**Speech Emotion Recognition**

Final Project Documentation

Suraj Pandey

Shanmukha Sarat Ponugupati

Mentor: Paras Ahuja

**Problem Introduction and its Usability:**

Detecting emotions is one of the most important marketing strategy in today’s world. You could personalize different things for an individual specifically to suit their interest. For this reason, we decided to do a project where we could detect a person’s emotions just by their voice which will let us manage many AI related applications. Some examples could be including call centers to play music when one is angry on the call. Another could be a smart car slowing down when one is angry or fearful. As a result this type of application has much potential in the world that would benefit companies and also even safety to consumers.

**Data Used:** We got audio datasets with around 2000 audio files which were in the wav format from the following website: <http://neuron.arts.ryerson.ca/ravdess/?f=3>.

The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) contains 7356 files (total size: 24.8 GB). The database contains 24 professional actors (12 female, 12 male), vocalizing two lexically-matched statements in a neutral North American accent. Speech includes calm, happy, sad, angry, fearful, surprise, and disgust expressions, and song contains calm, happy, sad, angry, and fearful emotions. Each expression is produced at two levels of emotional intensity (normal, strong), with an additional neutral expression. All conditions are available in three modality formats: Audio-only (16bit, 48kHz .wav), Audio-Video (720p H.264, AAC 48kHz, .mp4), and Video-only (no sound).  Note, there are no song files for Actor\_18.

The first website contains speech data which is available in three different format.

1. Audio Visual – Video with speech
2. Speech – Audio only
3. Visual – Video only

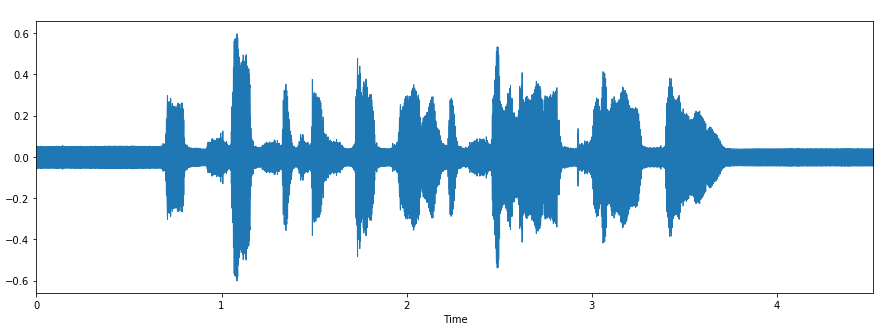
We went with the Audio only zip file because we are dealing with finding emotions from speech. The zip file consisted of around 1500 audio files which were in wav format.

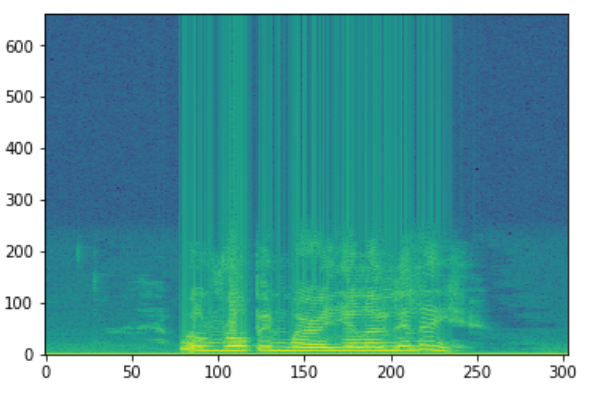
Each of the 7356 RAVDESS files has a unique filename. The filename consists of a 7-part numerical identifier (e.g., 02-01-06-01-02-01-12.mp4). These identifiers define the stimulus characteristics:   
  
**Filename identifiers**

* 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).

For the purpose of this project, we have used only four emotions (Calm, Happy, Sad, Angry) of all actors. Therefore, we landed with a dataset of 768 Audio files.

We tested out one of the audio file to know its features by plotting its waveform and spectrogram.





The next step involves organizing the audio files. Each audio file has a unique identifier at the 6th position of the file name which can be used to determine the emotion the audio file consists. We have 4 different emotions in our dataset.

1. Calm
2. Happy
3. Sad
4. Angry

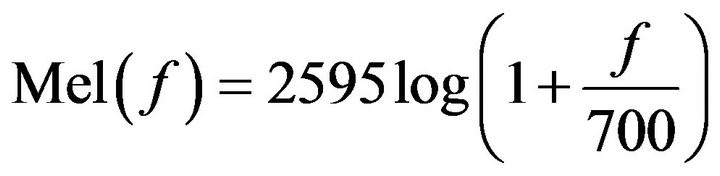
**FEATURE EXTRACTION**

We used Librosa library in Python to process and extract features from the audio files. Librosa is a python package for music and audio analysis. It provides the building blocks necessary to create music information retrieval systems. Using the librosa library we were able to extract important audio features such as

1. Mel Scale
2. Mel spectrogram
3. MFCC(Mel Frequency Cepstral Coefficient)

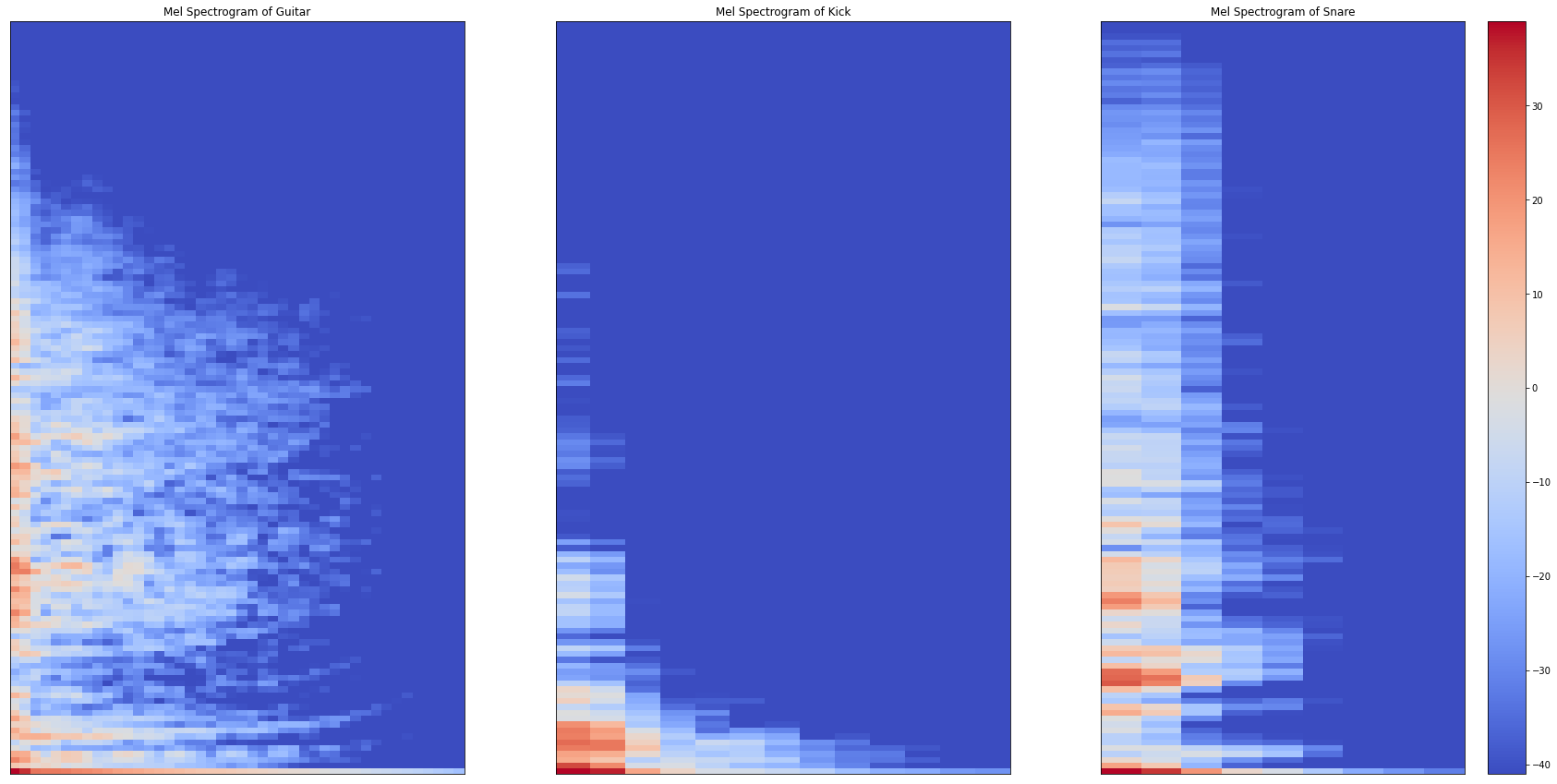
**MEL-SCALE:**

Pitch is one of the characteristics of a speech signal and is measured as the frequency of the signal. Mel scale is a scale that relates the perceived frequency of a tone to the actual measured frequency. It scales the frequency in order to match more closely what the human ear can hear (humans are better at identifying small changes in speech at lower frequencies). This scale has been derived from sets of experiments on human subjects. Let me give you an intuitive explanation of what the mel scale captures.



Mel Spectrograms:

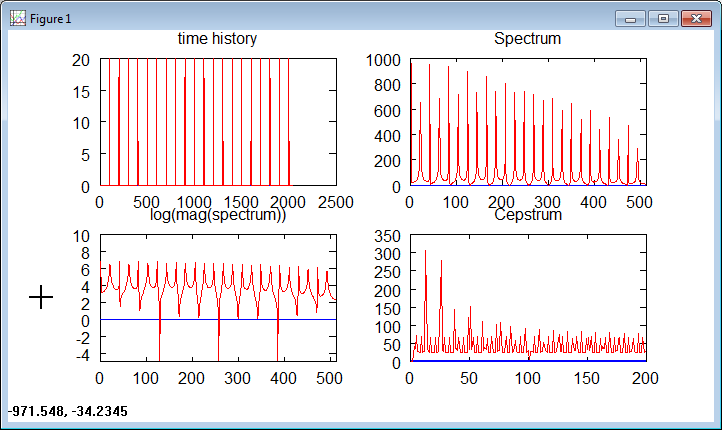
Mel Spectrograms are spectrograms that visualize sounds on the Mel scale as opposed [to the frequency domain](https://towardsdatascience.com/learning-from-audio-spectrograms-37df29dba98c).

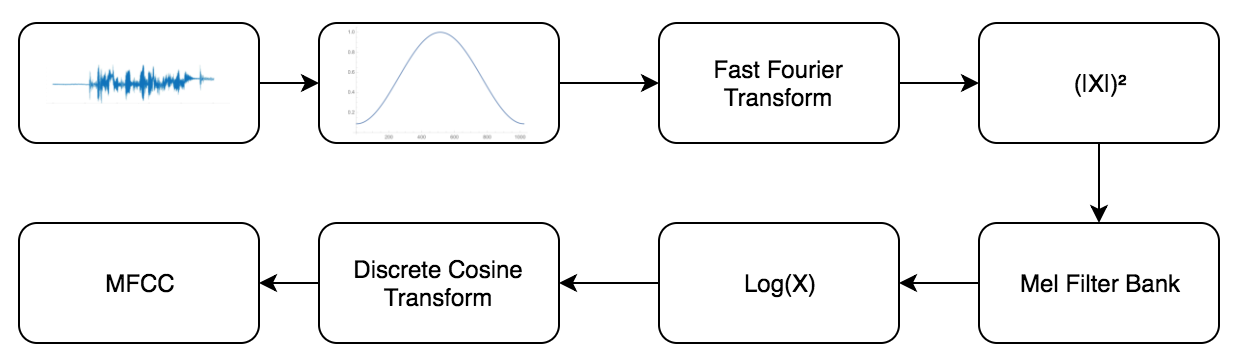


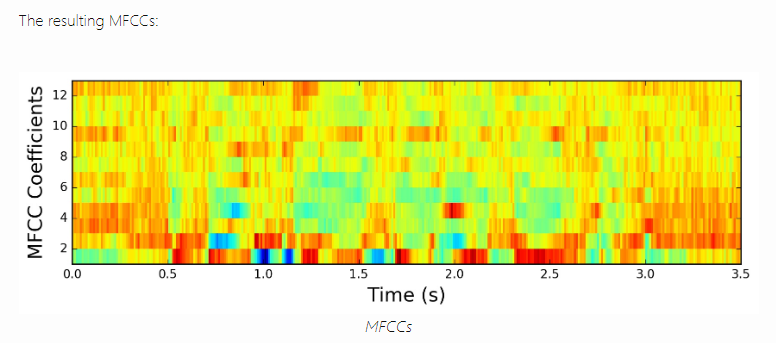
Similar to spectrograms, how each sound takes a unique shape based off of the sound it actually produces. The guitar (which is longer in length than the kick and snare) resonates outwards more than the other studied sounds. Intuitively, this should make sense as when one plays the guitar, the strings that were strummed are still vibrating even after being played, which is how this resonating structure is being portrayed. The kick drum, has a quite low and immediate sound. You can think of the kick drum as a sort of thump. The snare, is quite high frequency and while slightly resonates outward (and more so upwards,) dissipates quicker than the other sounds.

MFCC:

In the conventional analysis of time signals, any periodic component (for e.g. echoes) shows up as sharp peaks in the corresponding frequency spectrum (i.e. Fourier spectrum. This is obtained by applying a [Fourier transform](https://en.wikipedia.org/wiki/Fourier_transform) on the time signal). On taking the log of the magnitude of this Fourier spectrum, and then again taking the spectrum of this log by a cosine transformation we observe a peak wherever there is a periodic element in the original time signal. Since we apply a transform on the frequency spectrum itself, the resulting spectrum is neither in the frequency domain nor in the time domain and hence it is in the quefrency domain. And this spectrum of the log of the spectrum of the time signal was named cestrum. This result creates a spectrum over Mel frequencies as opposed to time, thus creating MFCCs

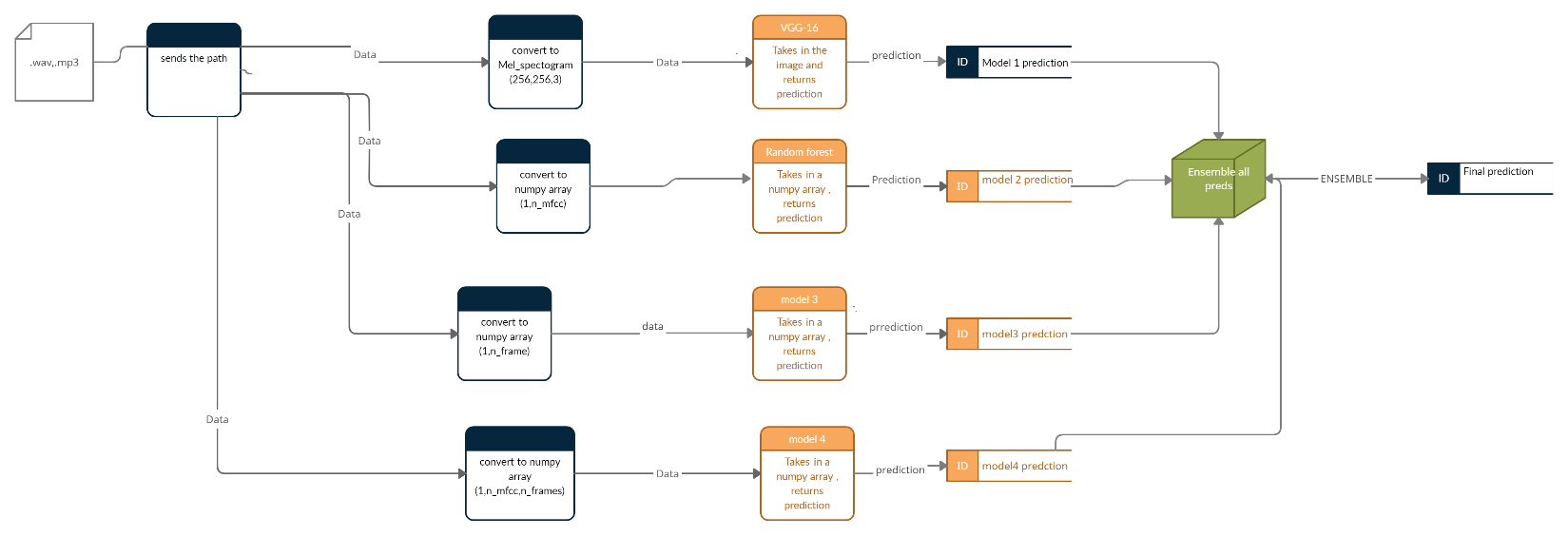






As Speech Emotion Recognition is a classification problem, all these features were taken into consideration to create several Machine Learning models .The evaluation metrics used were Accuracy and F1 score. Hence, various levels of accuracy were achieved by these models.

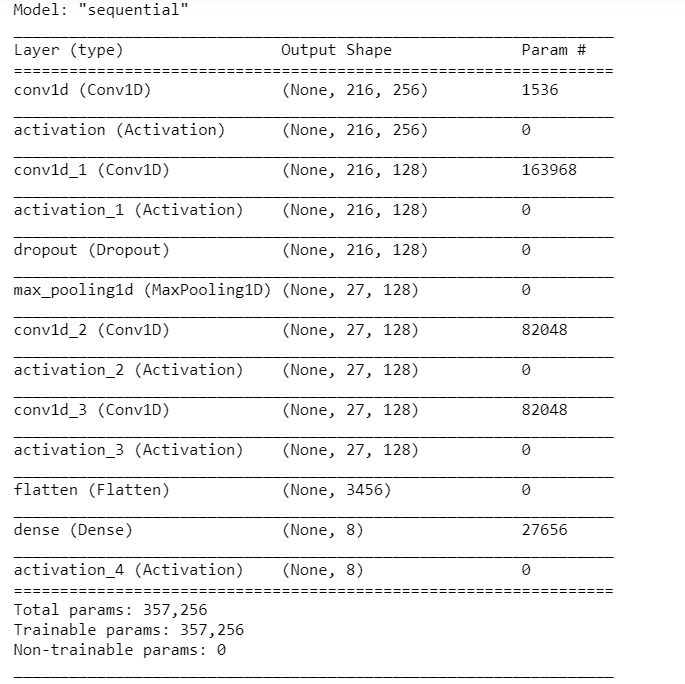
**Model Pipeline:**

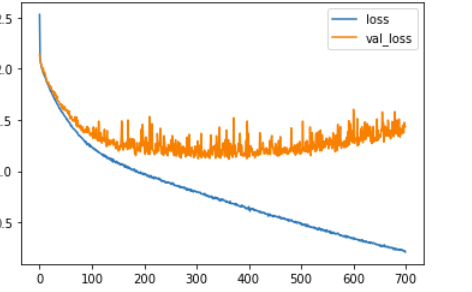
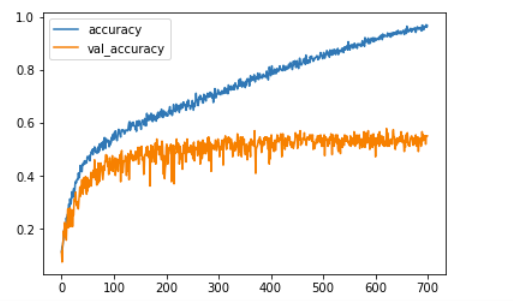
****

**Model 1:**

This is a base model. The audio data is converted to a NumPy array using Librosa. Load package with a sampling rate 0f 44100HZ. The Mel frequency cepstral coefficients are obtained and are averaged over time. i.e. the avg of all the Mels for that time frame. the shape of the output is (n\_examples, n\_frames).For the base model the output labels were: happy ,sad ,calm, angry for both male and female.

**Model Architecture:**





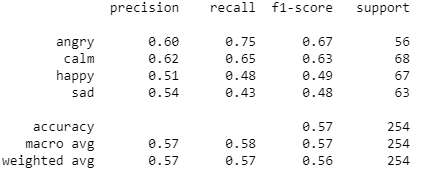
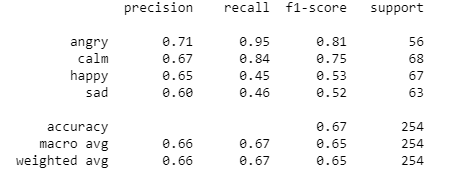
**Model 2:**

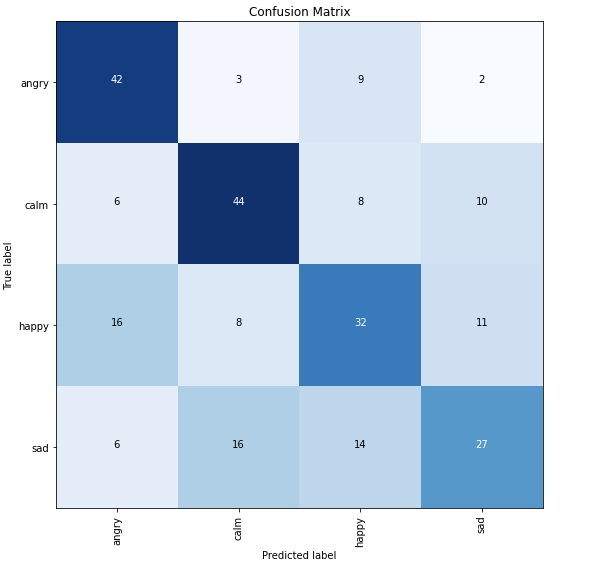
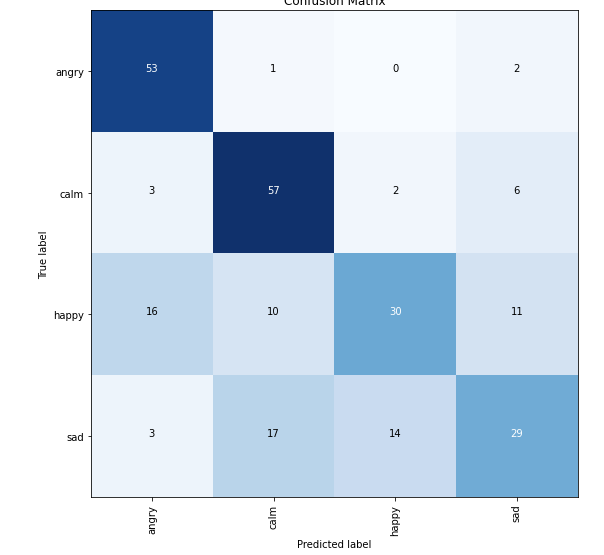
Instead of averaging over the Mel frequencies, these model average over all the time frames. i.e. the average of a particular frequency over the entire time period. output shape will be -(n\_examples,n\_mels).

Decision tree classifier and Random forest classifier are the models used to train this data.

The results obtained are listed below.

Decision Tree Classifier Random Forest Classifier

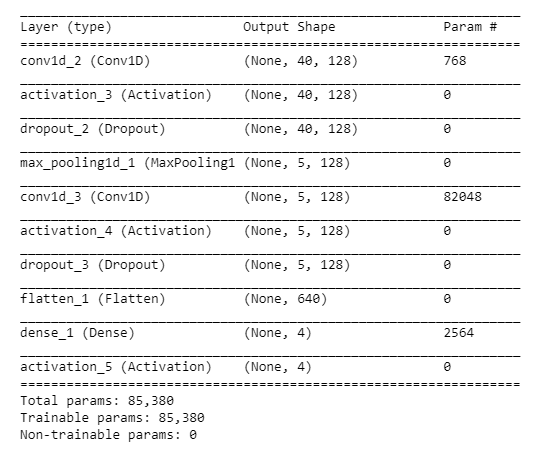
 

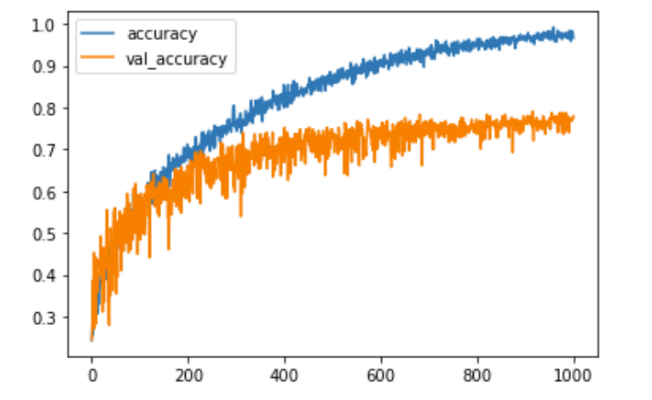
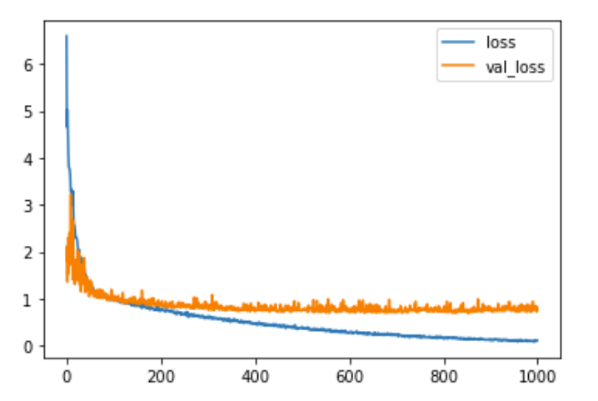
**Model 3:**

To experiment , a deep learning model is used to train the model. This model has two convolution layers to extract one to extract high level features and other to extract level features. Dropout Technique is also used to counter over-fitting. significant results were obtained.

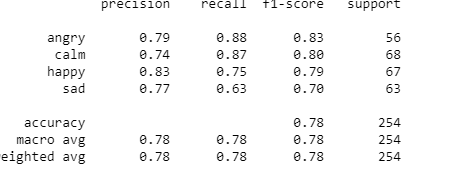
MODEL ARCHITECTURE



Results:

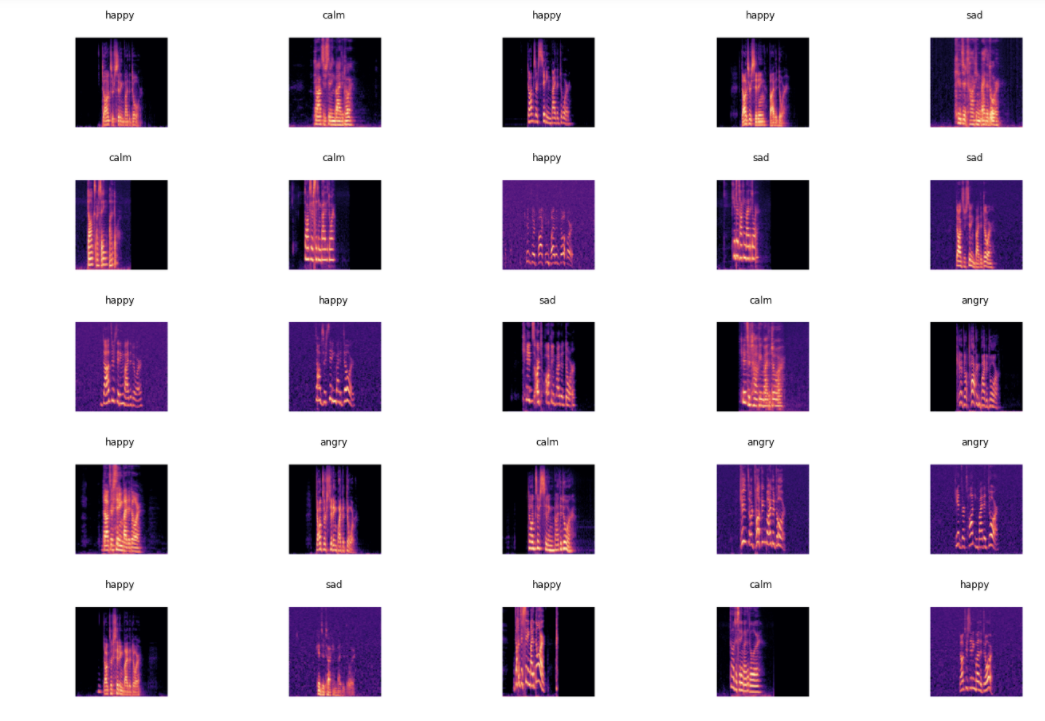


Classification Report:

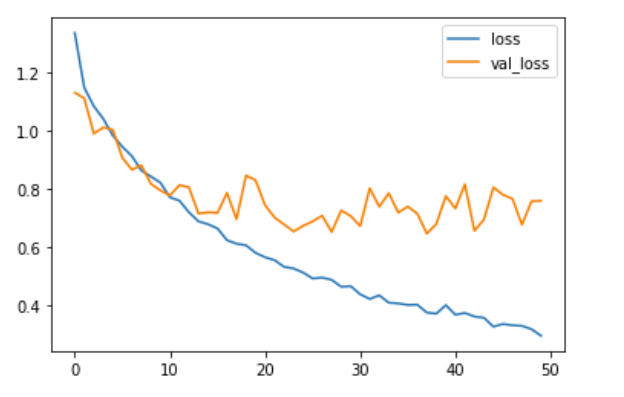
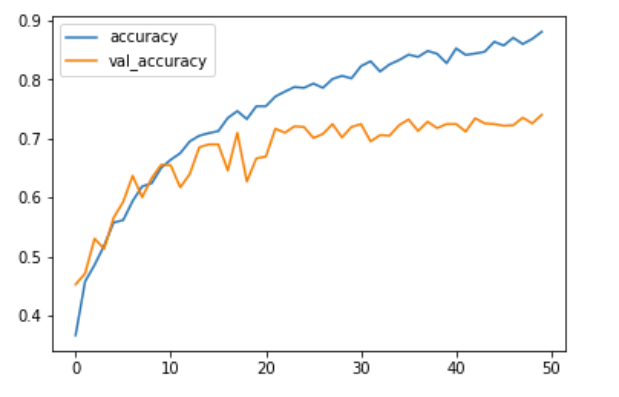


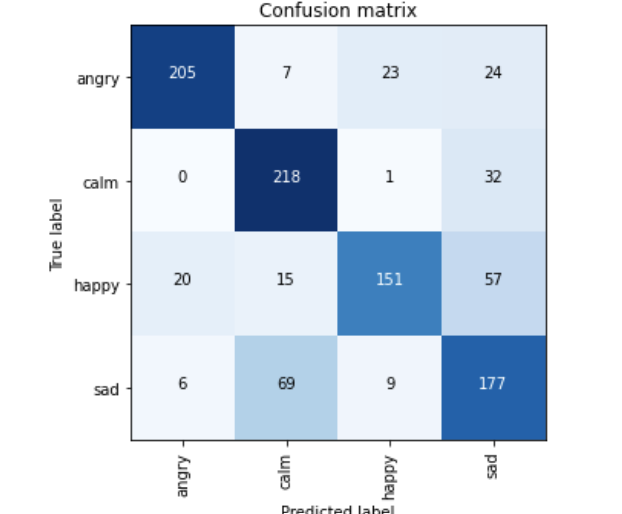
**Model 4:**

Instead of directly using the mfccs from the audio, a Mel-spectrogram image is generated for each audio with a fixed sample rate. A VGG-16 model is used to train on the images. Also, some noise , speed and pitch is also added to the audio. This is because to replicate to the real world audio and also this approaches increases the dataset. These are the images of the Augmented audio files.

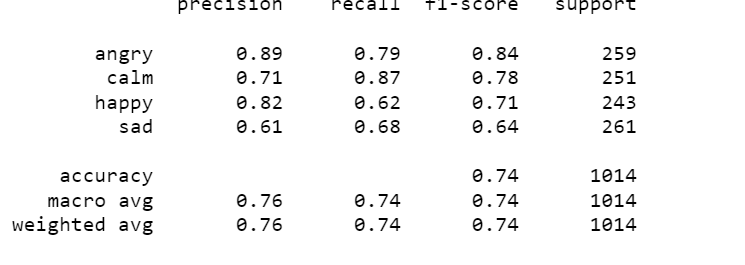


**Results:**



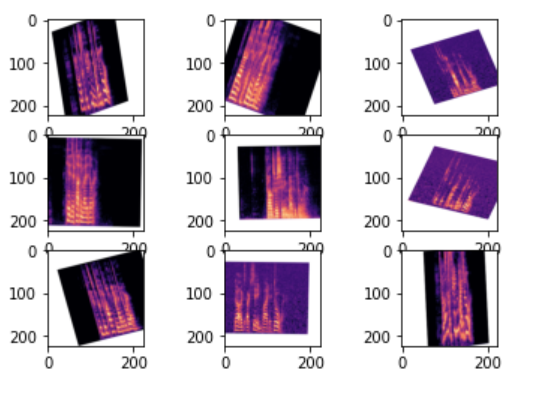


**Classification report:**

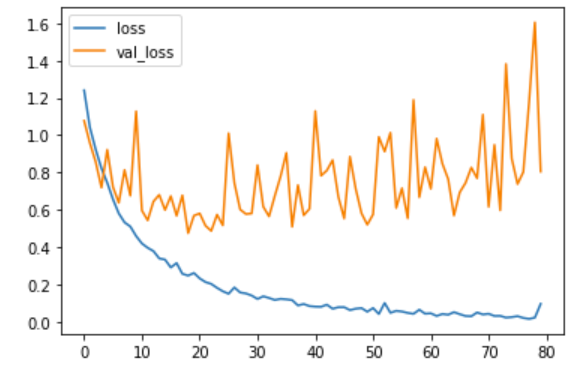
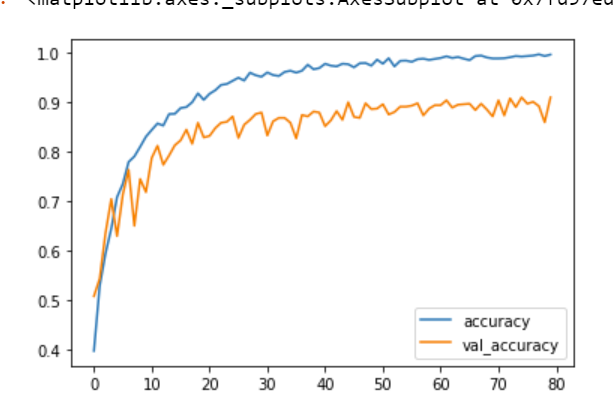


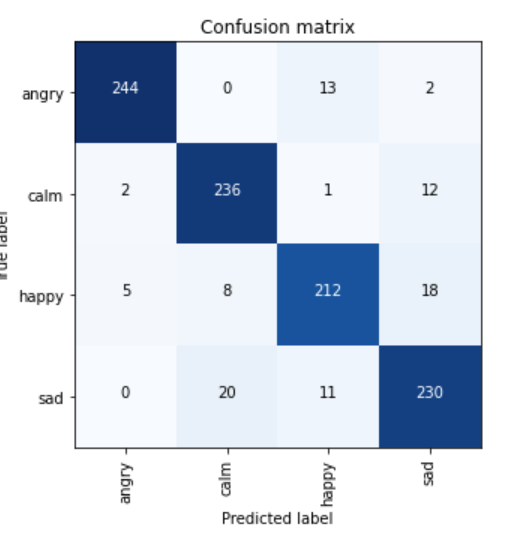
**Model-5:**

This model uses the same architecture as above, but uses ImageData generator instead.This creates multiple Augmented images of the dataset which can improve the metrics.These are the augmented images of the audio files.

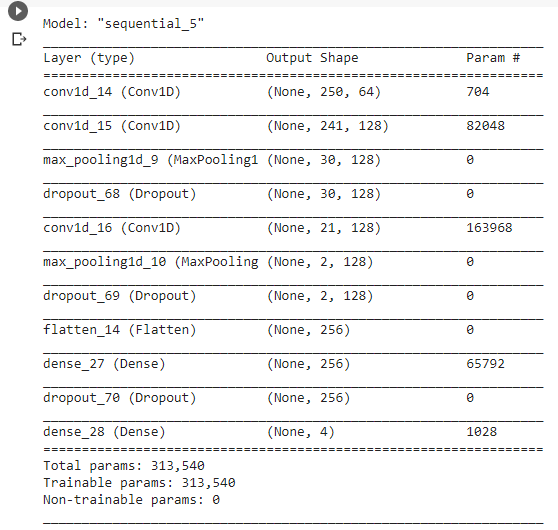


Results:

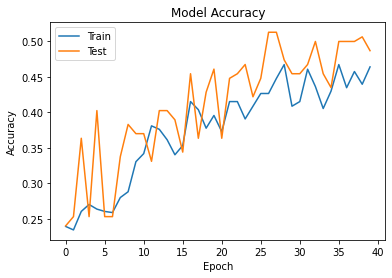
 



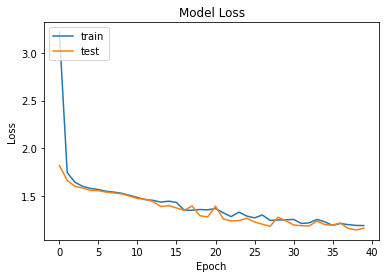
**Model-6 :1D(Mel Scale) : In** this architecture we used mel scale feature to train out deep neural network. First we extracted the mel scale feature and then took mean of that mel scale feature vectors and get final feature vector of 1D.



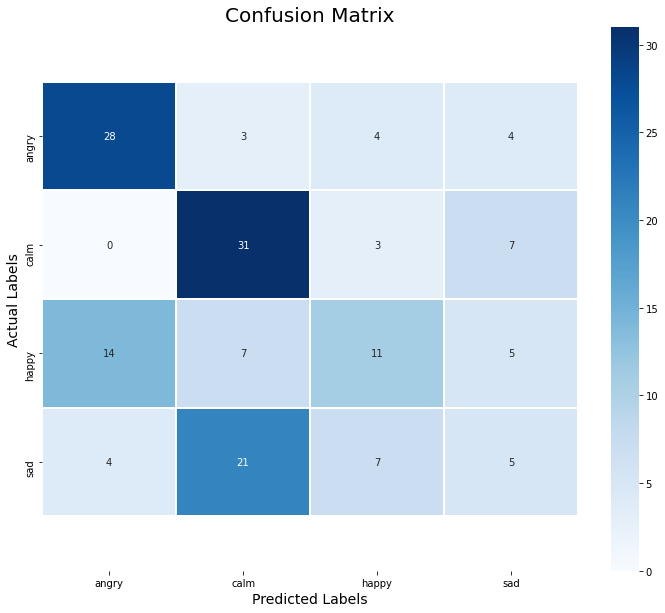
**Accuracy:**

****

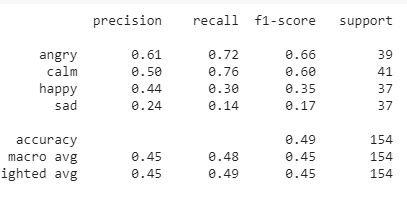
**Loss:**

****

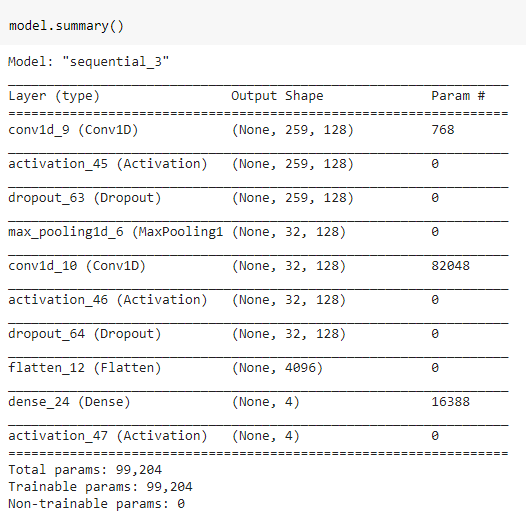
**Confusion matrix:**

****

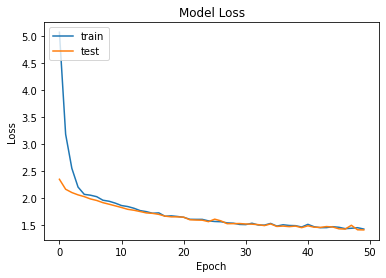
**Prediction report:**

****

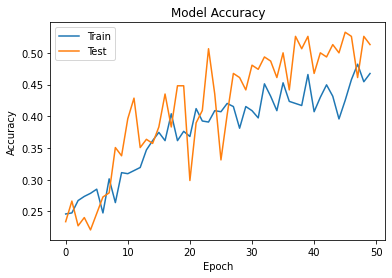
**Model-7:1-D( feature:mfcc):** In this architecture we used MFCC features to train our model. Here also we perform mean operation on the mfcc feature to get 1D feature.



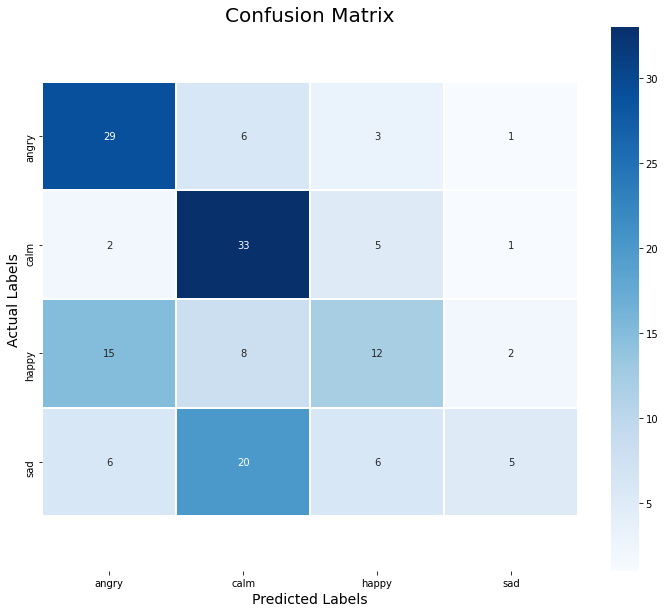
**Loss:**

****

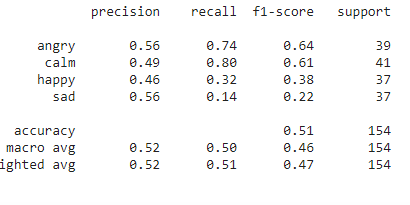
**Accuracy:**

****

**Confusion Matrix:**

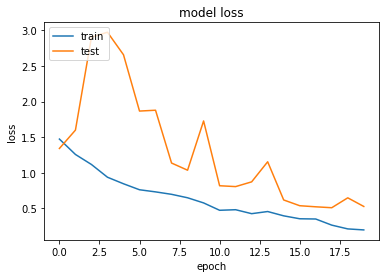
****

**Prediction report:**

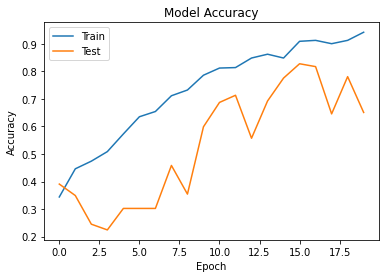
****

**Model-8: 2d feature (mfcc):** In ths architecture we used MFCC to train the model but here we aare not taking mean of the mfcc 2D feature vector. We directly used 2D feature of MFCC and pass it to deep neural network.

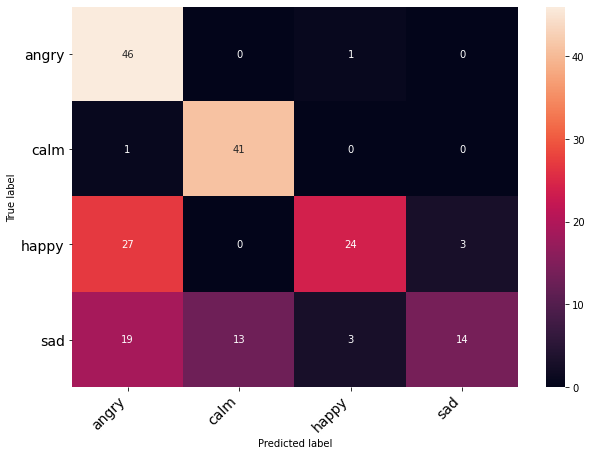
**Loss:**

****

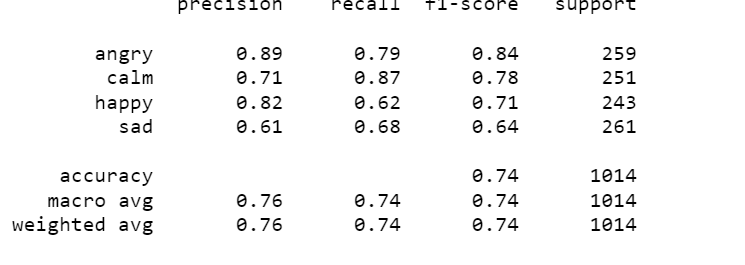
**Accuraccy:**

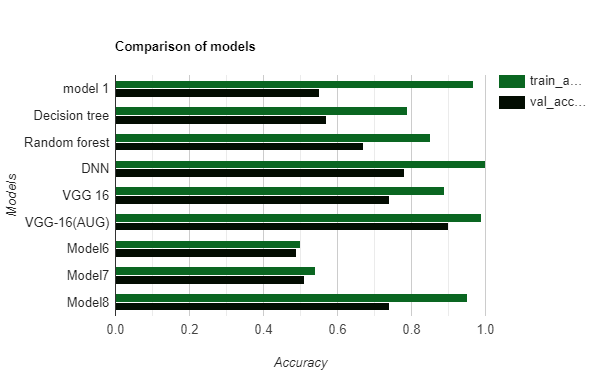
****

**Confusion matrix:**

****

**Classification Report:**





**Analysis:**

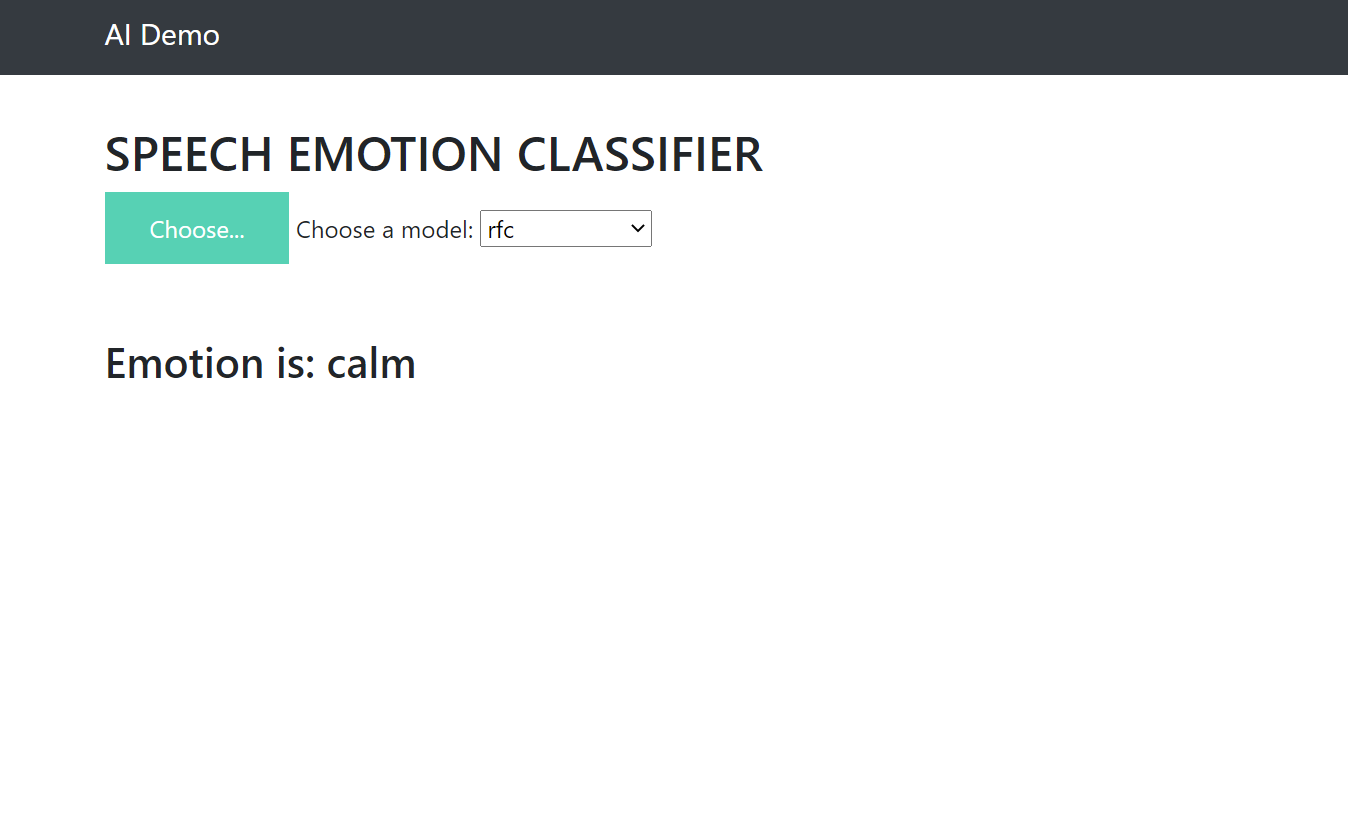
Random forest classifier and Decision tree classifier were able to achieve 75% accuracy but after experimenting with various models it was evident that Deep learning models provide better accuracy. This is because they tend to capture hidden features.

Although deep learning models perform better, they might raise the problem of overfitting because of the limited data. In order to get a good Accuracy, correct features from the audio should be extracted. Model 1 was overfitting because the average Mel frequency for each time frame was taken into consideration, hence for a 3-4 second audio, there are approx. 250 frames. This has transformed the data into more features and fewer examples which ultimately led to overfitting. Therefore, instead of calculating the average Mel frequency per frame, finding the average of each frequency over all the frames seemed more productive as it improves accuracy and reduced overfitting.

In order to increase the dataset, some noise and speed were also added to the copies original data. image data generator was used to create Augmented images of the MEL spectrograms. Finally, the VGG-16 model was used to train the data. This has led to an accuracy of 0.99 on the train set and 0.89 on the test set. Finally, the four best models were included for deployment.

**Deployment:**

Once the best four models have been finalized, A rest API has been created using Flask. The Rest API was designed in such a way that it takes an audio file and a model as input and returns the emotion of the audio file. The second input is optional, if no model has been selected then the audio file will be passed to all the models and the prediction with maximum vote becomes the final Predicted Emotion. The Flask APP is then deployed on the AWS ubuntu machine. Deploying on AWS is just as simple as deploying on a local machine and it gives control over the machine which can help in debugging errors.



**Challenges:**

As This project is not a stand-alone data science project, it required little to medium domain knowledge. Understanding Mfccs, Mel scale is very important for feature selection. secondly, data is very limited, the model metrics can be improved with little more data. Also, the dataset is not the complete representation of real-world data, real-world data has noise, every other person has his /her slang and pace. Therefore it was impossible to generalize based on 24 actors. . Although these models perform well, they tend to confuse between calm and sad. This is obvious as these two emotions sound similar. For better classification, more data would be helpful.

**Limitations:**

AWS Linux free tire has 1 GB ram, which makes it difficult to deploy large models. since speech is emotion in time rather than space, experimenting with RNN's would be helpful. In the future, we would like to experiment with Lstm's on many more emotions. Although there are various pre-trained models like YAM net which extracts embeddings of 1024 dimensions, this approach didn't help much. Secondly, during deployment in Heroku and Azure, there were problems as some additional libraries needed to installed. After installing these libraries , some of previous requirements had to be downgraded in order to be compatible with the installed ones.

**Conclusion:**

After building numerous different models, we have found our best VGG 16 model for our emotion classification problem. We achieved a validation accuracy of 85% with our existing model.