Voice emotion detection model. Ft. SIH

In [1]:
```
import librosa
import soundfile
import pickle
import os, sys, 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 warnings
if not sys.warnoptions:
    warnings.simplefilter("ignore")
warnings.filterwarnings("ignore", category=DeprecationWarning)
```

In [2]:
```
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
```

In [3]:
```
emotions={
  '01':'neutral',
  '02':'calm',
  '03':'happy',
  '04':'sad',
  '05':'angry',
  '06':'fearful',
  '07':'disgust',
  '08':'surprised'
}
observed_emotions=['calm', 'happy', 'fearful', 'disgust']
```

In [4]:
```
def load_data(test_size=0.2):
    x,y=[],[]
    for file in glob.glob("DataSets2\*.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)
```

In [5]:
```
x_train,x_test,y_train,y_test=load_data(test_size=0.25)
```

In [6]:
```
print((x_train.shape[0], x_test.shape[0]))
```

```
(9121, 3041)
```

In [7]:
```
print(f'Features extracted: {x_train.shape[1]}')
```

```
Features extracted: 180
```

In [8]:
```
model=MLPClassifier(alpha=0.01, batch_size=256, epsilon=1e-08, hidden_layer_sizes=(300,), learning_rate='adaptive', max_iter=500)
```

In [9]:
```
model.fit(x_train,y_train)
```

Out[9]:
```
MLPClassifier(alpha=0.01, batch_size=256, hidden_layer_sizes=(300,),
              learning_rate='adaptive', max_iter=500)
```

In [10]:
```
y_pred=model.predict(x_test)
```

In [11]:
```
accuracy=accuracy_score(y_true=y_test, y_pred=y_pred)
print("Accuracy: {:.2f}%".format(accuracy*100))
```

```
Accuracy: 100.00%
```

In [12]:
```
pickle.dump(model,open('model_v3.pkl','wb'))
model=pickle.load(open('model_v3.pkl', 'rb'))
```

Output Saved to model_v3.pkl