Team *Mood Miners*
Samarth Sharma, Alexis Ambriz, Joash Muganda, Justin Cho
[Project Github Link]

Sentiment Analysis of Audio Files using SEMMA Methods

Abstract

Goal

This sentiment analysis project involves creating a system that analyzes emotions from audio files in .wav format. As our system is taking the form of a Neural Network (NN) model that will be trained to predict the emotions, we will be following the SEMMA (Sample, Explore, Modify, Model, and Assess) methodology. This methodology is known to be useful for evaluating the creation of our model, starting from the data that is being sampled, exploring the features in the dataset, modifying the original data to pre-process the data for use in our model, and finally - assessing the strengths and weaknesses of our model and returning to previous stages for the betterment of the overall system.

Summary of Methods

I. Sample

In the sample stage, the raw data was accessed from the original zip file, loaded, processed to CSV, and then removed. A script was created in the sample directory to process all 6 Audio Actors' Speech data zip files.

II. Explore

The explore stage is where the data from the sample stage is visualized and inspected for initial insights and feature relevance to our model objective.

III. Modify

Targeted features were selected from insights from the explore stage and then statistically analyzed, as well as new features generated during the modify stage.

IV. Model

Emotional, pre-processed data prepared during the modify stage is used to train an ensemble random forest model (RFNN) & gradient boosting Model (GBNN) using the Sci-kit Learn package.

V. Assess

At the final stage, we evaluate the model predictions on a test dataset using a variety of metrics.

- One GBNN was trained & tested to predict the multi-class 'Emotion' feature.
- Two RFNN were trained & tested to predict the multi-class 'Emotion' feature and the binary 'Emotional Intensity' feature.

Results

Our best model (the Random Forest Neural Network) showed ~79% (using the weighted average. f1-score) metric on the test dataset of the 'Emotional Intensity' feature.

The confusion matrix metrics for the model had a weighted avg. recall of ~79% an avg. precision of 79%. This

means our model has a low false positive rate (meaning it is not mistaking normal for high intensities).

To evaluate the difference between the two classes, we used the purity metric. We found a purity score of ~73%, showing that the model is very sure that the classification's it made belong to the correct class.

Given this is a binary classification model, it made sense that this model had higher purity scores than the other two models that were trained to predict the multi-class 'Emotion' feature. In a multi-class setting, there are many more possibilities to consider, and individual certainties for each class is very low.

Introduction

Our guiding purpose for creating a neural network was to understand how the .wav audio files of actors' emotional speeches convey their emotions. Understanding the emotions of others' speech patterns is a fundamentally human task, however, the human brain - although powerful - has limits to the amount of attention and focus it can apply to understanding the emotions of large data. To undertake this challenge, we considered many models, with our best model being an ensemble method using a Random Forest Neural Network (RFNN) trained to predict 'Emotional Intensity', which had a weighted average F1-Score of ~79% during testing on the subset of testing data. Emotional Intensity was marked as a 1 if intensity was 'Normal' or 2 if intensity was 'Strong' for that particular emotion. Knowing which observations are likely 'stronger' can potentially help us build a more fine-tuned model that can detect the 'Emotion' feature later on.

Before landing on the RFNN to predict 'Emotional Intensity', we experimented with 2 other models on 'Emotion' - another RFNN, and a GBNN. We noted that the GBNN had a much higher purity score than the RFNN (~57% and ~35% respectively), with only a slightly lower f1-score (~62% and ~64% respectively). We chose the RFNN model due to its' nature of aggregating the predictions of many decision trees trained on a subset of the data. This ability allows our final predictions to have a lower training model bias as we are training a more complex model than when using a single network, which may come in handy on a multi-class dataset with a lot of complexity.

Low bias might introduce greater variance in the model (overfitting on the training data), resulting in a low purity score due to uncertainty in the testing data classes. To tackle this problem & increase the strength of our predictions, we saw value in shifting our focus to a simpler problem first, like predicting the binary class 'Emotional Intensity', using the same RFNN model architecture.

Related Work

Let's compare the findings and insights and architecture of our model retrospectively. A look at the journal article 'Multimodal Emotion Recognition on RAVDESS Dataset Using Transfer Learning' by Luna-Jimenez et. al. mentions using multi-modal audio-visual emotion data from the same source as our project (RAVDESS). A key difference between the authors implementation and ours is that their model takes into account not only the audio data, but also takes into account the visual emotional cues (Luna-Jimenez et al. pg. 1). Incorporating different data types and sources may improve the model, as in the real-world, emotions are reflected not only through voice - but through facial expressions.

Another key difference is that the model they implemented is known as a Spatial-Transformer Neural Network (STNN) and ours is a Random Forest Neural Network (RFNN). Due to the difference in the structure of their data, it seems that they have chosen a more suitable model to complement it. For example, the authors mention their implemented system is trained on 'saliency maps' and included the following is a diagram of their overall system:

Their model architecture includes two pre-processing stages (one for each data type, audio & visual).

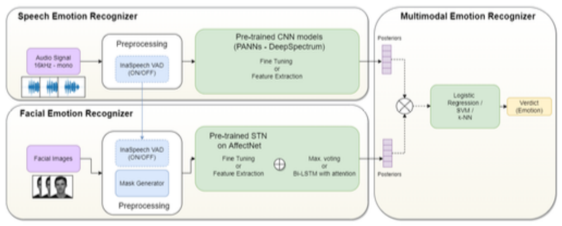

Speech Emotion Recognizer
Audio Signal 16kHz - mono
Preprocessing
InaSpeech VAD (ON/OFF)
Pre-trained CNN models (PANNs - DeepSpectrum)
Fine Tuning or Feature Extraction
Facial Emotion Recognizer
Facial images
InaSpeech VAD (ON/OFF)
Mask Generator
Preprocessing
Pre-trained STN on AffectNet
Fine Tuning or Feature Extraction
Max. voting or Bi-LSTM with attention
Multimodal Emotion Recognizer
Posteriors
Posteriors
Logistic Regression / SVM / k-NN
Verdict (Emotion)

Figure 1. Block diagram of the implemented systems.

Further, it seems that their STNN is actually an ensemble model containing three neural networks, one for each data type - and a final prediction model. The models the authors implemented include:

1. A Pre-trained Convolutional NN model (PANNs - DeepSpectrum) for the Audio data
2. A Pre-trained STN (Spatial Transformer Network) for the Image data
3. A final Logistic Regression / Support Vector Machine / k-Nearest Neighbors Machine Learning transformation on the posterior output of the two models.

The authors mention it was able to achieve 80.08% accuracy on classifying the 8 different emotions.

Some take-aways from this architecture is that the use of two neural network models in combination with three machine learning methods to combine the predictions of the models can significantly improve results on multi-class classification. Implementing a similar mix of models may assist us in improving our own model.

Data

The system considers several features that define the context and content of the audio sample. These features were included during the pre-processing stage ‘’01_Sample” as categorical features along with the audio data in the filename string:

- **Modality**: Identifying whether the audio is accompanied by video (Vocal Channel: Audio-Visual, video-only, or purely audio).
- **Emotion**: Categorizing the emotion conveyed in the audio file, ranging from neutral to surprised, including calm, happy, sad, angry, fearful, and disgust.
- **Emotional Intensity**: Measuring the strength of the emotion, with a note that 'neutral' does not have a 'strong' intensity level.
- **Statement**: Recognizing the content of the audio, whether it's a predefined sentence about kids or dogs.
- **Repetition**: Noting the number of times the statement is repeated to assess consistency and variation in emotional expression.
- **Actor**: Accounting for gender differences in emotional expression by identifying the actor, with odd and even numbers representing male and female actors, respectively.

We also incorporated new numerical features engineered using the Python library Librosa, which is capable of extracting valuable information from the audio .wav files of the actors’ emotional speech data such as:

- **Spectral Centroid:** The spectral centroid is the mean frequency of the spectrum of a sound. It is calculated by treating each frame of a magnitude spectrogram as a distribution over frequency bins, from which the mean (centroid) is extracted per frame.
- **Spectral Bandwidth:** The spectral bandwidth at a given frame is computed by summing the product of the spectrum and the difference between the frequency and the centroid, raised to a power (p), and then taking the pth root of the result
- **Spectral Rolloff:** The spectral rolloff is the frequency below which a certain percentage of the total spectral energy lies. It is defined for each frame as the center frequency for a spectrogram bin such that at least a specified percentage of the energy of the spectrum in this frame is contained in this bin and the bins below
- **Zero Crossing Rate:** The zero crossing rate is the rate at which the signal changes from positive to negative or vice versa. It is calculated by counting the number of times the signal crosses the zero line within a given frame.
- **Chroma STFT:** Chroma STFT is a representation of the spectrum of a sound that considers the pitch content of the sound. It is computed by taking the Short-time Fourier Transform (STFT) of the signal and then mapping the frequency bins onto a chroma vector.
- **MFCC:** Mel Frequency Cepstral Coefficients (MFCCs) are a type of feature used in speech and audio processing. They are derived from the Mel scale, which is a perceptual scale of pitches. MFCCs are used to capture the short-term power spectrum of a sound, with the coefficients providing a compact representation of the power spectrum.
- **RMSE:** Root Mean Square Error (RMSE) is a measure of the differences between values predicted by a model and the values observed. It is used in many fields, including signal processing, to quantify the accuracy of a model.
- **Onset Envelope:** The onset envelope is a measure of the intensity of a sound. It is computed by taking the difference between the signal and a low-pass filtered version of the signal. The result is a measure of the intensity of the sound, with onsets (changes in intensity) being highlighted.

Finally, another processing step we took to make our data (taken from Librosa) suitable for entry into our model is converting the multi-dimensional features, such as MFCC or Chroma STFT, or the raw audio signal (y), into constant numerical features for all the observations. We did this by reducing them by applying mean, std, and median, as well as selecting the min/max. This meant that for each Librosa feature listed above, ~ 15 new dimensions were added. The final shape of the dataset, before splitting into train and test and using it within our model, was (1440, 137). We used 80% of this data to train, and 20% to test.

As our model (RFNN) requires a fixed-length number of dimensions for all features, including multi-dimensional features would require the use of more complex model architectures.

Potential alternative ideas that we explored, but did not implement, include a DeepSets Convolutional Neural Network or a Graph Convolutional Neural Network. These models accept multi-dimensional datasets (with observations that vary in the number of dimensions).

Methods

SEMMA

We decided to use a SEMMA (Sample, Explore, Modify, Model, Assess) project architecture for our project because we are creating a model for exploration and testing of Emotional Audio data, and we were unsure of the final model architecture or the features we would hope to predict - as our dataset included many potential emotional indicators.

Therefore, as we are in the exploration and development phase, we were not actively looking to incorporate our model into a business process. An alternative project architecture that we may consider in the future if the goal were to center our model and project within a business organization is the CRISP-DM (Cross-Industry Standard Process in Data Mining) architecture. The CRISP-DM process is cyclical and communicates many of the stages of the process to business stakeholders with the addition of a further evaluation stage that would seek input from other business teams at the end of the SEMMA 'Assessment' stage during each CRISP-DM iteration.

CRISP-DM would be a great choice if we were to scaffold the work we have already done and improve upon it, as SEMMA is a subset architecture of CRISP-DM, many of the steps would already be implemented. CRISP-DM offers a benefit to businesses as it includes an extra stage focused on understanding the business, and aligning the project according to business objectives.

In the case that we were looking to begin using our model for a certain business process in an organization, a different model architecture may be chosen and new data may be included as well, potentially influencing the model created for the improvement of business KPI (Key Performance Indicators).

Sample

We used two different methodologies for sampling the data. The list features generated by the librosa library in python needed to be converted to their list representations using the json library for easier input and output. The method when used in google colab relied on loading the data directly each time from the raw csv and then converting it to a constant directly, for example, by processing each of the lists in the 'list of list' (multi-dimensional columns) individually and including them as columns, whereas the method when used in a local environment relied on converting the data into json objects and then converting it into a constant (across all features) which reduced the accuracy but simplified the processing step (one column could be expanded to ~15 new columns when processed in the colab notebook). This difference in data loading leads to a significant difference in overall results because the structure of the data is more accurate in the Colab notebook versus the local notebook. Employing the SEMMA methodology in this case helped us reiterate our model quickly as we could apply quick changes at earlier stages from differences at later stages within different branches (colab vs. local).

Explore

Seven attributes were extracted from metadata out of which two of them were target variables: Emotion and Emotional Intensity. The remaining 5 features had either constant value throughout or negligible correlation with the target variables and thus should be dropped. Furthermore, target variables have a low correlation (0.11), and thus different classifiers should be built for recognizing emotion and emotional intensity. Moreover, all the other features extracted from raw data are in the form of a list or list of lists and have to be in some way converted to numeric data so that it can be used to train machine learning models. Also, all the features have been derived from encoded raw audio 'y' and sampling rate (constant throughout) and thus keeping in mind the curse of collinearity must be dropped. Atlas, emotions (1 = neutral, 2 = calm, 3 = happy, 4 = sad, 5 = angry, 6 = fearful, 7 = disgust, 8 = surprised) are encoded from 1 to 8, but the data is nominal (not ordinal), thus we can consider one hot encoding it. On the other hand, emotional intensity (1-normal, 2-strong) is a binary classification task so there's no need for any encoding.

Modify

As the data extracted from raw audio 'y' and sampling rate are in the form list (spectral features, rmse, etc) or list of lists (mfcc) statistical summary (mean, median, standard deviation, min, max) representation is retained to convert it to numerical form while retaining the essence of the data. Also as discussed a lot of attributes were dropped due to either less correlation or static values. Furthermore, the data is normalized and one hot encoding isn't performed for the emotion class label (one hot encoding isn't working as per experimentation probably because there might be an ordinal relation in emotions or the target becomes a sparse matrix leading to difficulties in the mapping).

Model

This phase started with the train-test split of 80-20. Then experimenting with different algorithms and hyperparameter tuning, random forest classifiers performed the best for both the classification tasks. Accuracy for emotion classification came out to be 64.58 percent which is still not bad considering there are 8 classes and random predictions would result in accuracy of around 12.5%. For emotional intensity classification with random forest classifier accuracy came out to be 79.86 percent .

Assess

In this assessment, we review the performance metrics of two neural network models – the Gradient Boosted Neural Network (GBNN) and the Random Forest Neural Network (RFNN) – in the context of emotion and emotional intensity detection. This evaluation is based on various figures detailing their Receiver Operating Characteristic (ROC) and Area Under the Curve (AUC) metrics, as well as confusion metrics.

I. GBNN for Emotion Detection (Figures 1 & 2):
- ROC/AUC Metrics: The AUC analysis for emotion categories indicates a range from 88% (class 6) to 97% (class 1). The lower AUC in some classes suggests room for improvement in distinguishing between true and false positives.
- Confusion Matrix: The GBNN shows an overall f1-score of approximately 63%. This is a marginal improvement over a random chance (50%), indicating a need for further fine-tuning.

II. RFNN for Emotion Detection (Figures 3 & 4):
- ROC/AUC Metrics: Similar to the GBNN, the RFNN displays AUC values ranging from 87% to 96%, with class 5 being the least accurate.
- Confusion Matrix: The RFNN achieves a slightly higher f1-score of about 64%, which, while only marginally better than the GBNN, suggests a slightly more effective model for emotion detection.

Comparative Analysis

- The close performance metrics between GBNN and RFNN indicate a negligible difference in their effectiveness for emotion prediction. This observation suggests the potential exploration of alternative approaches, such as multimodal transfer learning.

I. RFNN for Emotional Intensity Detection (Figures 5 & 6):
- ROC/AUC Metrics: The RFNN demonstrates a promising AUC of

approximately 86% for Emotional Intensity, significantly better than a random choice.

- Confusion Matrix: The f1-score for Emotional Intensity prediction is around 79%, markedly higher than the scores for basic emotion detection.

Comparative Analysis

The RFNN model, particularly in the context of Emotional Intensity detection, emerges as the more effective model. While both GBNN and RFNN show potential in basic emotion detection, their performances are marginally better than chance. This calls for further investigation and potential integration of more sophisticated methods, such as those involving transfer learning, to enhance prediction accuracy. The relative success in Emotional Intensity detection by the RFNN underscores its potential as a more viable model in this domain.

Experiments and Results

Note: Full size images of the referenced images are available in the 'Supplementary Materials' section near the end of the paper.

Neural Network Experiments

Tried to build sequential neural network models such as ANN and LSTM for the classification of emotion class. But the results were poor (15% accuracy) which in this case was similar to coin flip as there are 8 classes (12.5 % probability of a random guess being correct). These failures are mostly due to lack of knowledge on building a neural network model architecture and fine-tuning a neural network model. So, there is a future scope of trying to solve the problem again with neural networks.

Visualization of ROC/AUC & Metrics

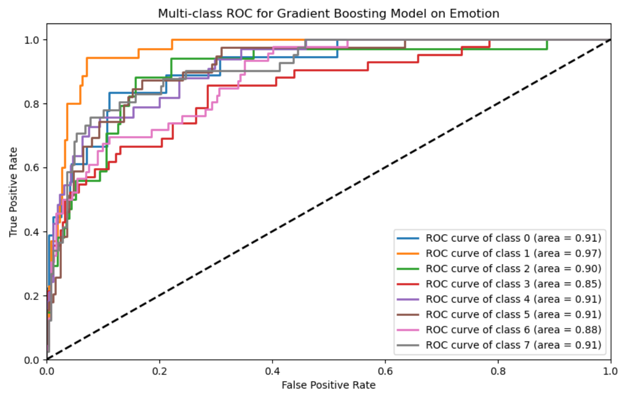
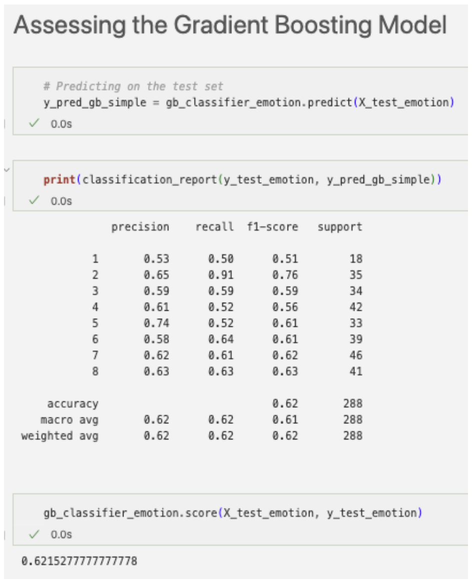


Figure 1: ROC of the GBNN for Emotion

The visualization of the class-wise AUC (Area Under the Curve) for each Emotion category shows that the lowest AUC (more false > true positives) is class 6 (88%), whereas the highest is class 1 (97%).



Assessing the Gradient Boosting Model

```
# Predicting on the test set
y_pred_gb_simple = gb_classifier_emotion.predict(X_test_emotion)
✓ 0.0s
```

```
print(classification_report(y_test_emotion, y_pred_gb_simple))
✓ 0.0s
```

```
              precision    recall  f1-score   support

           1       0.53      0.50      0.51        18
           2       0.65      0.91      0.76        35
           3       0.59      0.59      0.59        34
           4       0.61      0.52      0.56        42
           5       0.74      0.52      0.61        33
           6       0.58      0.64      0.61        39
           7       0.62      0.61      0.62        46
           8       0.63      0.63      0.63        41

    accuracy                           0.62       288
   macro avg       0.62      0.62      0.61       288
weighted avg       0.62      0.62      0.62       288
```

```
gb_classifier_emotion.score(X_test_emotion, y_test_emotion)
✓ 0.0s
0.6215277777777778
```

Figure 2: Confusion Metrics of the GBNN for Emotion

The confusion metrics for the GBNN trained to predict the 8 emotion categories showed an f1-score of ~63%. This is slightly better (13%) than a coin-flip accuracy, which shows potential in fine-tuning the model for greater accuracy.

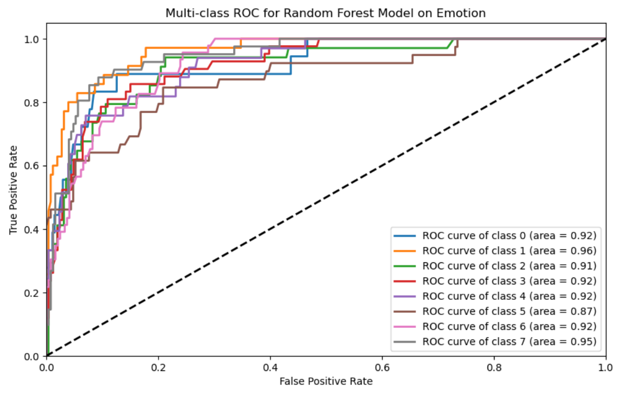
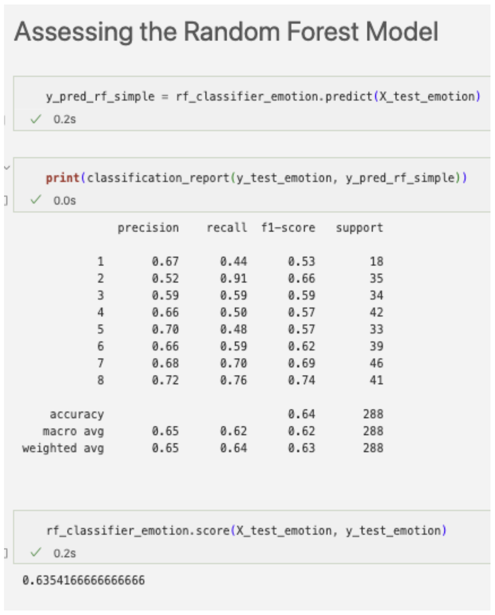


Figure 3: ROC of the RFNN for Emotion

The visualization of the class-wise AUC (Area Under the Curve) for each Emotion category shows that the lowest AUC (more false > true positives) is class 5 (87%), whereas the highest is class 1 (96%).



Assessing the Random Forest Model

```
y_pred_rf_simple = rf_classifier_emotion.predict(X_test_emotion)
✓ 0.2s
```

```
print(classification_report(y_test_emotion, y_pred_rf_simple))
✓ 0.0s
```

```
              precision    recall  f1-score   support

           1       0.67      0.44      0.53        18
           2       0.52      0.91      0.66        35
           3       0.59      0.59      0.59        34
           4       0.66      0.50      0.57        42
           5       0.70      0.48      0.57        33
           6       0.66      0.59      0.62        39
           7       0.68      0.70      0.69        46
           8       0.72      0.76      0.74        41

    accuracy                           0.64       288
   macro avg       0.65      0.62      0.62       288
weighted avg       0.65      0.64      0.63       288
```

```
rf_classifier_emotion.score(X_test_emotion, y_test_emotion)
✓ 0.2s
0.6354166666666666
```

Figure 4: Confusion Metrics of the RFNN for Emotion

The confusion metrics for the RFNN trained to predict the 8 emotion categories showed an f1-score of ~64%, only ~1% difference from the GBNN. This is slightly better (14%) than a coin-flip accuracy, which shows potential in fine-tuning the model for greater accuracy. However, this ablation of the two models shows that there is little benefit in using a gradient boosting model over a random forest, and other methods (such as the multimodal transfer learning STNN previously mentioned in the related work section) should be considered when predicting emotion, a highly complex variable.

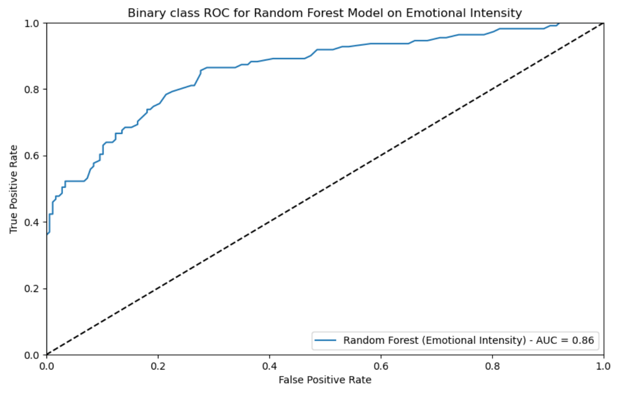
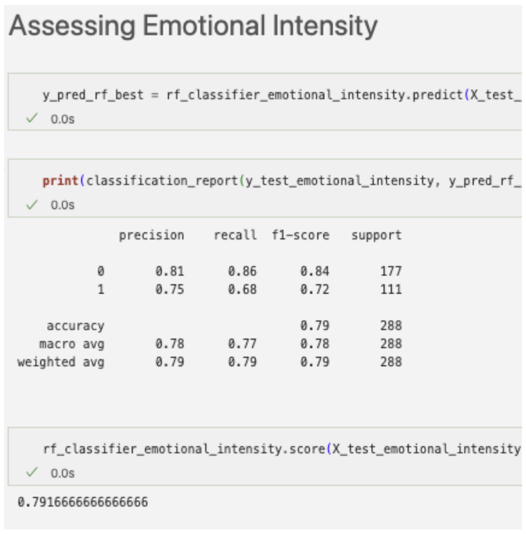


Figure 5: ROC of the RFNN for Emotional Intensity

The visualization of the AUC (Area Under the Curve) for the Emotional Intensity feature shows an AUC value of ~86%. As Emotional Intensity is a binary value, this means that it is able to correctly identify 01 (Normal) intensity from 02 (Strong) intensities with an AUC of 86% meaning that the

RFNN is more likely to assign test classes to their correct assignments with high probability. This is ~36% better than a random coin flip probability (50%), meaning that the model is a better predictor than a random guess.



Assessing Emotional Intensity

```
y_pred_rf_best = rf_classifier_emotional_intensity.predict(X_test_
✓ 0.0s
```

```
print(classification_report(y_test_emotional_intensity, y_pred_rf_
✓ 0.0s
```

```
              precision    recall  f1-score   support

           0       0.81      0.86      0.84       177
           1       0.75      0.68      0.72       111

    accuracy                           0.79       288
   macro avg       0.78      0.77      0.78       288
weighted avg       0.79      0.79      0.79       288
```

```
rf_classifier_emotional_intensity.score(X_test_emotional_intensity
✓ 0.0s
0.7916666666666666
```

Figure 6: Confusion Metrics of the RFNN for Emotional Intensity

The confusion metrics for the RFNN trained to predict the 2 Emotional Intensity categories showed an f1-score of ~79%. This is also better (29%) than a coin-flip accuracy, which shows our best model is the RFNN trained to predict Emotional Intensity.

Conclusion

To conclude, we successfully created machine learning models using the SEMMA methodology to determine one of eight different types of emotion and the intensity of that emotion taking in the raw audio data and processing it.

Future Scope: The problem was tackled using the machine learning algorithms which performed well but still have a good scope for improvement. Neural networks have been proven exceptionally well in the domain of audio-related tasks. So, the problem can be reattempted with a deep learning approach such as Long Short Term Memory is a type of Recurrent neural network and also Convolutional Neural networks.

Bibliography

Luna-Jimenez, Cristina, et al. "Multimodal Emotion Recognition on RAVDESS Dataset Using Transfer Learning." *Sensors*, vol. 2021, no. 21, 2021, p. 29. *Multidisciplinary Digital Publishing Institute (MDPI)*, https://www.mdpi.com/1424-8220/21/22/7665.

Supplementary Materials

Figure 1: ROC of the GBNN for Emotion

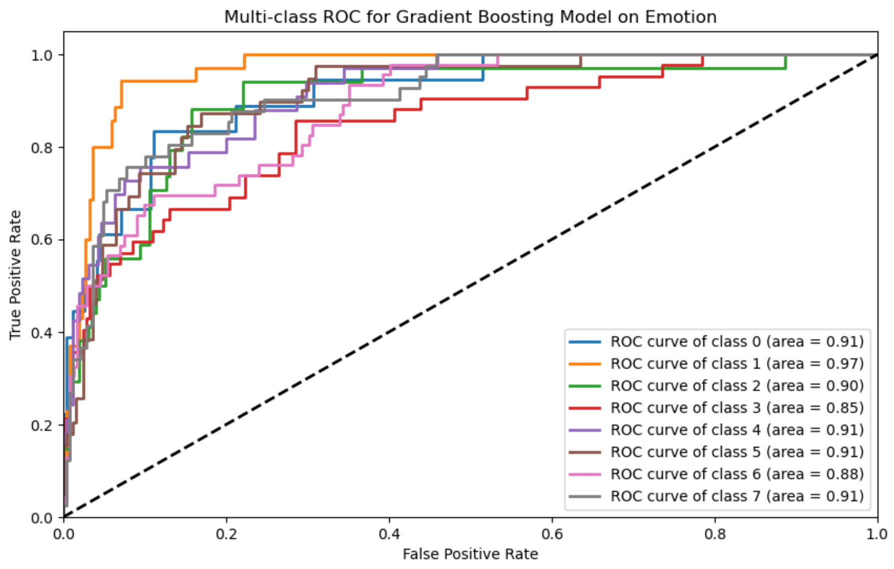
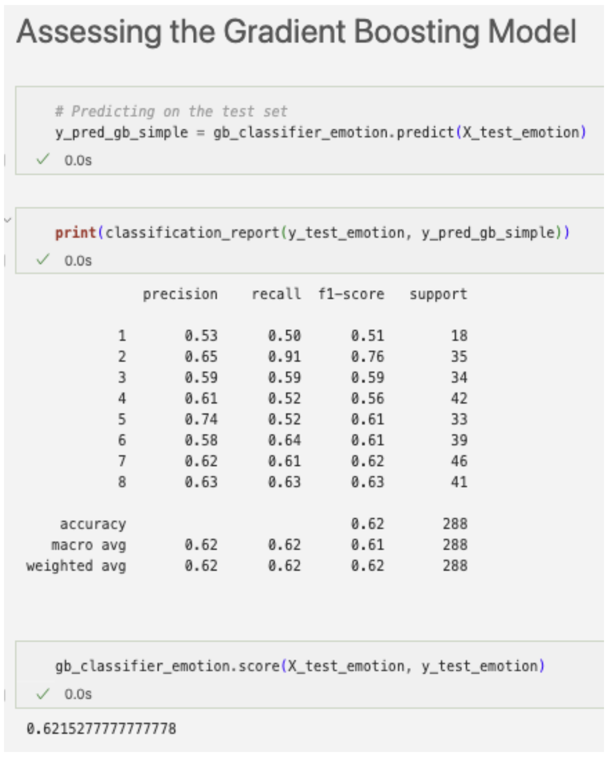


Figure 2: Confusion Metrics of the GBNN for Emotion



Assessing the Gradient Boosting Model

```
# Predicting on the test set
y_pred_gb_simple = gb_classifier_emotion.predict(X_test_emotion)
```
✓ 0.0s

```
print(classification_report(y_test_emotion, y_pred_gb_simple))
```
✓ 0.0s

```
              precision    recall  f1-score   support

           1       0.53      0.50      0.51        18
           2       0.65      0.91      0.76        35
           3       0.59      0.59      0.59        34
           4       0.61      0.52      0.56        42
           5       0.74      0.52      0.61        33
           6       0.58      0.64      0.61        39
           7       0.62      0.61      0.62        46
           8       0.63      0.63      0.63        41

    accuracy                           0.62       288
   macro avg       0.62      0.62      0.61       288
weighted avg       0.62      0.62      0.62       288
```

```
gb_classifier_emotion.score(X_test_emotion, y_test_emotion)
```
✓ 0.0s

```
0.6215277777777778
```

Figure 3: ROC of the RFNN for Emotion

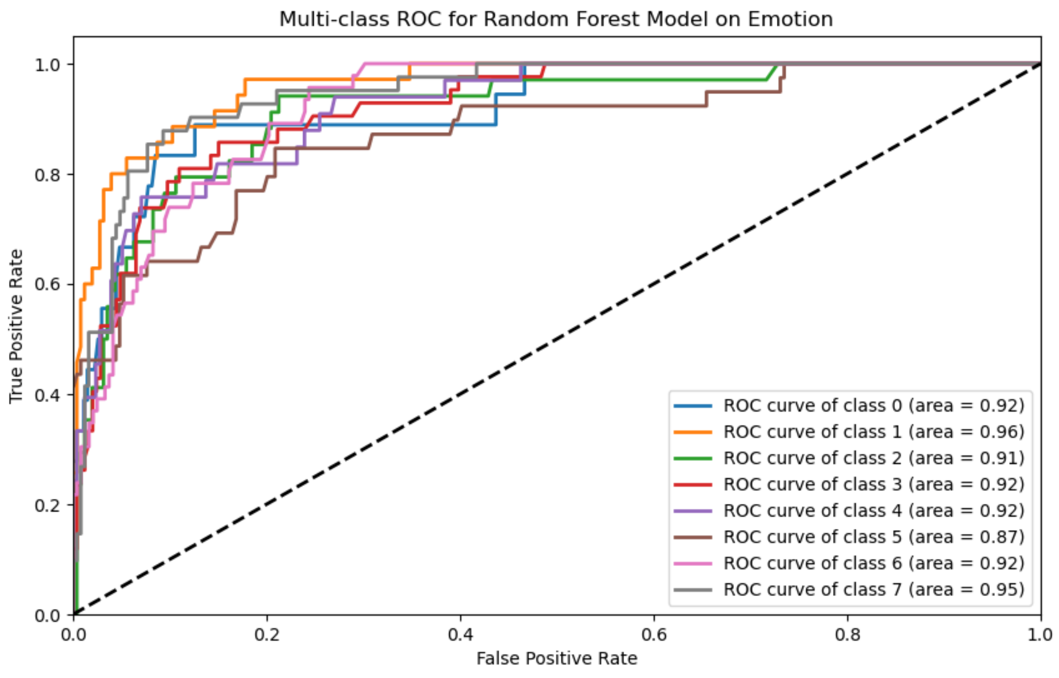
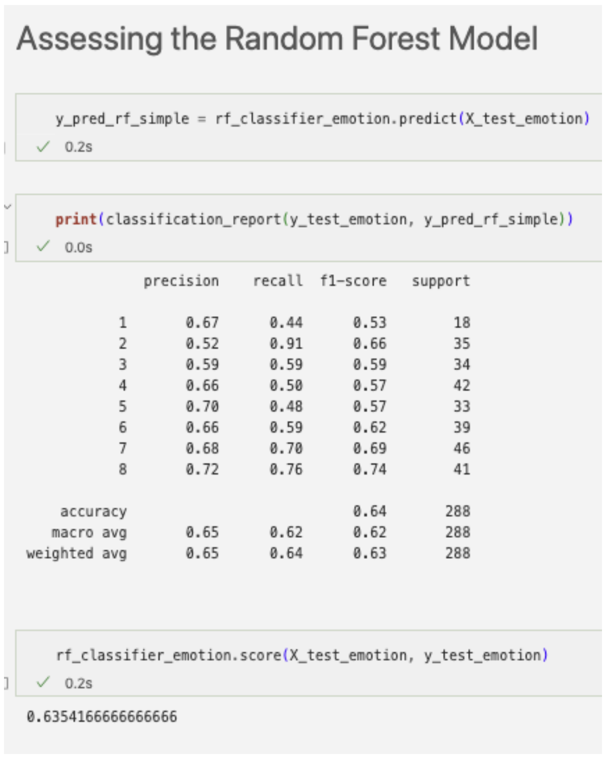


Figure 4: Confusion Metrics of the RFNN for Emotion



Assessing the Random Forest Model

```
y_pred_rf_simple = rf_classifier_emotion.predict(X_test_emotion)
```
✓ 0.2s

```
print(classification_report(y_test_emotion, y_pred_rf_simple))
```
✓ 0.0s

```
              precision    recall  f1-score   support

           1       0.67      0.44      0.53        18
           2       0.52      0.91      0.66        35
           3       0.59      0.59      0.59        34
           4       0.66      0.50      0.57        42
           5       0.70      0.48      0.57        33
           6       0.66      0.59      0.62        39
           7       0.68      0.70      0.69        46
           8       0.72      0.76      0.74        41

    accuracy                           0.64       288
   macro avg       0.65      0.62      0.62       288
weighted avg       0.65      0.64      0.63       288
```

```
rf_classifier_emotion.score(X_test_emotion, y_test_emotion)
```
✓ 0.2s

```
0.6354166666666666
```

Figure 5: ROC of the RFNN for Emotional Intensity

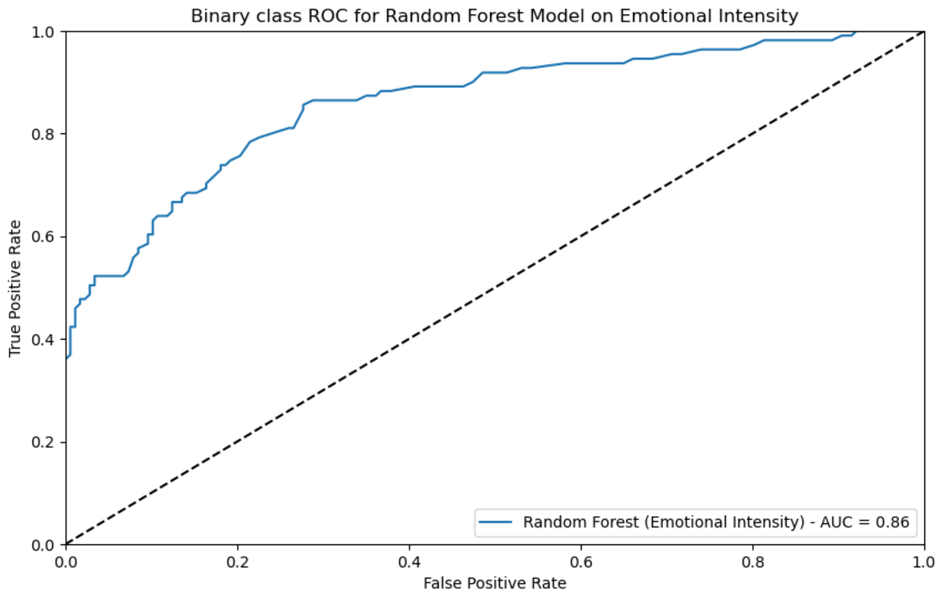
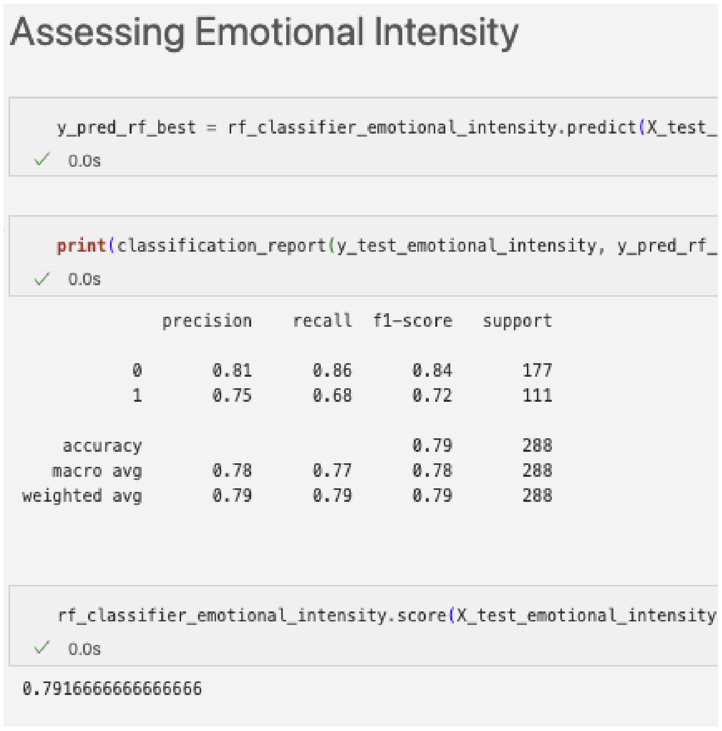


Figure 6: Confusion Metrics of the RFNN for Emotional Intensity



Assessing Emotional Intensity

```
y_pred_rf_best = rf_classifier_emotional_intensity.predict(X_test_
✓ 0.0s
```

```
print(classification_report(y_test_emotional_intensity, y_pred_rf_
✓ 0.0s
```

```
              precision    recall  f1-score   support

           0       0.81      0.86      0.84       177
           1       0.75      0.68      0.72       111

    accuracy                           0.79       288
   macro avg       0.78      0.77      0.78       288
weighted avg       0.79      0.79      0.79       288
```

```
rf_classifier_emotional_intensity.score(X_test_emotional_intensity
✓ 0.0s
```

```
0.7916666666666666
```