Confidence-Level Module

In this module, work has to be carried out with the **audio data.**

Initially, we had gone through various datasets and articles which included Emotion Detection from audio.

Problem:

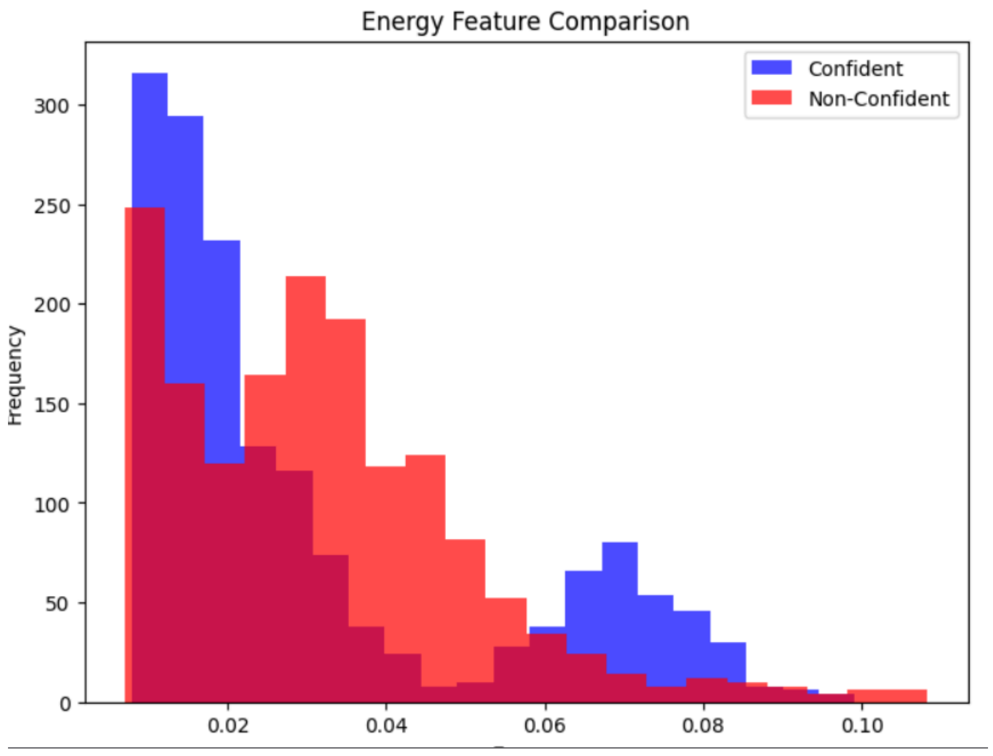
There is no dataset available which identifies the confidence level of a person.

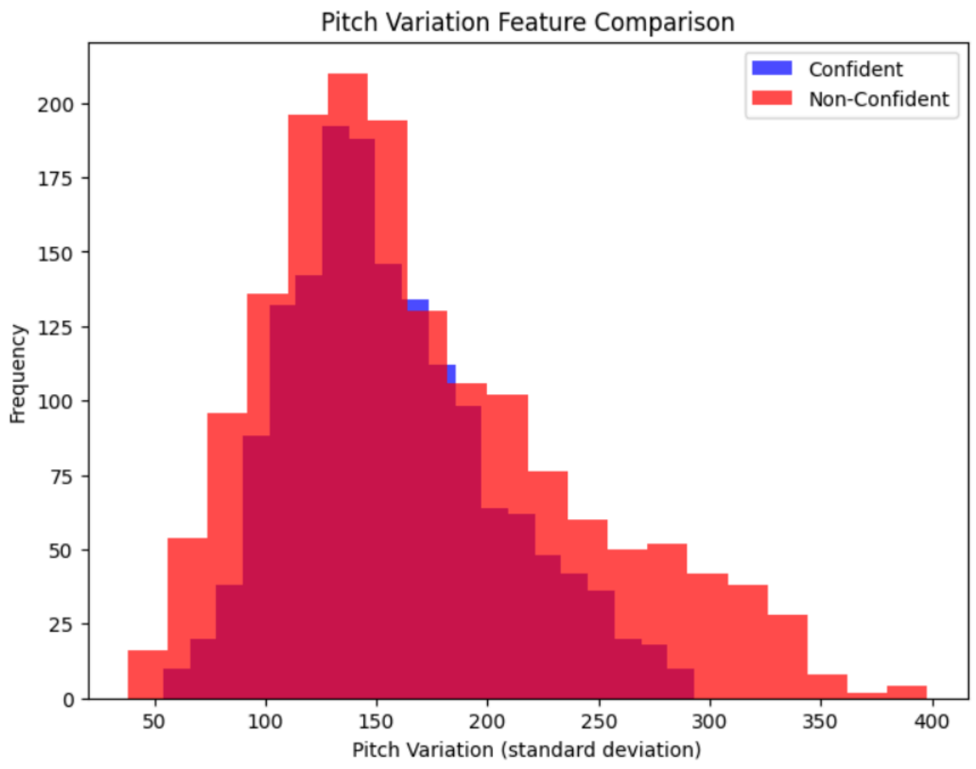
Approach:

Carried out feature extraction from the TESS audio dataset which identify emotions like happy, sad, fear etc.

- ➔ **Happy, neutral** labels are mapped in confident label
- ➔ **sad , fear** labels are mapped in non-confident label

Extraction of features like speech-rate, pitch, energy, mfcc features etc. Some of them shown below:





There's more pitch variation in non-confident(fear,sad labels) as compared to confident(happy, neutral labels)

From different graphs, relevant features which can show difference are pitch variation, energy, speech rate, mfcc feature, zcr, spectral flux.

- **Pitch Variation:** Variations in pitch between confident and non-confident speech.
- **Energy:** Differences in energy levels between confident and non-confident speech.
- **Speech Rate:** Variation in speech rate between confident and non-confident speech.
- **MFCC Features**: Mel-Frequency Cepstral Coefficients capturing spectral characteristics of speech.

- **Zero Crossing Rate (ZCR):** Rate at which the audio signal changes its sign (positive to negative or vice versa), indicating speech dynamics.
- **Spectral Flux:** Variation in spectral flux, which measures the change in spectral content over time.

I was asked to add new features (timbre features) so I added spectral contrast, HNR and spectral_rolloff_mean

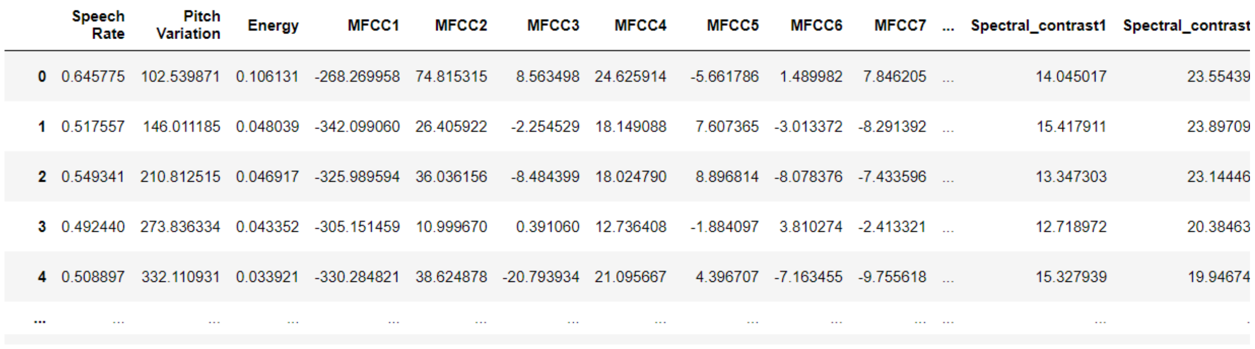
- **Spectral Contrast**: Spectral contrast measures the difference in amplitude between peaks and valleys in a spectrum.
- **Harmonic-to-Noise Ratio (HNR)**: HNR quantifies the ratio of harmonic (tonal) components to noise (non-tonal) components in the audio signal.
- **Spectral_rolloff_mean** : frequency below which a certain percentage of the total spectral energy lies.

Dataset

Audio Dataset prepared after feature extraction contains 26 input features features and 1 output label which determines 'Confident' or 'Not-Confident'

It contains a total of **3200** records :

➔ 1600 for 'Confident' and 1600 for 'Non-Confident'.



	Speech Rate	Pitch Variation	Energy	MFCC1	MFCC2	MFCC3	MFCC4	MFCC5	MFCC6	MFCC7	...	Spectral_contrast1	Spectral_contrast
0	0.645775	102.539871	0.106131	-268.269958	74.815315	8.563498	24.625914	-5.661786	1.489982	7.846205	...	14.045017	23.55439
1	0.517557	146.011185	0.048039	-342.099060	26.405922	-2.254529	18.149088	7.607365	-3.013372	-8.291392	...	15.417911	23.89709
2	0.549341	210.812515	0.046917	-325.989594	36.036156	-8.484399	18.024790	8.896814	-8.078376	-7.433596	...	13.347303	23.14446
3	0.492440	273.836334	0.043352	-305.151459	10.999670	0.391060	12.736408	-1.884097	3.810274	-2.413321	...	12.718972	20.38463
4	0.508897	332.110931	0.033921	-330.284821	38.624878	-20.793934	21.095667	4.396707	-7.163455	-9.755618	...	15.327939	19.94674
...	...	...	...	...	...	...	...	...	...	...	...	...	.

Model Training:

- I tried several classification algorithms from which Random forest found to be most promising with **96-97%** accuracy. With **n_estimators** set to ‘10’ and **max_depth** set to ‘5’

- Neural network model gave the similar accuracy but it was inconsistent when parameters were changed or feature selection was changing.

- So, finally I decided to go with Random Forest Classifier and prepared a joblib file for the same.

Problems:

Checking with Real Data :

I collected some audio data using youtube mock interviews videos and checked how well it is performing.

Results:

The model cannot properly identify for **confident** audios.

For audios which I found out to be **non-confident** audios, it is performing right when only when there are significant signs of being a non-confident audio.

Possible approaches:

- To prepare a new dataset by collecting files over the internet (can take a long time) and then use Deep Learning Models .

- To train neural-network model on the already available dataset I prepared using feature extraction.

- Perform statistical analysis on the available dataset for each and every feature and correlate with confidence (Initially it was done already, but need to identify errors there).

New Phase

As decided, I along with my teammate Achal have been working on collecting data from videos from online mock interviews and then train the models.

Data Collected was in (.wav) format which contained audio files in time ranginging from **5-30 seconds ,** then feature extraction was performed to prepare dataset for model training.

Initially I collected **100** records (50 for each label), And the accuracy for that using Random Forest, SVC, KNN was coming around **65**% which was not a good accuracy.

- ➔ Model was unable to identify Non-confident data points properly.
- ➔ Confident label was performing fine.

Improvement phase:

Now, the target was the collect more data so that model training can be improved. We have collected **250** records (**139** for confident and **111** for non-confident)

The dataset contains:

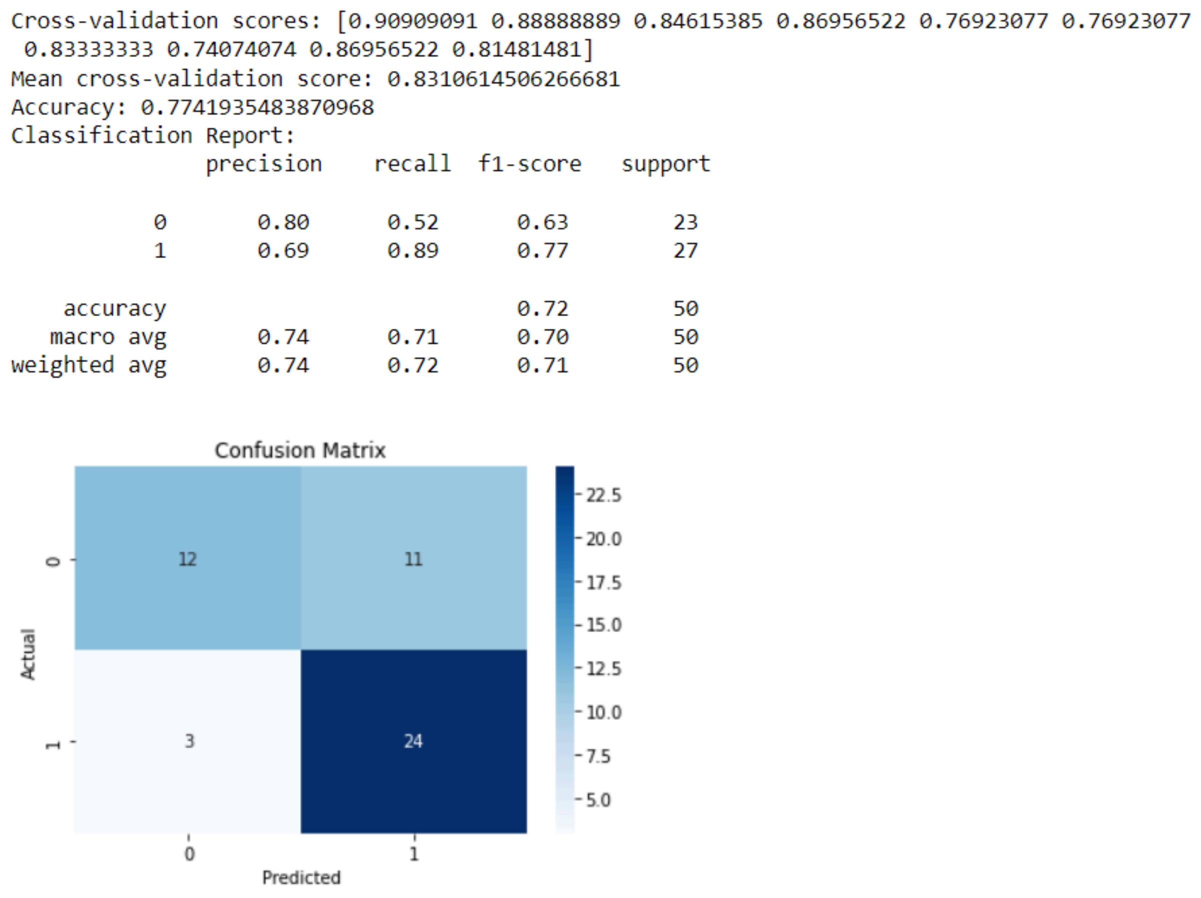
- ➔ 250 records (139 + 111)
- ➔ 26 input features
- ➔ 1 output label (High confidence , Low Confidence)

After checking each data points manually and doing some changes like removing noisy audios, putting new data points which were collected etc.

The Best performing models were:

- ➔ KNN(n_neighbours = 3) (**75%** f1_score)
- ➔ SVC(kernel='rbf', c=1.0) (**77%** f1_score) (cv_score = 83 for 10 folds)

The figure for Confusion Matrix for SVC is below:



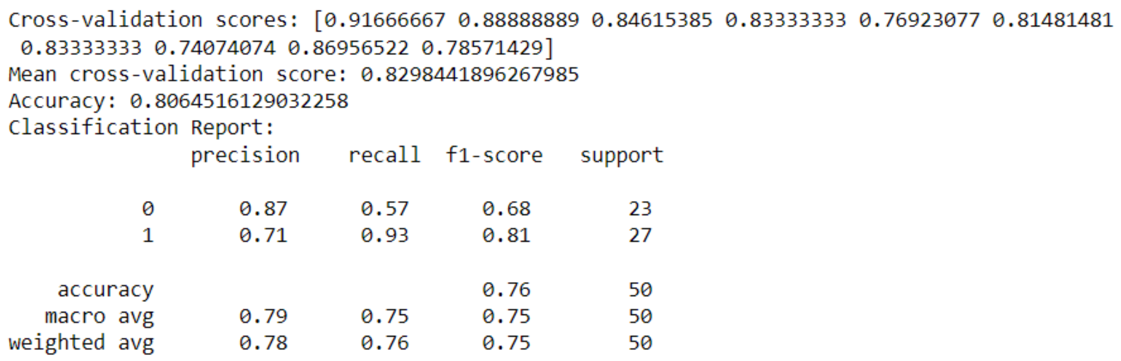
```
Cross-validation scores: [0.90909091 0.88888889 0.84615385 0.86956522 0.76923077 0.76923077
 0.83333333 0.74074074 0.86956522 0.81481481]
Mean cross-validation score: 0.8310614506266681
Accuracy: 0.7741935483870968
Classification Report:
              precision    recall  f1-score   support

           0       0.80      0.52      0.63        23
           1       0.69      0.89      0.77        27

    accuracy                           0.72        50
   macro avg       0.74      0.71      0.70        50
weighted avg       0.74      0.72      0.71        50
```

f1_score is used for evaluation here as it is a classification problem.

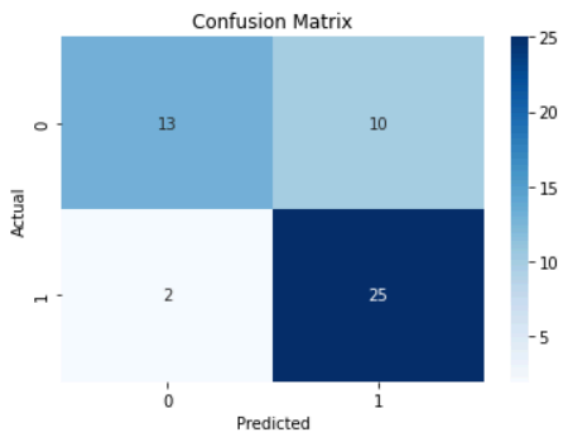
Model	**f1_score(approx)**
SVC	77
RFC	72
KNN	74

Performing Feature Importance:
For the best performing model i.e. SVC, performed feature importance using permutation technique where I removed 2 unimportant features.

After that, SVC is giving close to **80%** f1_score as shown in the figure below :



```
Cross-validation scores: [0.91666667 0.88888889 0.84615385 0.83333333 0.76923077 0.81481481
 0.83333333 0.74074074 0.86956522 0.78571429]
Mean cross-validation score: 0.8298441896267985
Accuracy: 0.8064516129032258
Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.57      0.68        23
           1       0.71      0.93      0.81        27

    accuracy                           0.76        50
   macro avg       0.79      0.75      0.75        50
weighted avg       0.78      0.76      0.75        50
```



Conclusion:
- ➔ Its performing well for Confident label,
- ➔ And for Non-confident label, the model is better than before but still there's a scope of improvement.

Next possible thing to be tried:

Try to perform more feature importance and remove those features which are stopping the model to improve.