International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2025: 7.089

# A Multimodal Study of Depressive Indicators and Emotional Expressions

**Alaa Khazem[1], Dr. S. Jhansi Rani[2]**

[1]Department of Computer Science and Systems Engineering, Andhra University, Visakhapatnam, India
Corresponding Author Email: *ellaabodi550[at]gmail.com*

[2]Department of Computer Science and Systems Engineering, Andhra University, Visakhapatnam, Andhra Pradesh, India
Corresponding Author Email: *dr.sjrani[at]andhrauniversity.edu.in*

**Abstract:** ***The project explores different approaches to analyze human expressions from textual data, facial images, and vocal tone. By implementing machine learning and deep learning models, the system studies textual expressions and emotional cues from images and audio analysis. Rather than aiming for clinical diagnosis, it examines each model separately and focuses on the importance of artificial intelligence in supporting structured analysis of human expression. Artificial intelligence can be used not only in technical tasks, but also as a tool to analyze different types of information and understand patterns in human data.***

**Keywords:** Multimodal Analysis, Emotion Recognition, Depression Detection, Machine Learning

## 1. Introduction

People naturally express their thoughts and feelings through text, speech, and facial expressions. These forms can be represented as data and analyzed using computational models [4]. However, relying on only one type of data may not provide a complete understanding of behavior [2]. This project aims to analyze human patterns using artificial intelligence techniques across different types of data. Each modality requires a different analytical approach. The modalities are examined separately to explore patterns in human expressions.

## 2. Literature Survey

Previous studies have explored depression detection using machine learning and deep learning techniques in the context of social media data [4]. Social media users express their thoughts and emotions through various forms of online content such as comments, tweets, and posts [5]. Deep learning approaches, such as recurrent neural networks (RNN) and LSTM models, were applied to detect depression from textual data, [7]. Facial expression recognition has been studied using convolutional neural networks (CNN) for emotion analysis [2]. Other studies explored depression detection using facial images, where emotional states can be identified through visual cues [1]. Speech emotion recognition was studied using acoustic features and machine learning approaches [3]. In this project, audio is used to focus on vocal expressions and analyze them using machine learning models to analyze how emotions naturally appear in voice. Recent work has explored deep learning approaches for speech emotion recognition, using models such as CNN and LSTM to capture both acoustic features and temporal dependencies in speech signals [8]. The goal is to rely on patterns within the voice itself and analyze them using machine learning models.

## 3. Methodology

### 3.1 System Architecture

This project aims to understand human emotional states using three types of information: what people write, how they look, and how they sound. Each type of information is processed separately using methods that are suitable for its characteristics. The text modality focuses on detecting depression-related patterns, while the image and audio are used for general emotion recognition. The workflow begins by collecting and organizing the three types of input data. Each modality undergoes its own preprocessing stage. After preprocessing, relevant features are extracted: textual features using TF-IDF, visual features using convolutional neural networks (CNN) [2], and audio features using MFCC [3]. These features are then passed to machine learning models that classify the input. Multiple models are tested for each modality to evaluate their performance and determine the most effective approach. The output of each pipeline is an emotion prediction, representing the system's interpretation of the input data. The system does not combine the three modalities; instead, each one is analyzed and evaluated independently.

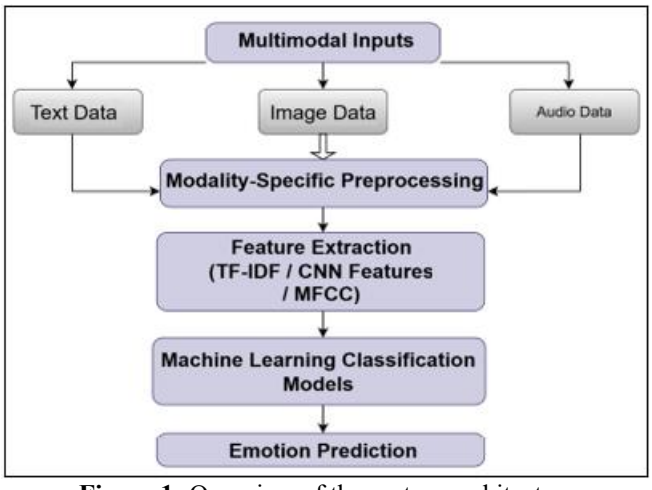


**Figure 1:** Overview of the system architecture

This figure shows the system workflow. Each type of data goes through preprocessing and feature extraction using methods such as TF-IDF, CNN, and MFCC. The extracted features are then sent to machine learning models for classification. Qt designer was used to build a graphical user interface (GUI), where users can enter text, images, or audio data and get predictions from the trained models. The final output is displayed with a non-diagnostic message.

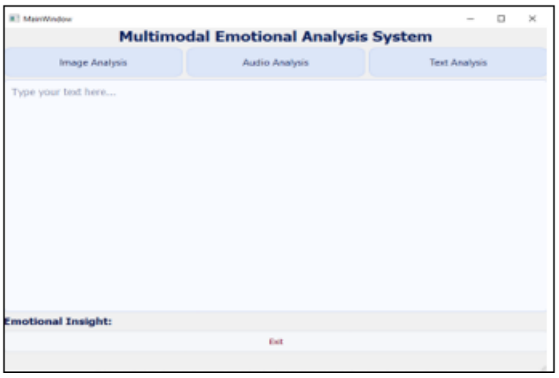


**Figure 2:** User interface of the proposed system

### 3.2 Text Processing

The text modality represents a key aspect of capturing linguistic expressions related to user behavior. This figure shows the main steps where the process starts with raw text input and includes preprocessing steps like converting text to lowercase, removing URLs and symbols, and general cleaning. The cleaned text is then converted into numerical features using TF-IDF with uni-grams and bi-grams. Next, the dataset is divided into training, validation, and testing sets. Finally, machine learning models are used for classification, producing output labels that indicate different levels of depression.

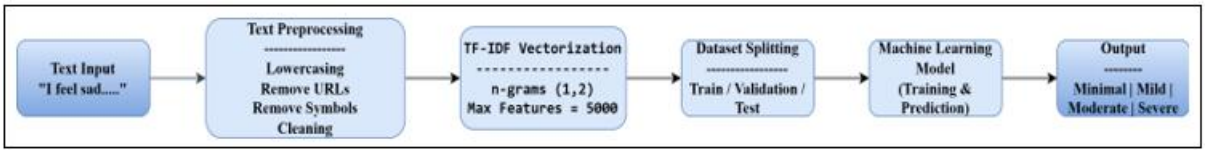


**Figure 3 :** Text processing pipeline, feature extraction, and classification steps

#### *3.2.1 Data Preparation*

The process started by loading the text data. The text data had words and phrases along with labels representing different levels of depression. At this point the text data is what people have written, with each piece having a sentence or post and a label that says what kind of depression it is.

**1) Understanding the dataset structure:**

The dataset is organized into two main columns:
sentence: contains the textual input written by users.
target: contains the corresponding label that represents the level of depression. The dataset contains 3553 samples, each consisting of a text entry and its corresponding label. The labels are grouped into four types: minimum, mild, moderate, and severe. For modeling purposes, these labels were converted into numerical values as follows:

- minimum→ 0
- mild → 1
- moderate → 2
- severe → 3

**2) Text Cleaning and Preprocessing**

First, all text is converted to lowercase to ensure consistency. URLs are removed as they do not contribute to understanding emotional content. Special characters, symbols, and numbers are removed to retain meaningful text. Extra spaces were handled. These steps help reduce noise and improve input quality.

**3) Data Splitting**

The dataset was divided into three subsets: training, validation, and testing. We used a ratio to divide the data. 80% For training 10% for validation and 10% for testing. This allows the model to be trained and evaluated on unseen data. The training set contains 2842 samples, the validation set 355 samples, and the test set 356 samples. At the end of this step, data was saved into files.

**4) Feature Extraction (TF-IDF):**

TF-IDF (Term Frequency–Inverse Document Frequency) method was used to convert text into numerical features based on the importance of words in the dataset. Words that appear frequently in a specific text but not across all texts are considered more important and words that are very common are considered to be less important. TF-IDF was used by considering both single words and pairs of words. The output is a numerical representation of the text suitable for model training.

#### *3.2.2 Machine Learning Models*

The project tested several models, such as Logistic Regression, Naive Bayes, Support Vector Machine (SVM), Random Forest, and XGBoost. The goal was to evaluate how well each model could find patterns in the data and distinguish between different classes.

1) ***Logistic Regression (Baseline Model)***
   Logistic Regression was trained using the TF-IDF features. The training was done to ensure convergence by setting the appropriate number of iterations and selecting a suitable solver for the dataset.
2) ***Naive Bayes (Baseline Model)***
   Naive Bayes is a probabilistic classifier that depends on the probability of a text belonging to a specific class based on word frequency. The model was trained using the TF-IDF vectors. It is also fast and easy to implement.

3) ***Support Vector Machine (SVM) (Baseline Model)***
Support Vector Machine (SVM) was used due to its good performance in text classification. It works by finding an optimal decision boundary that separates different classes. A linear version of the SVM was used as it is more suitable for text classification.

### *3.2.3 Handling Class Imbalance*

We used balancing techniques to reduce the effects of class imbalance. This project focused on two main methods: class weighting and oversampling. Class weighting was used in some models. Oversampling helped increase the number of samples in the minority classes. These techniques were applied before the models were retrained. The goal was to evaluate their impact on performance.

### *3.2.4 Improved Models*

***a) Logistic Regression (Balanced Model)***
Logistic Regression was retrained using class weighting to handle the class imbalance. The parameter "class_weight =balanced" was used to give greater weight to the minority classes (mild, moderate, and severe) while reducing the weight of the majority class (minimum). The model was trained using the same TF-IDF features to keep consistency. Logistic Regression showed better performance in identifying the moderate and severe classes compared to the baseline. However, the performance for the mild class remained low. The balanced model performed better than the baseline.

***b) Naive Bayes (Oversampling)***
Naive Bayes was retrained after applying an oversampling technique to handle class imbalance. The model showed improvement in identifying the minority classes that were previously underrepresented in the data set and showed improvement in identifying the moderate and severe classes compared to its baseline version. But there was a slight decrease in the model's performance on the minimum class because it became more balanced in its predictions. Performance on the mild class improved slightly, but not as much as other classes. Overall, Naive Bayes performed better than its baseline version.

***c) Support Vector Machine (SVM) (Balanced Model)***
The parameter "class_weight=balanced" was also used. The model was retrained using the same TF-IDF features. This helped the model handle the imbalance more effectively. The results became more balanced across classes. Performance improved for the severe class compared to the baseline, while the performance on the moderate class showed some improvement but was still limited. The results were better in the minimum class. The balanced SVM model showed good generalization and performed better than its baseline models.

***d) Random Forest***
The model was also trained using class weighting and remained biased toward the majority class (minimum), with many minority samples misclassified. This showed that class weighting was not sufficient to handle the class imbalance problem for this dataset. As a result, oversampling was applied. The model showed a slight improvement in performance compared to the previous attempt. But, the performance of the model was still low, especially for the minority classes.

***e) XGBoost***
It is an advanced learning technique that uses gradient-boosting algorithms to build multiple decision tree models and combine the output of these decision tree models. Oversampling was applied before training the model which reduced bias toward the majority class and improved recognition of other classes. the same TF-IDF features were used. Several parameters are defined to control the learning process, which improves the performance of the model while minimizing the risk of overfitting. These parameters include the number of trees, learning rate, and tree depth. Results show that XGBoost performed better than Random Forest and Logistic Regression and SVM. It improved performance on the mild and moderate classes, which means it can recognize these minority classes better than other models. It maintained good performance on the minimum class. While performance on the severe class is still limited. Overall, XGBoost showed a balanced performance and proved its effectiveness as an advanced model for handling imbalanced text classification problems.

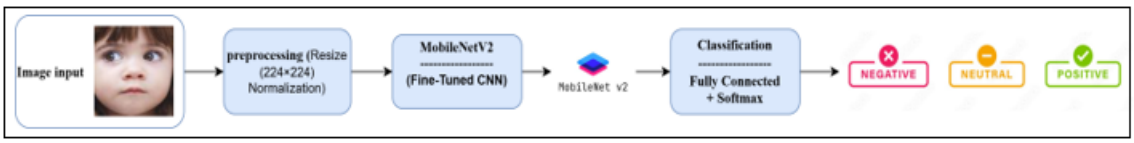


**Figure 4:** Image processing pipeline, data preparation, preprocessing, model training, and evaluation

## 3.3 Image Processing

The image modality focuses on capturing facial images that reflect human emotions. Images give direct visual cues, like facial features and expressions. The aim of this part is to find general emotional patterns based on how people look. This is achieved using deep learning models that extract visual features and identify patterns in images. The figure illustrates the main steps from preparing the data to training and testing the model. The process starts with collection and organization of the images. The data is then processed and restructured, and divided into training, validation, and testing. The deep learning models are then used to learn the visual features and classify the images according to different emotional categories.

### *3.3.1 Data Preparation (Images)*

The image dataset was created by integrating images from various sources, including the RAF dataset and some images from the FER dataset to increase diversity. Definition of the image classification was made by considering a valence-based classification approach with three different classes: positive, neutral, and negative. In this setup, 'happy' was considered positive, while 'sad', 'fear', and 'angry' were grouped as negative. The neutral class remained unchanged. The dataset was split into training, validation, and testing sets using stratified sampling. All images are resized to a fixed

input size, which is required by the deep learning models. Normalization of pixel values is performed to ensure consistency in input scaling.

### *3.3.2 MobileNet Model*

#### *a) Baseline*

MobileNetV2 was set up with pre-trained weights from ImageNet and used as a fixed feature extractor by freezing all of the convolutional layers. A custom classification head was attached. It had a Global Average Pooling layer, a Dropout layer (0.3), and a Dense layer with three output neurons using SoftMax activation to match the valence-based classification task. Training was done using the Adam optimizer (learning rate 1e-3) with categorical cross-entropy as the loss function. To fix the imbalance between classes, class weighting was used. The training process was limited to a few epochs, and a Model Checkpoint mechanism was used to save the model that performed well on validation accuracy. The model was good in recognizing the positive and negative classes, but the neutral class was still challenging. These results show that the model could find general emotional patterns, but it still had problems because some visual features were the same across classes.

#### *b) Fine-Tuning*

To improve performance, a fine-tuning strategy was applied to MobileNetV2. Where only the lower layers were kept fixed while upper layers were allowed to adapt to the dataset. Only a few of the upper layers were made trainable. During training, data augmentation methods like rotation, shifting, zooming, and horizontal flipping were used which helped the training data be more varied. Class weighting was used again. The Adam optimizer with a lower learning rate of 1e-4 also used to make the training more stable for the model. Model Checkpoint was used and an early Stopping was utilized in order to stop the training process when there was no improvement in performance. After fine-tuning, there was a clear improvement in the model compared to the baseline. The model showed a strong performance.

### *3.3.3 EfficientNet Model*:

EfficientNet-B0 model was used to continue testing the effect of the use of a more complex and deeper model on the classification performance. The idea was to check whether a more complex model would give better results under the same conditions. The same dataset (positive, neutral, negative) was used, along with the same data split and class weighting, making it comparable to MobileNetV2. Unlike MobileNet, EfficientNet uses a specific preprocessing function, so the preprocess input function was used instead of standard normalization to better match the model's original training on ImageNet. Data augmentation was also applied to improve generalization. Partial fine-tuning was used, where most of the layers were frozen, and the last set of layers were made trainable. A classification head was added, consisting of a Global Average Pooling layer, followed by a Dropout layer with a dropout rate of 0.3, and a Dense layer with three neurons and SoftMax activation. The model was trained using the Adam optimizer with a learning rate set to 1e-4, and the categorical cross-entropy function was used. Model checkpoint was used to save the best model, and early stopping was applied. Overall, increasing model complexity did not improve performance.

## 3.4 Audio Processing

The audio component focuses on analyzing speech signals to identify emotional patterns. It provides additional cues such as tone, pitch, and speaking patterns that reflect emotional states. This is done by extracting relevant features from speech and using machine learning models for classification. The figure shows the main steps. The process begins with collecting and organizing the audio data. Then data is prepared and structured. Then, the audio signals are used to get useful features that the models can use as input. The models are trained and tested to sort the audio into different emotional groups.

### *3.4.1 Data Preparation*

A new structure was created based on emotional valence instead of specific emotion labels. The data was sorted into three main groups: positive, neutral, and negative. The positive class included audio samples with a positive or stable tone, while the negative class included samples with a tense or negative tone. Neutral samples represented balanced or less expressive speech. Audio files were arranged in a directory based on each class. In order to make it easier to handle data and train models efficiently, a CSV file was created to serve as an indexing system, where each audio file path was mapped to its class label. The dataset was then split where 70% of the data was used for training, 15% for validation, and 15% for testing.

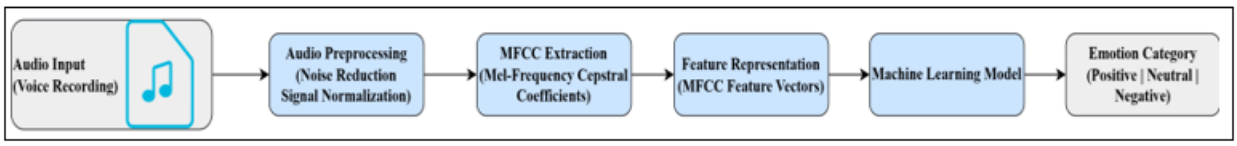


**Figure 5:** Audio processing pipeline, data preparation, feature extraction, model training, and evaluation

### *3.4.2 Logistic Regression (Audio)*

This model was selected because it is simple, quick, and gives clear results when working with structured numerical features. Mel-Frequency Cepstral Coefficients (MFCC) were used to get the important features from each audio signal before training the model. A fixed-length feature vector was created using the mean and standard deviation of the MFCC coefficients. This ensures that all audio samples have the same numerical representation. Standard scaling was applied to normalize the feature values. Class weighting was used to deal with any possible imbalance between the classes. Performance was evaluated on the test dataset. the neutral class showed good separation between positive and negative samples, with only minor of confusion between them. MFCC-based features effectively represent emotional patterns in speech, and that Logistic Regression can efficiently capture these patterns.

#### *3.4.3 Support Vector Machine (Audio)*

Support Vector Machine (SVM) model was used to improve the audio modality performance. It was selected for its ability to handle non-linear patterns. The same feature extraction process was followed. Each sample was represented as a fixed-length vector based on the mean and standard deviation of the coefficients. Feature values were normalized using standard scaling to ensure consistent input during training. The model was evaluated on the test dataset. It showed strong performance across all three classes with particular reference to the neutral. It was able to differentiate well between positive and negative classes. The SVM algorithm is more capable of capturing the complex patterns.

## 4. Results and Discussion

### 4.1 Text Classification Results

The analysis uses test data and evaluates the performances using measures of accuracy and the confusion matrix. This helps to determine the classification capability of the models in recognizing depression levels.

#### *4.1.1 Logistic Regression (Baseline)*

The Logistic Regression model performed well on the validation data with an accuracy score of 0.73 and a f1 score of 0.64.

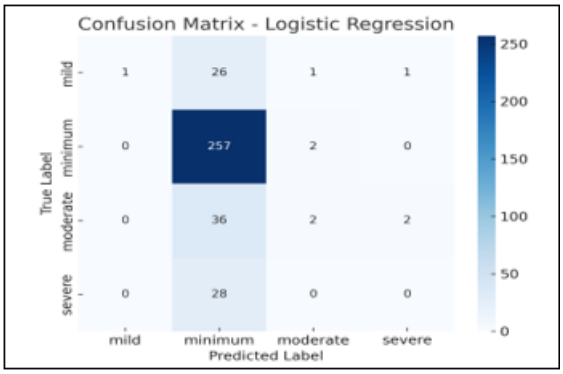


**Figure 5:** Confusion matrix for Logistic Regression model

It is shown that the model performs well on the minimum class. The problem is in the accuracy of the classifier on the rest of the classes (mild, moderate, and severe), as there are numerous instances that fall under the minority class. This reflects the impact of class imbalance on the model.

#### *4.1.2 Naive Bayes (Baseline)*

The performance score of the algorithm is 0.73, which is similar to Logistic Regression. The performance score based on macro-F1 was lower (0.21), showing the model's low performance on minority classes.

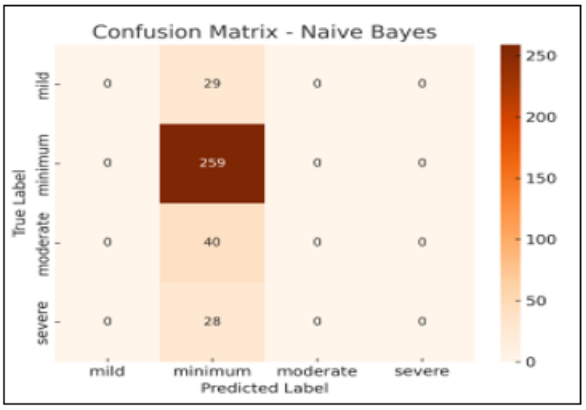


**Figure 6:** Confusion matrix for Naive Bayes model

The model is able to predict almost all data as the majority class (minimum) but it is unable to recognize the other classes (mild, moderate, and severe), which are always misclassified. This model is highly sensitive to class imbalance because it prefers the majority class.

#### *4.1.3 Support Vector Machine (SVM) (Baseline)*

SVM was evaluated as another baseline model. It attained an accuracy score of 0.75, which outperformed both Logistic Regression and Naïve Bayes Classifier models. The F1 score was 0.71 by the SVM model.

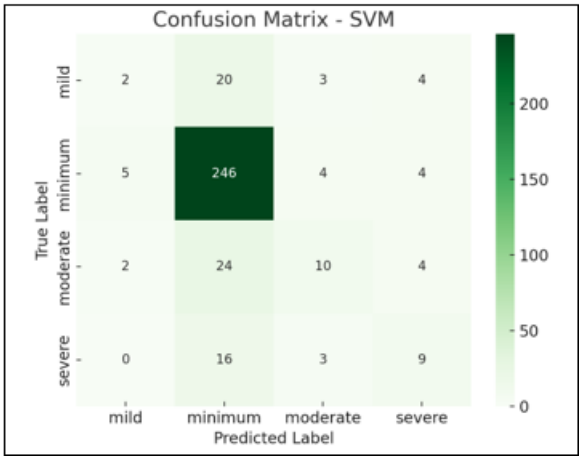


**Figure 7:** Confusion matrix for SVM model

The model performs well in the minimum class also managed to correctly classify some data points from the moderate and severe classes, However, classification within the mild class is poor, and most data points from the mild class have been misclassified to the majority class. The SVM model shows improved performance, but class imbalance still affects the results.

#### *4.1.4 Logistic Regression (Balanced Model)*

The updated model performed slightly better than its baseline, particularly in recognizing the minority classes.

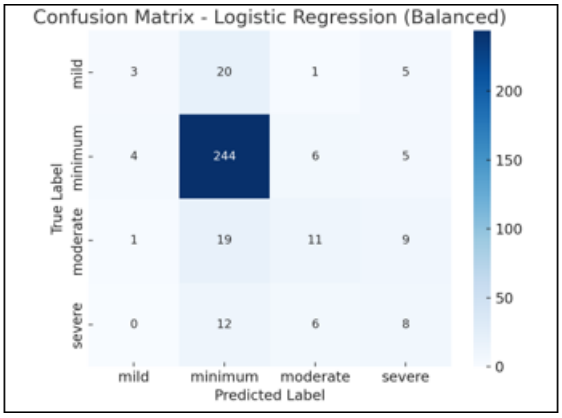


**Figure 8**: Confusion matrix for balanced Logistic Regression

The model still performs strongly on the minimum class, where most samples are correctly classified. The mild class remains the weakest point. The model does better for the moderate class. The severe class shows some improvement, but misclassification is still present, toward minimum and moderate classes. The model achieved an accuracy of 0.75, and a weighted F1-score of 0.72. Overall, using class weighting improved performance for some of the minority classes. But the model is still affected by class imbalance, especially in the mild class.

### *4.1.5 Naive Bayes (Oversampling)*

Oversampling increased the number of minority samples, which led to a noticeable change in model behavior. The Naive Bayes model achieved a better performance comparing to the baseline version.

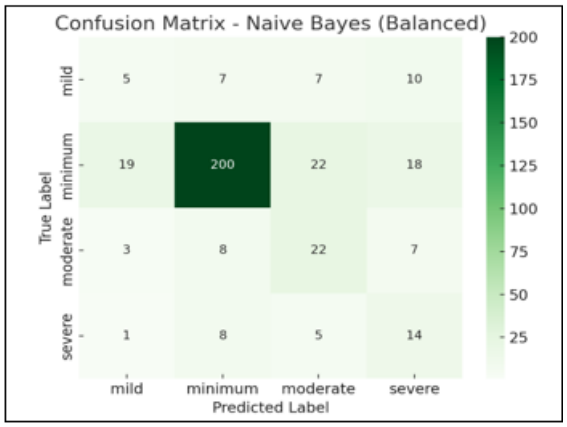


**Figure 9:** Confusion matrix for Naive Bayes model with oversampling

The model works well on the minimum class. The model shows a big improvement for the moderate class because it can correctly classify many samples. The severe class shows clear improvement, with better recognition and fewer mistakes in classification. The mild class remains difficult. Oversampling helped Naive Bayes handle minority classes better, but the model still struggles with class separation, especially for the mild category. It achieved an accuracy of 0.68, and a weighted F1-score of 0.67 after applying oversampling.

### *4.1.6 Support Vector Machine (SVM) (Balanced Model)*

The SVM model after implementing the balancing showed similar performance to the Logistic Regression model, with stable performance on all the categories.

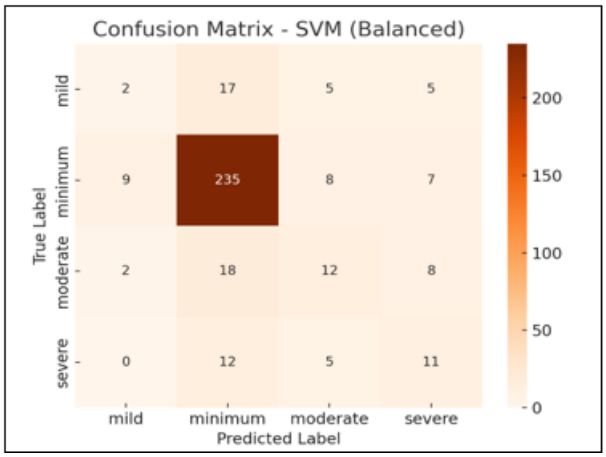


**Figure 10:** Confusion matrix for balanced SVM model

the model performs well on the minimum class, correctly classifying most of its samples. Its performance on the dominant class remains stable. For the severe class, the model achieves better results than Logistic Regression in terms of correctly classified samples. Even so, the model struggles with the moderate class, with some samples still being misclassified into other classes. It finds it hard to classify the samples in the mild class because most of the samples remain unclassified within other classes. The balanced SVM model performs comparably to the balanced Logistic Regression model, except for the mild class, where the Logistic Regression outperforms the SVM. Compared to Naive Bayes with oversampling, SVM is more stable on the majority class, while Naive Bayes performs better on the moderate class. The model reached an accuracy of 0.73, with a weighted F1-score of 0.71.

### *4.1.7 Random Forest*

At first, it was trained using class weighting to handle class imbalance. The results showed that the model was biased toward the majority class. This showed that using class weighting alone was not enough for the model. Random Oversampling was used to balance the dataset before training. Then the model was trained again. The performance of the Random Forest model was still not as good as other models.

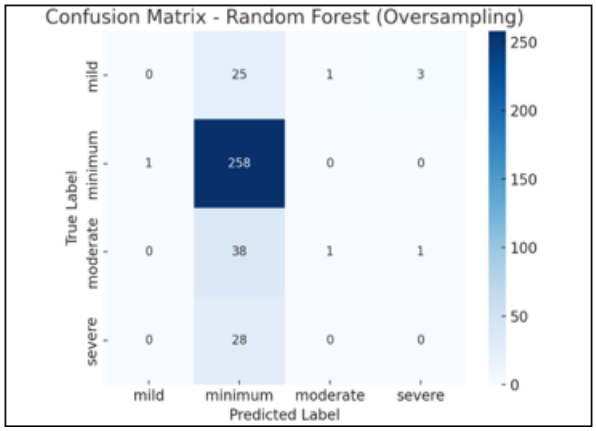


**Figure 11:** Confusion matrix for Random Forest model with oversampling

It achieved good performance on the minimum class. But it struggles with the minority classes. Many samples from the mild class are misclassified. The moderate and severe classes also show poor performance, with most samples misclassified as the minimum class. While oversampling has slightly improved the results compared to the first attempt. The Random Forest model did not perform competitively in this task. Even after applying class balancing methods. It showed lower performance compared to Logistic Regression and SVM. After oversampling it generated an accuracy of 0.73, with a weighted F1-score of 0.60.

#### *4.1.8 XGBoost*

XGBoost model training was done after oversampling the minority classes using Random Oversampling. XGBoost performed well and gave the best accuracy results amongst all the other models tested.

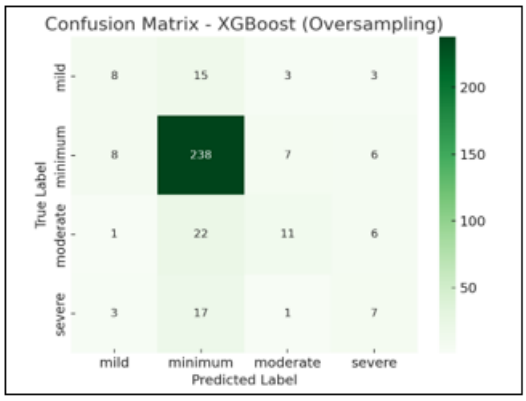


**Figure 12:** Confusion matrix for XGBoost model with oversampling

For the minimum class the performance is very high. The model shows some improvement in predicting the mild class compared to Random Forest model. The moderate class shows some improvement. Performance on the severe class has improved, with some samples no longer being misclassified. Nonetheless, the class poses one of the most difficult challenges, with many misclassified samples. The model achieved an accuracy of 0.74, with a weighted F1-score of 0.71 after applying oversampling.

#### *4.1.9 Comparison of Improved Models*

Logistic Regression and SVM produced the most reliable and consistent outcomes across various classes. Both models performed well on the minimum class, and they did better at recognizing minority classes than their baseline versions. SVM performed better in the severe class, while Logistic Regression had more balanced results overall. Naive Bayes improved after oversampling, especially in the moderate and severe classes. But its overall performance was still lower than Logistic Regression and SVM. Random Forest had weaker performance on this particular classification problem. The algorithm continued to show strong bias towards the smallest class and could not properly distinguish most samples from the minority class. XGBoost had the best results out of all the methods applied. Its performance was comparable to the Logistic Regression and SVM models. Logistic Regression and SVM provided stable overall performance. XGBoost was chosen to be used in the system.

**Table 1:** Comparison of improved machine learning models

| *Model* | *Accuracy* | *F1-score* |
|---|---|---|
| Logistic Regression (Balanced) | 0.75 | 0.72 |
| *Naive Bayes (Oversampling)* | 0.68 | 0.67 |
| SVM (Balanced) | 0.73 | 0.71 |
| Random Forest (Oversampling) | 0.73 | 0.60 |
| XGBoost | 0.74 | 0.71 |

### 4.2 Image Classification Results

Emotion classification through images was performed using deep learning models and facial expression analysis. Performance was evaluated using metrics such as accuracy, classification report, and confusion matrix. Various models were compared to recognize visual patterns of emotions.

#### *4.2.1 MobileNet (Baseline)*

The MobileNet architecture served as the baseline model for image-based emotion classification. MobileNet achieved a test accuracy score of about 0.72.

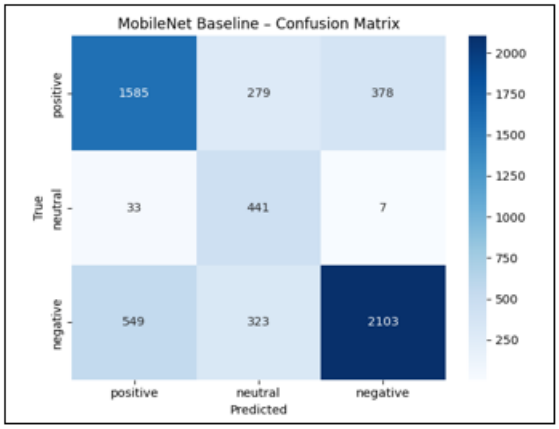


**Figure 13:** Confusion matrix for MobileNet baseline model

The positive class shows strong performance there was some confusion with the negative class because certain facial expressions may look visually similar. The neutral class achieved high recall which means that the model often correctly identifies neutral samples; however, this is at the expense of lower precision since some samples from other classes are misclassified as neutral. For the negative class, the model performs well in identifying most negative samples correctly with little confusion across other classes. The model is able to capture general emotional patterns from images, although some overlap between classes still affects performance.

#### *4.2.2 MobileNet (Fine-Tuned Model)*

A fine-tuning approach was used to make the model better at classifying emotions based on images. We used standard metrics like accuracy, classification report, and confusion matrix to test the model. The fine-tuned MobileNet had a test accuracy of 0.86, which is an improvement over the baseline model.

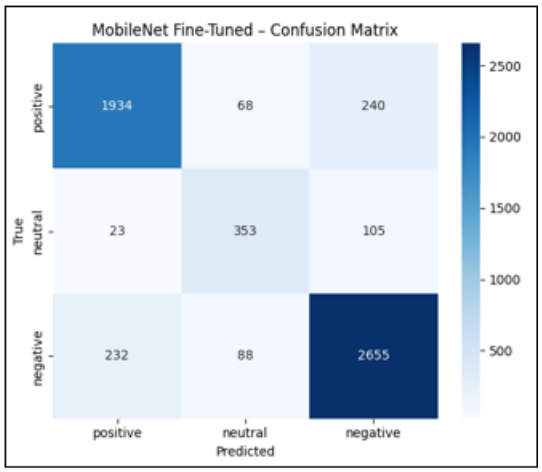


**Figure 14**: Confusion matrix for fine-tuned MobileNet model

The positive class does very well, with most samples correctly classified and only a little confusion with the negative class. The negative class achieves very strong results, and both precision and recall show that the classification performance is stable. The neutral category is the hardest one, but there is no confusion with other categories yet. The model was improved compared to the base model concerning precision and recall balance. The results showed that through fine-tuning, the model learns more discriminating characteristics of an object which reduces misclassifications and enhances the overall accuracy. That fine-tuned MobileNet will be used further for the image classification task.

#### 4.2.3 *EfficientNet (Fine-Tuned Model)*

The EfficientNet-B0 model was applied after implementing partial fine-tuning to. It was evaluated on the test dataset using the same previous metrics. The model reached a test accuracy of 0.84, which shows that it has good generalization performance.

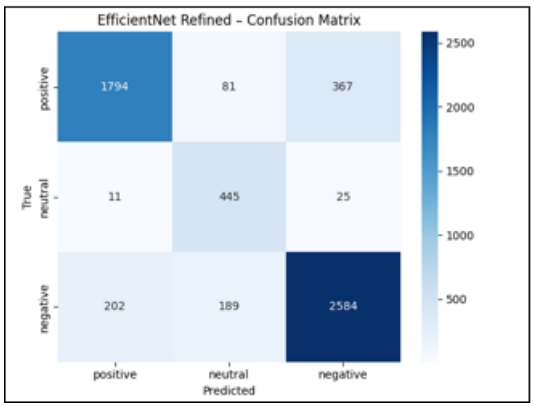


**Figure 15:** Confusion matrix for EfficientNet model

The model has high capability in differentiating among the three classes. The positive class shows good performance but there is some confusion with the negative class. The neutral class has high recall which means that most of the neutral samples are correctly classified by the model at the expense of precision since some other class samples are misclassified as neutral. For the negative class, the model works well and consistently, with both precision and recall showing the same results. Overall, the model performed well across all classes without falling into one category. But, it doesn't do better than the fine-tuned Mobile Net model, which had better accuracy and more consistent generalization.

#### *4.2.4 Comparison of Image Models*

The two models fine-tuned MobileNetV2 and EfficientNet-B0 were compared in terms of their performance on image classification. The results showed that the test accuracy of MobileNet exceeded the accuracy of EfficientNet (0.867 and 0.846, respectively). Although the performance of EfficientNet was stable and balanced across all three categories, it did not outperform MobileNet which had better generalization capabilities with a simple architecture. It means adding more layers does not always improve the accuracy of the model. MobileNet fine-tuned model was selected for image classification in this study.

**Table 2:** Comparison of image classification models

| *Model* | *Accuracy* | *F1-score* |
|---|---|---|
| MobileNet Fine-Tuned | 86.7% | 0.82 |
| EfficientNet | 84.6% | 0.82 |

### 4.5 Audio Classification Results

Emotion classification was achieved through machine learning approaches using MFCC feature. Evaluation was done using performance measures such as accuracy, classification report, and confusion matrix.

#### 4.5.1 Logistic Regression (Audio Model)

Logistic Regression model was used as the first model to classify emotions based on audio signals with features MFCCs. The model was tested using the testing dataset using with various performance evaluation metrics. It achieved an overall test accuracy of 0.94, with a macro F1-score of 0.95 and a weighted F1-score of 0.94, indicating strong and balanced performance.

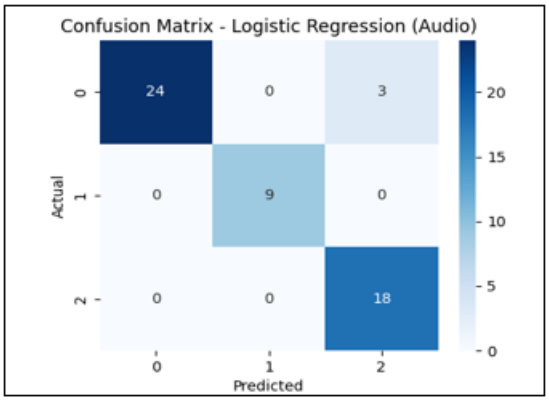


**Figure 16:** Confusion matrix for Logistic Regression model

The model performs well and has a balanced performance for the three classes involved (negative, neutral, and positive). The neutral class exhibits perfect classification performance because there is a good distinction among the acoustic parameters. The model's classification performance between the two extreme classes of positive and negative classes is good and has a small number of errors which are understandable since emotions may share some similarities.

#### 4.5.2 *Support Vector Machine (SVM) (Audio Model)*

We used the Support Vector Machine (SVM) model to classify emotions in audio using MFCC features. Standard performance metrics were used. It got a test accuracy of 0.96, a macro F1-score of 0.96, and a weighted F1-score of 0.96. This shows that it did very well and was balanced across all classes.

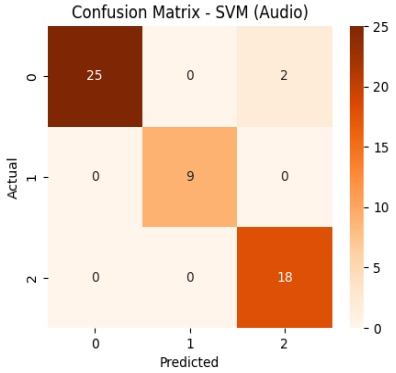


**Figure 17:** Confusion matrix for SVM model

The model has a very good capability of differentiating the three classes. The neutral class is perfectly classified. the model has also improved in separating the positive and negative classes since there is only slight confusion between them. The enhancement in both accuracy and F1-scores indicate SVM's capability to model non-linear relationships in the audio features. This model has provided robust and balanced performances for all categories with no inclination toward any particular class. The SVM was chosen as the final model for emotion classification based on audio.

#### 4.5.3 *Comparison of Audio Models*

Both models were trained and evaluated under similar conditions to ensure fair comparisons. The results showed that the SVM model outperformed the Logistic Regression model with a test accuracy of about 0.96 as compared to 0.94 for Logistic Regression. SVM achieved higher macro and weighted F1-scores which means more consistent and balanced classification. Logistic Regression gave good performance but it cannot fully capture complex patterns in the audio features. Both models did well on this task but SVM proved to be better as well as more stable.

**Table 3:** Comparison of audio classification models

| *Model* | *Accuracy* | *F1-score* |
|---|---|---|
| Logistic Regression | 0.94% | 0.94 |
| SVM | 0.96% | 0.96 |

## 5. Conclusion

This study introduced a multimodal system for the analysis of human emotional states through three different types of data modalities: text, images, and audio. Machine learning models achieved high performance in classifying text, with XGBoost getting the most balanced and accurate results. In images the MobileNetV2 fine-tuned model performed better. Both Logistic Regression and SVM gave high results at classifying audio, but SVM was more stable. The results overall show that each modality gave a different understanding of human emotions. The system shows that you don't have to use complicated deep learning models to do good emotion analysis. Instead, you can do it by carefully choosing models and representing features. In this work, the modalities were intentionally analyzed separately to better understand the contribution of each data type without introducing additional complexity.

## References

[1] S. Nepal, et al "Mood Capture: Depression Detection Using In-the-Wild Smartphone Images," Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI 2024), 2024, doi: 10.1145/3613904.3642680.
[2] C. Pramerdorfer and M. Kampel, "Facial Expression Recognition using Convolutional Neural Networks: State of the Art," arXiv:1612.02903, 2016.
[3] M. B. Er, "A Novel Approach for Classification of Speech Emotions Based on Deep and Acoustic Features," IEEE Access, vol. 8, 2020, doi: 10.1109/ACCESS.2020.3043201.
[4] W. B. Tahir, S. Khalid, S. Almutairi, M. Abohashrh, S. A. Memon, and J. Khan, "Depression Detection in Social Media: A Comprehensive Review of Machine Learning and Deep Learning Techniques," IEEE Access, vol. 13, 2025, doi: 10.1109/ACCESS.2025.3530862.
[5] M. M. Tadesse, H. Lin, B. Xu, and L. Yang, "Detection of Depression-Related Posts in Reddit Social Media Forum," IEEE Access, vol. 7, pp. 44883–44893, 2019.
[6] C. Cortes and V. Vapnik, "Support-Vector Networks," Machine Learning, vol. 20, no. 3, pp. 273–297, 1995.
[7] A. Amanat, M. Rizwan, A. R. Javed, M. Abdelhaq, R. Alsaqour, S. Pandya, and M. Uddin, "Deep Learning for Depression Detection from Textual Data," Electronics, vol. 11, no. 5, p. 676, 2022, doi: 10.3390/electronics11050676.
[8] S. Jena, S. Basak, H. Agrawal, B. Saini, S. Gite, K. Kotecha, and S. Alfarhood, "Developing a Negative Speech Emotion Recognition Model for Safety Systems Using Deep Learning," Journal of Big Data, vol. 12, no. 54, 2025.