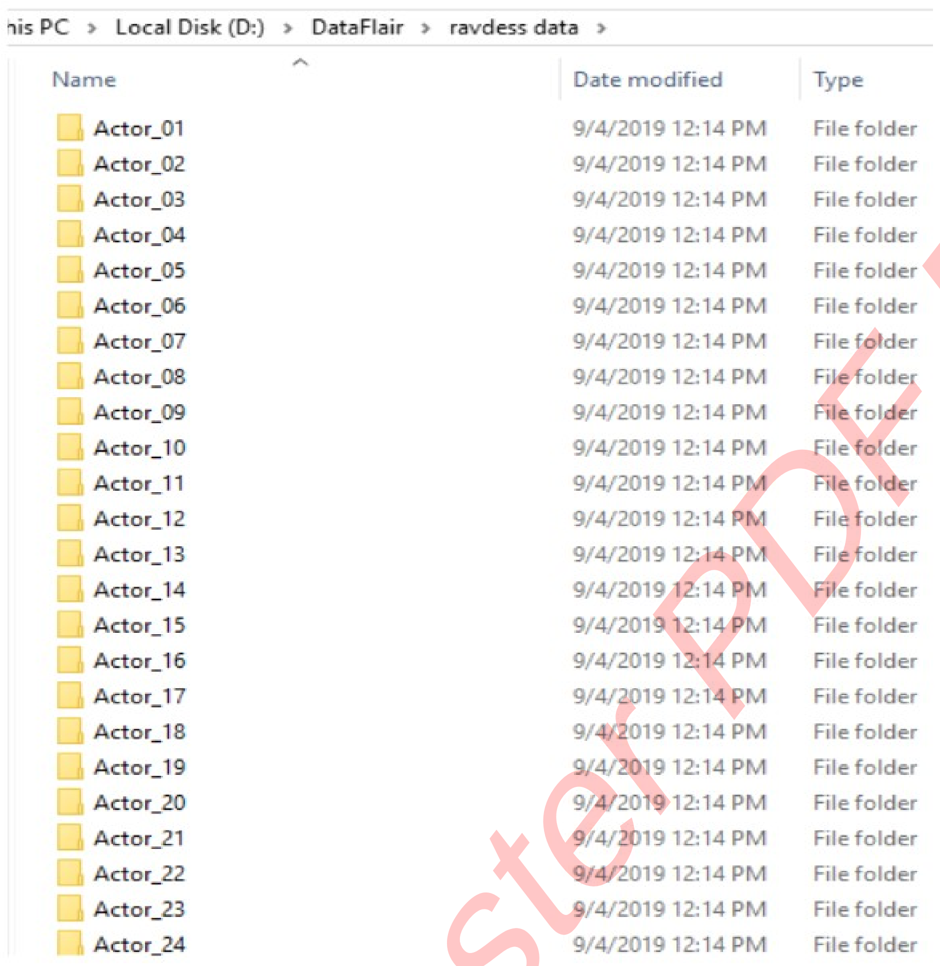
3. Now, let's define a dictionary to hold numbers and the emotions available in the RAVDESS dataset, and a list to hold those we want- calm, happy, fearful, disgust.

```
#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']
```

4. Now, let's load the data with a function load_data() – this takes in the relative size of the test set as parameter. x and y are empty lists; we'll use the glob() function from the glob module to get all the pathnames for the sound files in our dataset.

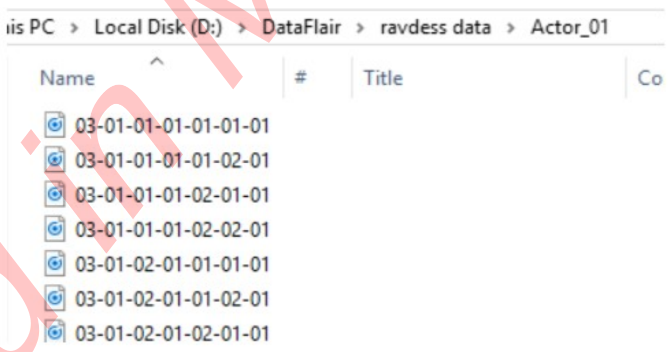
The pattern we use for this is: "D:\\DataFlair\\ravdess data\\Actor_*\\*.wav".
This is because our dataset looks like this:



his PC › Local Disk (D:) › DataFlair › ravdess data ›

Name	Date modified	Type
Actor_01	9/4/2019 12:14 PM	File folder
Actor_02	9/4/2019 12:14 PM	File folder
Actor_03	9/4/2019 12:14 PM	File folder
Actor_04	9/4/2019 12:14 PM	File folder
Actor_05	9/4/2019 12:14 PM	File folder
Actor_06	9/4/2019 12:14 PM	File folder
Actor_07	9/4/2019 12:14 PM	File folder
Actor_08	9/4/2019 12:14 PM	File folder
Actor_09	9/4/2019 12:14 PM	File folder
Actor_10	9/4/2019 12:14 PM	File folder
Actor_11	9/4/2019 12:14 PM	File folder
Actor_12	9/4/2019 12:14 PM	File folder
Actor_13	9/4/2019 12:14 PM	File folder
Actor_14	9/4/2019 12:14 PM	File folder
Actor_15	9/4/2019 12:14 PM	File folder
Actor_16	9/4/2019 12:14 PM	File folder
Actor_17	9/4/2019 12:14 PM	File folder
Actor_18	9/4/2019 12:14 PM	File folder
Actor_19	9/4/2019 12:14 PM	File folder
Actor_20	9/4/2019 12:14 PM	File folder
Actor_21	9/4/2019 12:14 PM	File folder
Actor_22	9/4/2019 12:14 PM	File folder
Actor_23	9/4/2019 12:14 PM	File folder
Actor_24	9/4/2019 12:14 PM	File folder

So, for each such path, get the basename of the file, the emotion by splitting the name around '-' and extracting the third value:



iis PC › Local Disk (D:) › DataFlair › ravdess data › Actor_01

Name	#	Title	Co
03-01-01-01-01-01-01			
03-01-01-01-01-02-01			
03-01-01-01-02-01-01			
03-01-01-01-02-02-01			
03-01-02-01-01-01-01			
03-01-02-01-01-02-01			
03-01-02-01-02-01-01			

Using our emotions dictionary, this number is turned into an emotion, and our function checks whether this emotion is in our list of observed_emotions; if not, it continues to the next file. It makes a call to extract_feature and stores what is returned in 'feature'. Then, it appends the feature to x and the emotion to y. So, the list x holds the features and y holds the emotions. We call the function train_test_split with these, the test size, and a random state value, and return that.

```
#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)
```

5. Time to split the dataset into training and testing sets! Let's keep the test set 25% of everything and use the load_data function for this.

```
#DataFlair - Split the dataset
x_train,x_test,y_train,y_test=load_data(test_size=0.25)
```

6. Observe the shape of the training and testing datasets:

```
#DataFlair - Get the shape of the training and testing datasets
print((x_train.shape[0], x_test.shape[0]))
```

7. And get the number of features extracted.

```
#DataFlair - Get the number of features extracted
print(f'Features extracted: {x_train.shape[1]}')
```

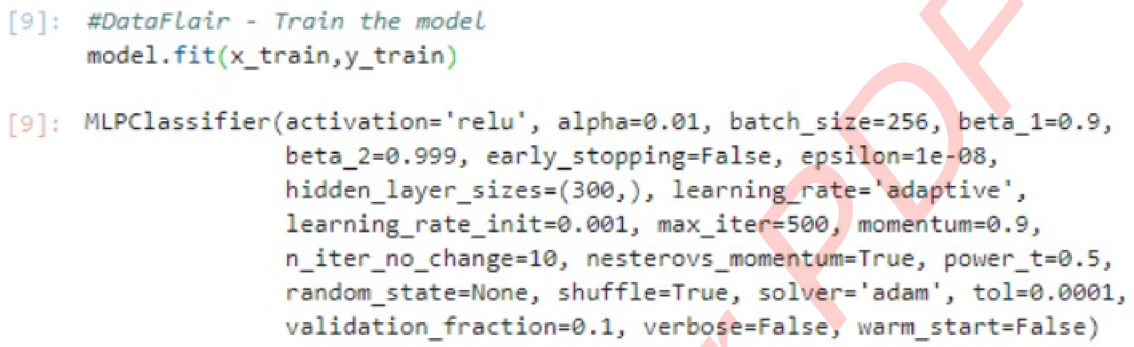
8. Now, let's initialize an MLPClassifier. This is a Multi-layer Perceptron Classifier; it optimizes the log-loss function using LBFGS or stochastic gradient descent. Unlike SVM or Naive Bayes, the MLPClassifier has an internal neural network for the purpose of classification. This is a feedforward ANN model.

```
#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)
```

9. Fit/train the model.

```
#DataFlair - Train the model
model.fit(x_train,y_train)
```

Output Screenshot :



```
[9]: #DataFlair - Train the model
     model.fit(x_train,y_train)

[9]: MLPClassifier(activation='relu', alpha=0.01, batch_size=256, beta_1=0.9,
                   beta_2=0.999, early_stopping=False, epsilon=1e-08,
                   hidden_layer_sizes=(300,), learning_rate='adaptive',
                   learning_rate_init=0.001, max_iter=500, momentum=0.9,
                   n_iter_no_change=10, nesterovs_momentum=True, power_t=0.5,
                   random_state=None, shuffle=True, solver='adam', tol=0.0001,
                   validation_fraction=0.1, verbose=False, warm_start=False)
```

10. Let's predict the values for the test set. This gives us y_pred (the predicted emotions for the features in the test set).

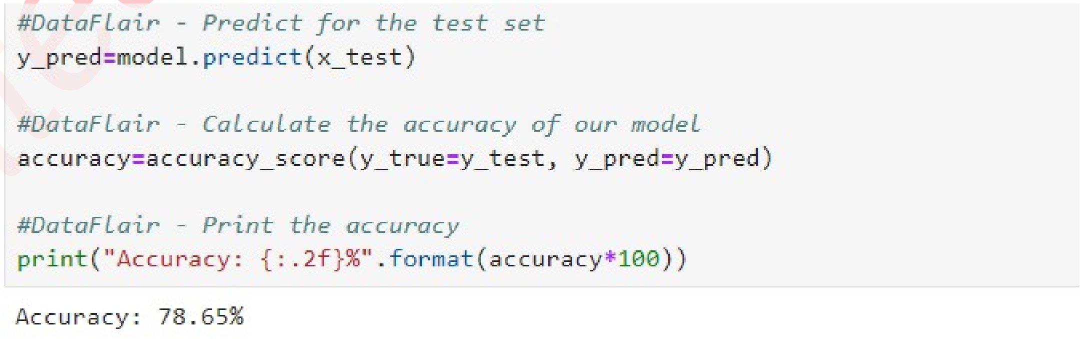
```
#DataFlair - Predict for the test set
y_pred=model.predict(x_test)
```

11. To calculate the accuracy of our model, we'll call up the accuracy_score() function we imported f sklearn. Finally, we'll round the accuracy to 2 decimal places and print it out.

```
#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))
```

Output Screenshot :



```
#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))

Accuracy: 78.65%
```

CONCLUSION:

The paper explored the idea of detecting the emotional state of a person by speech processing techniques. The study on words and letters under different emotional situations proved that the emotional state can alter the speech. It was observed that there are distinguishable features in a speech segment that characterizes each emotion state. After performing the classification tests on a dataset , it was observed that it is better considering data from an individual subject rather than a group of people. The development of a software based agent for emotion detection and heart rate analysis can greatly improve telemedicine based systems.

REFERENCES:

[1] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2002S28

[2] R.Banse, K.R.Scherer, “Acoustic profiles in vocal emotion expression”, Journal of Personality and Social Psychology, Vol.70, 614-636, 1996

[3] T.Bänziger, K.R.Scherer, “The role of intonation in emotional expression”, Speech Communication, Vol.46, 252-267, 2005

[4] F.Yu, E.Chang, Y.Xu, H.Shum, “Emotion detection from speech to enrich multimedia content”, Lecture Notes In Computer Science,Vol.2195, 550-557, 2001

[5] D.Talkin, “A Robust Algorithm for Pitch Tracking (RAPT)”, Speech Coding & Synthesis, 1995

[6] https://data-flair.training/blogs/python-mini-project-speech-emotion-recognition.