MACHINE LEARNING HACKATHON

TEAM NUMBER: 4

SECTION: F

NAME: N S TUSHAR

SRN: PES2UG22CS327

NAME: MRUNAL ANANDACHE

SRN: PES2UG22CS323

NAME: M V PARTH

SRN: PES2UG22CS325

NAME: SHAMBHAVI RAIKAR

SRN: PESS2UG22CS919

1) Data Preprocessing Steps :

i)Text Preprocessing Function:

All non-alphabetic characters (like punctuation or numbers) are removed from each text entry, and the text is converted to lowercase. This standardizes the text data, reducing variations caused by capitalization or special characters.The text is split into individual tokens (words) using the word_tokenize function from NLTK, making each word available for processing.Common stopwords (e.g., "the," "is," "and") are removed using NLTK's stopword list, reducing noise in the dataset by removing words that carry little semantic meaning for most NLP tasks. The preprocessing function is applied to the Utterance column in both train_df and df, creating a new column, cleaned_utterance, with the cleaned text data

ii) Audio Preprocessing Function:

Using the pydub library we have extracted the audio from the provided Video Datasets. Wrap the conversion process in a try-except block to manage any conversion errors without halting the script.

iii) Handling Video Data

- Generating Video Paths: The get_video_clip_path function generates paths for each video file based on Dialogue_ID and Utterance_ID.
- Video Feature Extraction: The extract_video_features function captures frames from each video, resizing them to 64x64 pixels. It retrieves a specified number of frames (num_frames=5), evenly spaced, to reduce computational complexity.
- Frame Padding: If a video has fewer frames than required, the last frame is repeated to ensure consistency.

2) FEATURE EXTRACTION STEPS :

Feature extraction steps

i) Text feature extraction:

The function preprocess_text() is designed to clean and prepare text data for further analysis or modeling. Below are the main features or steps that are being extracted and applied during the text preprocessing:

1)Removal of special charecters: It ensures that the text is uniform, with only letters and spaces remaining, which is helpful for text analysis as it removes unwanted characters that don’t contribute to the meaning (like punctuation marks).

Tokenization: Tokenization breaks down the input text into smaller units (words), allowing further analysis to be done at the word level.

Feature extraction steps

ii) Text feature extraction:

The function **preprocess_text()** is designed to **clean** and **prepare** text data for further analysis or modeling. Below are the main **features** or **steps** that are being extracted and applied during the text preprocessing:

1. **Removal of special charecters**: It ensures that the text is uniform, with only letters and spaces remaining, which is helpful for text analysis as it removes unwanted characters that don’t contribute to the meaning (like punctuation marks).

2. **Tokenization**: Tokenization breaks down the input text into smaller units (words), allowing further analysis to be done at the word level.

3. **Removal stopwords**: Stopwords do not add meaningful information for most text analysis tasks, such as classification or sentiment analysis, and can thus be safely removed to reduce noise in the data.

4. **Return cleaned text**: After processing, the tokens (words) are rejoined into a single string with spaces separating them. The final output is a cleaned, preprocessed text that can now be used for further processing, such as feature extraction for machine learning models.

ii)Audio feature extraction:

we have use python module Librosa to load audio files and convert them into numerical arrays (in the form of waveform)

librosa provides various features such as-

1. **MFCC Features**:

These columns will represent the **mean values of the first 13 Mel-frequency cepstral coefficients (MFCCs)**, which are used to capture the timbral texture of the audio. MFCCs are often used in speech and music analysis to model the spectral properties of sound.

2. **Chroma feature**

These columns represent the **mean chroma features**, which are related to the pitch content of the audio. Chroma features are useful for tasks involving harmony, tonality, and music analysis.

3. Spectral contrast feature

These columns represent the mean spectral contrast, which measures the difference in amplitude between peaks and valleys in a sound spectrum. It's often used to capture timbral texture and can be useful in differentiating musical genres.

4. Zero crossing rate

This column represents the **mean zero crossing rate**, which is the rate at which the signal changes from positive to negative or vice versa. This feature is useful for distinguishing between percussive and non-percussive sounds.

5. Tempo

This column represents the **tempo (beats per minute, BPM)** of the audio, which is a measure of the speed of the audio (typically used in music analysis).

Combining Text and Video Features

- **Stacking Features**: Both text and video features are combined horizontally using np.hstack(), resulting in a single feature set for each dataset.

Encoding Labels

Label Encoding: The Sentiment labels in train_df are encoded to integers using LabelEncoder.

Feature Extraction with TF-IDF

The TfidfVectorizer is applied to extract meaningful features from the text data.

iii) Early vs Late Fusion:

Text and video features are combined into a single feature vector before they are fed into the model for training. both the text and audio features are combined into a single feature matrix. This combined feature set is then used as input to the machine learning model during the training phase. The combination can be done by concatenating the text and audio feature vectors into a single, larger feature vector for each sample. This allows the model to learn from both modalities simultaneously.

During testing, only the text modality is used to make predictions. The trained text model processes the text input, and its output is used to make the final prediction. This is done without relying on the audio features, highlighting the contribution of text data alone to the model's performance. The text model is evaluated on the testing set, and its performance is assessed using metrics like accuracy, F1-score, and precision, similar to how it would be assessed in any traditional text-based classification task.

Model decisions

Audio Features:

- **MFCC (Mel Frequency Cepstral Coefficients):** Widely used in speech and audio signal processing for capturing timbral features, which are crucial for recognizing speech and emotions in audio.

- **Chroma:** Represents harmonic and melodic characteristics and is often used in music and speech analysis.
- **Spectral Contrast:** Captures the difference in amplitude between peaks and valleys in a sound spectrum, useful for distinguishing between different types of sounds.
- **Zero Crossing Rate (ZCR):** Measures the rate at which the signal changes sign, often used in speech/music separation.
- **Tempo:** Key to capturing rhythm, especially for distinguishing between faster and slower speech, which can correlate with emotional states.

Video Features:

- **Optical Flow:** Measures motion between consecutive video frames. It's used to capture the movement patterns of objects, people, or speakers in video, and can help identify emotional expressions based on physical gestures.
- **Color Histogram:** Provides color distribution information, which can be useful in emotional recognition, as emotions are sometimes associated with specific color patterns (e.g., red for anger, blue for sadness)

PCA (Principal Component Analysis): Applied to audio features after standardization (via StandardScaler) to reduce the feature space while retaining most of the variance in the data. This is important for mitigating the risk of overfitting and speeding up training. Reducing the number of features can also help in enhancing the model’s generalization performance.

Multi-modal Approach: Using both audio and video data allows the model to capture different aspects of emotions (speech intonation, facial expressions, and physical movements).

Dimensionality Reduction (PCA): Helps mitigate overfitting and reduces computational complexity.

Random Forest Classifier: Well-suited for handling both high-dimensional data and imbalanced classes while being easy to interpret..

Performance Analysis:

A RandomForestClassifier is trained on the combined features of both text and video.

Validation: The model's performance is evaluated on the validation set using accuracy and classification metrics.

Accuracy Scores for Audio-text :0.5

Accuracy Score for Video-text: 0.48