Pattern Recognition and Machine Learning Project Report
Speech Emotion Recognition

Siddharth Singh B20EE067, Shreya Sachan B20EE065, Aditi Tiwari B20EE005

Abstract

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. In this project We particularly focused on feature engineering techniques for audio data and provide an in-depth look at the logic, concepts, and properties of the Multilayer Perceptron (MLP) model, an ancestor and the origin of deep neural networks (DNNs) today. We also provide an introduction to a few basic machine learning models.

1 Introduction

In this project we are going to recognize the emotion from speech tone(of audio file) in sentiment category(calm, happy, sad, angry, fearful, neutral, surprised, and disgust).An audio file can be represented as a time series with the dependent axis being the amplitude of the audio waveform. The waveform of the sound file is all the information we have with to create features to train our model. For the project we had used dataset provided in link Link for dataset.

2 Libraries for the Model

Here we installed some dependencies to work with audio file such as :

- Librosa: Python package for music and audio analysis.
- Matplotlib: This allows us to plot spectrograms
- Scipy.io.wavfile: Return the sample rate (in samples/sec) and data from a WAV file.
- Glob: Used to return all file paths that match a specific pattern.
- Fastai: Fastai is a deep learning library that provides practitioners with high-level components that can quickly and easily provide state-of-the-art results in standard deep learning domains. We will be using fastai.vision.
- Numpy: Used for working with arrays.
- Pandas: Used to make DataFrame.
- Matplotlib: This allows us to plot spectrograms

Libraries For Making It Work Live:

- Struct: Used to unpack audio data into integers.
- Tkinter: Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications.
- Sounddevice: This Python module provides bindings for the PortAudio library and a few convenient functions to play and record NumPy arrays containing audio signals.

3 Data Description, Visualization and Preprocessing

3.1 Data Description

We're going to use the audio-only speech portion of the RAVDESS dataset, 200MB. Audio is sourced from 24 actors (12 male, 12 female) repeating two sentences with a variety of emotions and intensity. We get 1440 speech files (24 actors * 60 recordings per actor). Each audio sample has been rated by a human 10 times for emotional quality.

3.2 Data Preprocessing

The first thing we will do is to convert the sound clips into a graphical spectrogram using the librosa library. That way they can be used to train the resnet model(Live Speak emotion recognition).
We are converting the sound data into a spectrogram with the MEL scale as this scale. This scale is designed to make the image representations of the audio more interpretable inorder to allow the CNN to find patterns in the data.
Secondly we converted the spectrograms to csv data file by extracting the features(matrix representation) and further separated data in to training and testing set for better visualization of accuracy.

3.3 Data Visualization

Here is visualization for dataset, the classes appears to be balanced. That makes the task easier. All emotions except the **neutral** class have a ”strong” intensity so there are half as many neutral samples.
for our Dataset we had considered ’01’:neutral, ’02’:calm, ’03’:happy, ’04’:sad, ’05’:angry, ’06’:fearful, ’07’:disgust and ’08’:surprised.

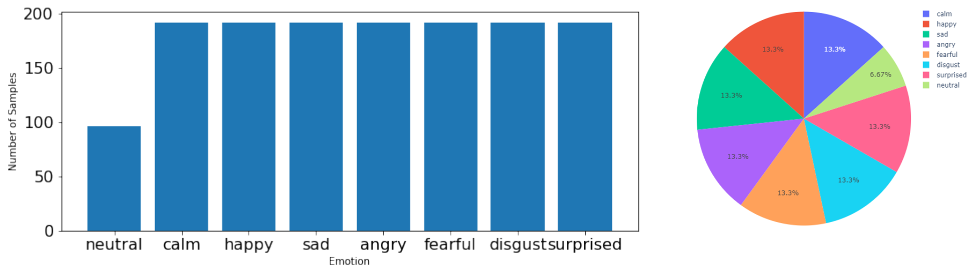


Figure 1: Dataset distribution for different classes available

Different type of Chromagram, Spectrogram -
We build our feature extraction functions to get a chromagram, a mel spectorgram, and MFC coefficients for each of our audio files. Because the chromagram, mel spectrogram and MFCCs are calculated on audio frames produced by STFT, we’re going to get a matrix back from each function, so we’ll take the mean of those matrices to produce a single feature array for each feature and each audio sample.

- **Chromagram**: Will produce 12 features; One for each of 12 pitch classes.
- **Mel Spectrogram**: Will produce 128 features; and the number of mel frequency bands at num mels=128.
- **MFCC**: Will produce 40 MFCCs; I’ve set the number of coefficients to return at num mfcc=40.

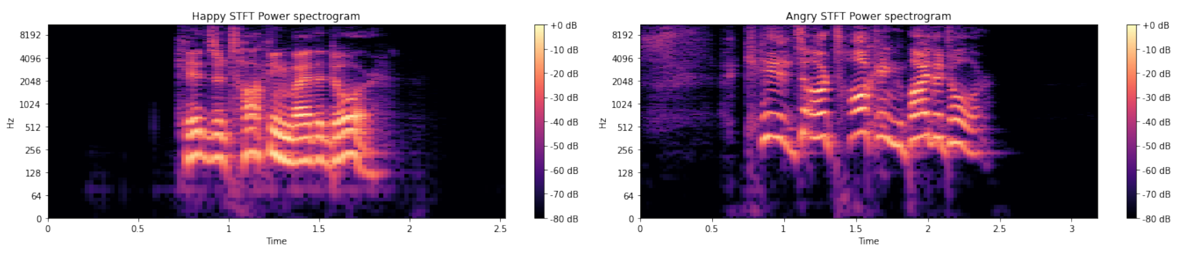


Figure 2: *STFT Power Spectrogram for Happy and Angry Emotions*

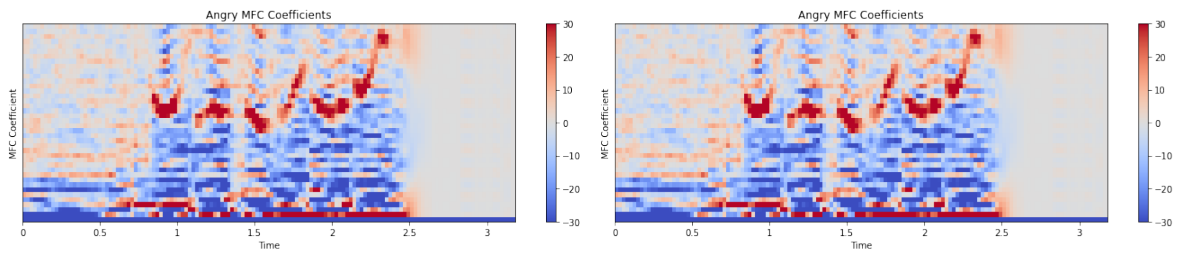


Figure 3: *MFC Coefficients for Happy and Angry Emotions*

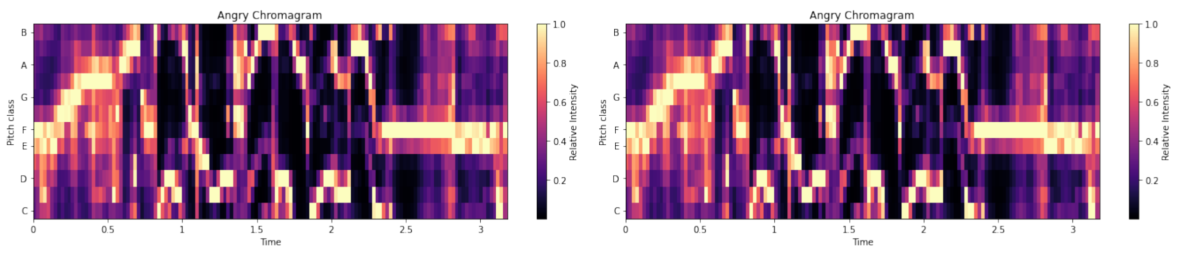


Figure 4: *Chromagram for Happy and Angry Emotions*

4 Training The Model

4.1 Machine Learning Models

4.1.1 K Nearest Neighbor

KNN is one of the simplest forms of machine learning algorithms mostly used for classification. KNN classifies the new data points based on the similarity measure of the earlier stored data points. We performed hyper-parameter tuning using GridSearchCv and have taken best parameter, leaf_size=1, p=1 and n_neighbors=1. Achieved accuracy of 61.3% over testing.

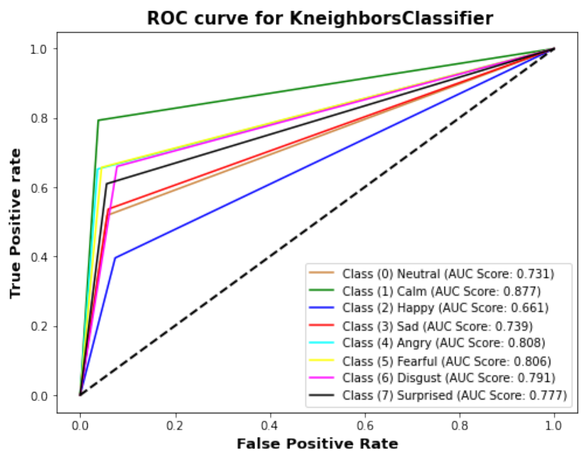


Figure 5: *ROC_AUC Curve for KNN*

4.1.2 Random Forest Classifier

Random forests algorithm is general technique of bootstrap aggregating to tree learners. Given a training set X with responses Y, bagging repeatedly selects a random sample with replacement of the training set and fits trees to these samples.Random forest was fitted with different parameters. We find the best accuracy of 54.0% by taking parameter, max_depth=110, min_samples_leaf=1, min_samples_split=8, n_estimators=300.

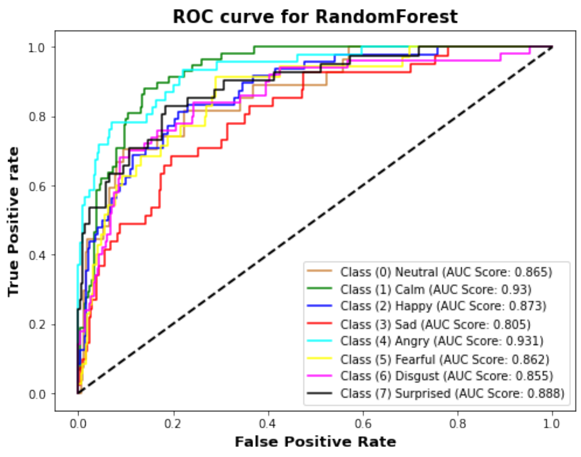


Figure 6: *ROC_AUC Curve for Random Forest*

4.1.3 XGB Classifier

XGBoost, which stands for Extreme Gradient Boosting is an implementation of Gradient Boosted decision trees. In it decision trees are created in sequential form. Weights are assigned to all the independent variables which are then fed into the decision tree which predicts results. These individual classifiers then ensemble to give a more precise model. We achieved an accuracy of 53.5% over testing.

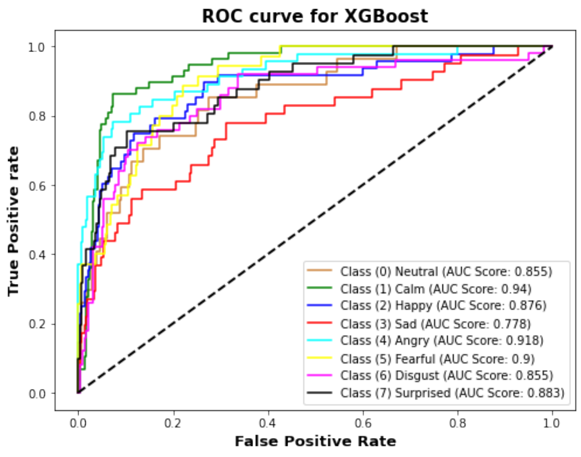


Figure 7: *ROC_AUC Curve for XGBoost*

Accuracy for different models like KNN, RFC, XGB

Classifier	**Accuracy**
KNeighboursClassifier	61.3
RandomForestClassifier	54.6
XGBoost Classifier	53.5

4.2 Deep Learning Models

4.2.1 Neural Network

A three dense layers Neural Network was trained. The activation function for the first layer was Relu (Rectified Linear Unit), for second it was sigmoid and the last hidden layer utilized Softmax activation function. A Dropout layer with dropout rate of 0.07 was added. The Dropout layer randomly sets input units to 0 at each step during training time, which helps prevent overfitting.The model is then compiled with loss='categorical_crossentropy' and optimizer='RMSProp' and runned for 100 epochs.
It achieved a training accuracy of 99.38% and validation accuracy of 55.95%, which indicates that the model was not able to generalise well.

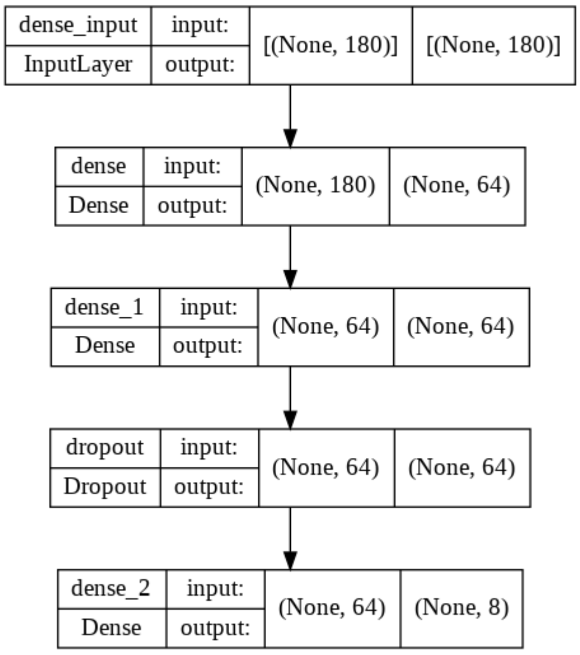

dense_input | InputLayer | input: [(None, 180)] | output: [(None, 180)]
dense | Dense | input: (None, 180) | output: (None, 64)
dense_1 | Dense | input: (None, 64) | output: (None, 64)
dropout | Dropout | input: (None, 64) | output: (None, 64)
dense_2 | Dense | input: (None, 64) | output: (None, 8)

Figure 8: *Neural Network Architecture*

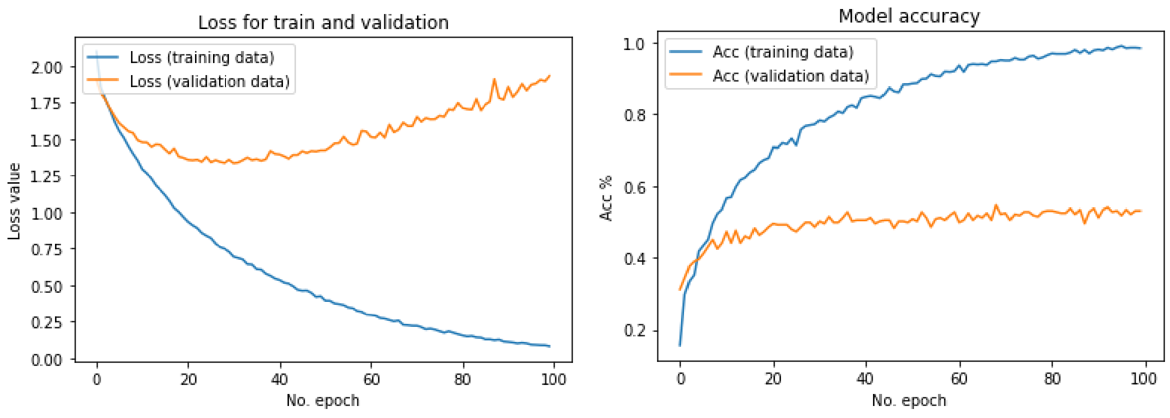


Figure 9: *Loss and Accuracy Plots for train and Validation Data*

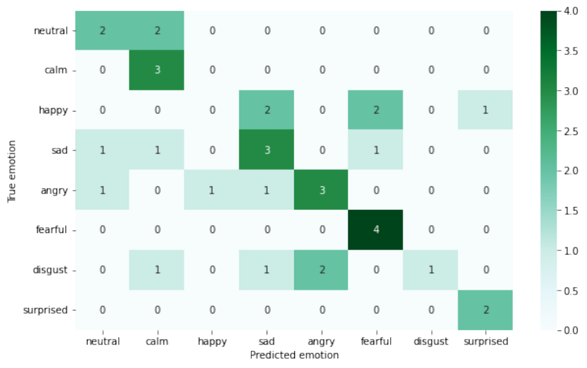


Figure 10: *Confusion Matrix for Testing Data*

4.3 Training the model for live speech emotion Recognition

We trained, pre-trained CNN (resnet34) model on the spectrogram images earlier generated and labeled.Then we saved updated weights to pickle file.

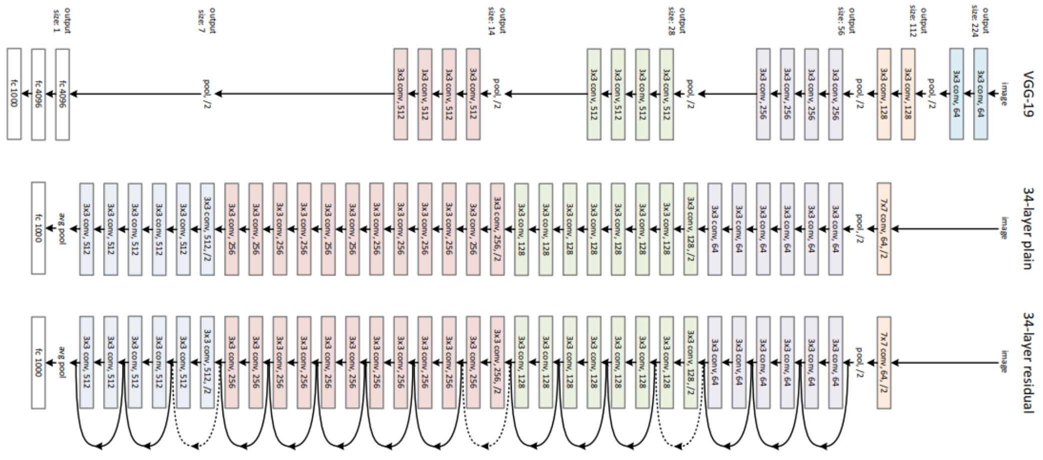


Figure 11: *ResNet-34 architecture fig.*

Now, for live speech emotion Recognition, First load in the model which is in the form of a pickle file. Then we record audio for the time range of the variable second which in our case is set to 3 seconds. Then the audio is exported as a wav file. After it we are going to use that wav file and convert it into a spectrogram image. Finally, use model.predict(img) to predict the sentiment of the image.

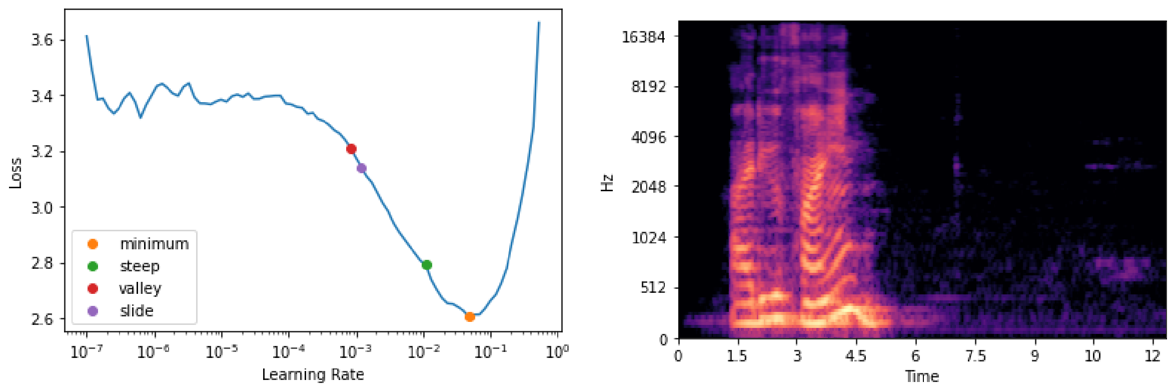
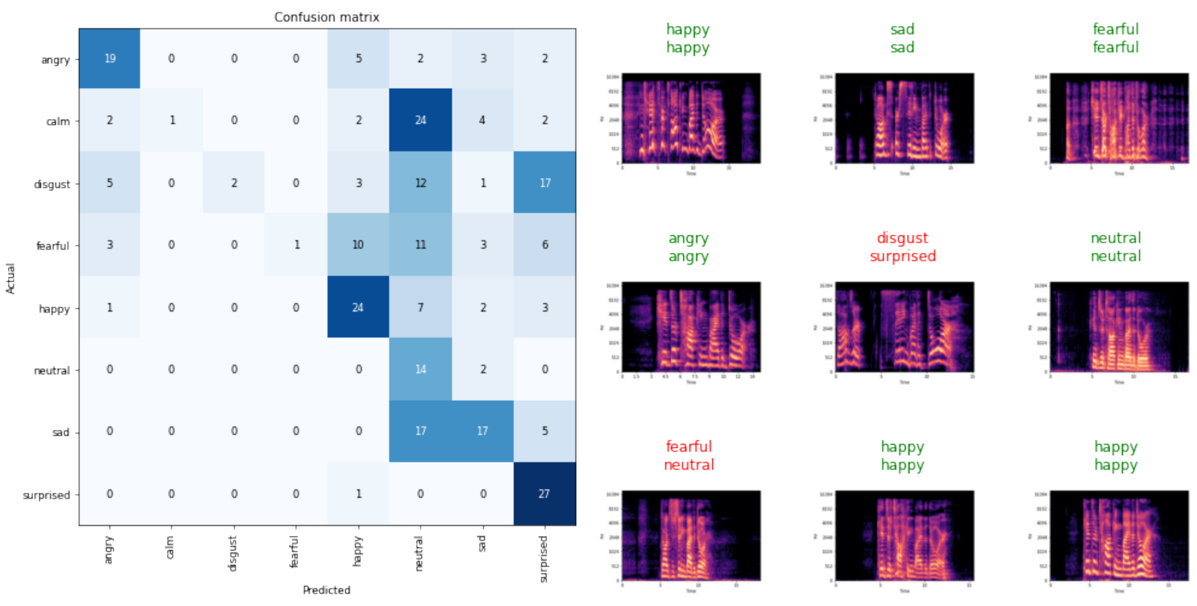


Figure 12: *Loss vs Learning Rate and Spectogram for Resnet Model*



	angry	calm	disgust	fearful	happy	neutral	sad	surprised
angry	19	0	0	0	5	2	3	2
calm	2	1	0	0	2	24	4	2
disgust	5	0	2	0	3	12	1	17
fearful	3	0	0	1	10	11	3	6
happy	1	0	0	0	24	7	2	3
neutral	0	0	0	0	0	14	2	0
sad	0	0	0	0	0	17	17	5
surprised	0	0	0	0	1	0	0	27

Figure 13: *Confusion Matrix and Prediction Result of Resnet Model*

5 Conclusion

Classical machine learning models such as KNeighborsClassrifier (KNN), XGBClassifier (XGB), and Random Forests have distinct advantages to deep neural networks in many tasks but do not match the performance. KNeighbours Classifier performs the best in case of traditional Machine learning Algoritm.
Deep learning models mainly Neural Network architecture always overperforms traditional machine learning algorithms.We got the best accuracy for Resnet34 model.The use of three features (MFCC's, MSF's and chroma STFT) gave impressive accuracy in both simple and deep learning models, reiterating the importance of feature selection and understanding the data in order to select the proper preprocessing methods.For future improvements, some possible features to explore concerning speech would be MFCC Filterbanks or features extracted using the perceptual linear predictive (PLP) technique. These features could affect the performance of models in the emotion classification task.

Type	**Classifier**	**Accuracy**
Machine Learning	KNeighbours	61.3
Machine Learning	RandomForest	54.0
Machine Learning	XGBClassifier	53.5
Deep Learning	Neural Network	55.95
Deep Learning	Resnet34 Model	61.55

References

[1] Understanding and visualizing ResNets— by Pablo Ruiz —Towards DataScience

[2] Simple understanding and implementation of KNN algorithm!— Sai Patwardhan — Analytics Vidya

[3] Preprocess Audio Data with the Signal Envelope — by Jeremy Pagirsky — Towards DataScience

[4] Learning from Audio: The Mel Scale, Mel Spectrograms, and Mel Frequency Cepstral Coefficients — by mlearnere — Towards Data Science Wesley, Massachusetts, 2nd ed.

[5] K-Nearest Neighbors in Python + Hyperparameters Tuning — by Adipta Martulandi — Towards DataScience

Contribution

- **Aditi Tiwari(B20EE005)-** Worked on iterating different preprocessing aspects and best in-depth exploratory data analysis. She has written the code to convert audio files in spectograms. She has also written code to visulize (Chromagram,Mel-Frequency Cepstral Coefficients, Mel Spectrogram), also worked on emotion analysis also.
- **Shreya Sachan(B20EE065)-** Worked on exploratory data analysis. She has worked extensively on configuring KNeighborsClassifier, RandomForestClassifier, XGBClassifier model and making Classification Report. She has also did feature extraction from Audio files and worked on the emotion analysis code.
- **Siddharth Singh(B20EE067)-** Worked on implementing the 2 elaborate Deep Learning models of Neural Network, resnet34 model. He is also responsible for Live speech emotion recognition. He has also analysed the performence of models. He has written the code get emotion to predict emotion of audio file. He has also plotted ROC_AUC curve for traditional Machine Learning Algorithm.