Explanation of Approach:

The approach follows a standard pipeline for speech emotion recognition using machine learning. The main steps are:

1. Feature Extraction:

- Features extracted from the audio files include Mel-frequency cepstral coefficients (MFCCs), chroma features, and mel-frequency spectrogram. These features are commonly used in speech emotion recognition tasks and provide a representation of the spectral characteristics of the audio signal.

2. Data Loading:

- The script loads audio data from the RAVDESS dataset. It extracts relevant features for each sound file based on the specified emotions to observe.

3. Model Training:

- The Multi-Layer Perceptron (MLP) Classifier from scikit-learn is used for training. This classifier is a type of neural network suitable for classification tasks. The model is trained on the extracted features, and the training process is controlled by parameters such as learning rate, alpha (regularization term), and hidden layer sizes.

4. Prediction:

- The trained model is used to predict emotions for a test set. Additionally, the model predicts responsiveness based on the features of a single sample.

5. Visualization:

- Results are visualized using matplotlib. Bar charts show the predicted probabilities of each emotion and responsiveness. The script also includes an example of a smooth line chart for visualizing emotions and responsiveness over time.

Chosen Features:

- MFCCs: Capture the spectral characteristics of the audio signal.
- Chroma Features: Represent the energy distribution of pitch classes.
- Mel-frequency Spectrogram: Describes the energy distribution across frequency bands.

Preprocessing Steps:

- Audio files are loaded and processed using the librosa library. Features are extracted from the preprocessed audio signals.

Libraries and Tools:

- Librosa: Used for audio file processing and feature extraction.
- scikit-learn: Provides tools for machine learning, including the MLP Classifier.
- Matplotlib: Used for data visualization.
- Pandas: Used for creating and manipulating data tables.
- Joblib: Used for model persistence (saving and loading).

Challenges Faced:

- Data Quality: The performance of the model heavily depends on the quality of the dataset. Noisy or inconsistent labels in the dataset could impact the model's ability to generalize.
- Hyperparameter Tuning: Fine-tuning hyperparameters of the neural network can be challenging. The choice of parameters such as learning rate, hidden layer sizes, and regularization terms can significantly impact model performance.
- Limited Data: In some cases, the amount of available data may be limited, leading to challenges in training a robust model.

User Manual:

1. Inputting Audio Files:

- Place your audio files in a folder structure similar to the RAVDESS dataset, where each file is

named with the format "Actor_{actor_number}_{file_number}.wav".

2. Accepted Audio File Formats:

• The script currently works with WAV audio files. If your audio files are in a different format (e.g., MP3), you may need to convert them to WAV before using the script.

3. Running the Script:

• Make sure you have the required libraries (librosa, scikit-learn, matplotlib, pandas, joblib) installed in your Python environment.

• Replace the dataset_path variable with the path to your dataset.

• Run the script, and it will print information about the dataset, train the model, make predictions on the test set, and visualize the results.

4. Printing Results:

• The script will print tables showing actual and predicted emotions, a table for predicted responsiveness, and visualizations of emotion and responsiveness predictions over time.

5. User-Specific Predictions:

• To obtain predictions for your own audio file, replace the file_path1 variable with the path to your audio file.

• The script will then print the predicted emotion and responsiveness for the provided audio file.

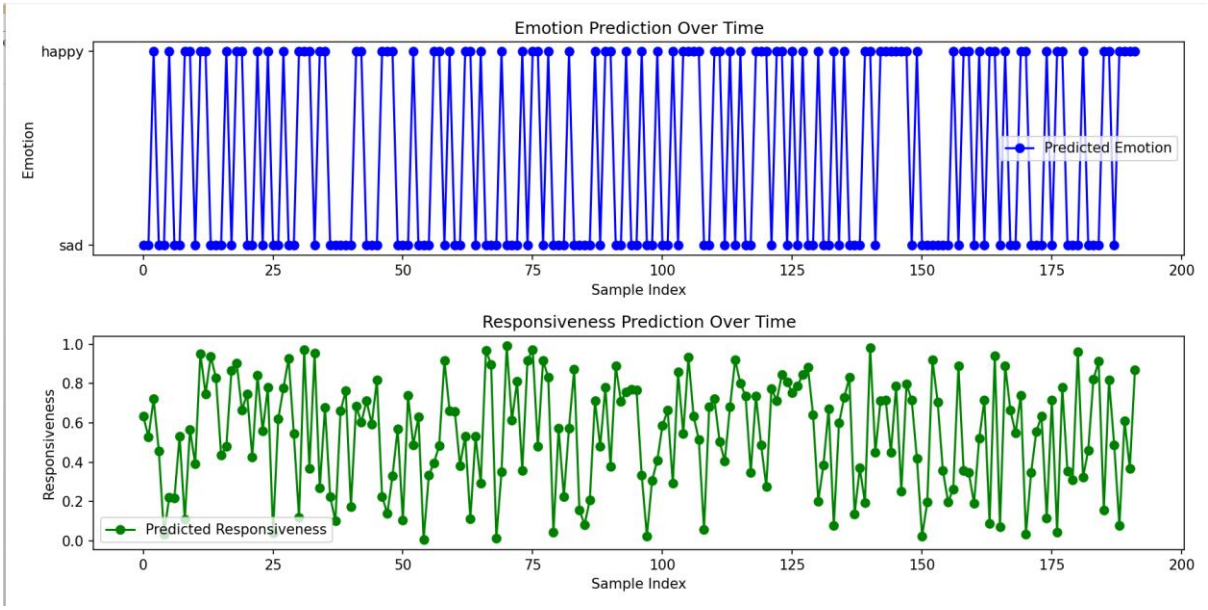
6. Interpreting Results:

• Emotion predictions are presented in tables, and visualizations show the predicted probabilities over time.

• Responsiveness is predicted for a single sample, and the result is shown in both table and chart format.

7. Accuracy:

• The script calculates and prints the accuracy of the model on the test set.



```
Error: 'DataFrame' object has no attribute 'append'
PS C:\Users\JEEVAN\OneDrive\Desktop\Assignment> & C:/Users/JEEVAN/AppData/Local/Programs/Python/Python311/python.exe c:/Users/J
EEVAN/OneDrive/Desktop/Assignment/app.py
(576, 192)
Features extracted: 180

Actual and Predicted Emotions Table:
    Audio File Actual Emotion Predicted Emotion
0        happy          happy             happy
1         calm           calm              calm
2        happy          happy             happy
3        happy          happy             happy
4      disgust        disgust           fearful
..         ...            ...               ...
187    fearful        fearful           fearful
188      happy          happy             happy
189      happy          happy             happy
190    disgust        disgust           disgust
191       calm           calm              calm

[192 rows x 3 columns]

Accuracy: 67.71%

Predicted Emotion for happy.wav: fearful

Predicted Responsiveness Table for the Single Sample:
   Predicted Responsiveness
0             3.159392e-28
1             1.719395e-03
2             8.865562e-01
3             1.117245e-01
```