Pre-processing

1. **Drop Duplicate**:
    - **Purpose**: Remove any repeated data entries (duplicate tweets) from the dataset.
    - **Reason**: Duplicate entries can lead to overfitting during training, where the model learns specific patterns that only exist in the repeated data. Removing duplicates helps the model generalize better to new, unseen data.
2. **Label Encoding**:
    - **Purpose**: Convert the categorical labels (emotion categories such as "happy," "angry," etc.) into numerical format.
    - **Reason**: Machine learning models require numerical input. By using label encoding, we transform the emotion labels into integers so that the model can process them effectively.
3. **Train-Test Split**:
    - **Purpose**: Divide the dataset into two parts: one for training the model and one for validating it.
    - **Reason**: Splitting the data ensures that the model has separate data to learn from and to test on. This helps in evaluating the model's performance and ensuring that it does not overfit to the training data. The "stratify" option ensures that both sets have a similar distribution of emotions, which prevents class imbalances.
4. **Tokenization**:
    - **Purpose**: Convert the textual data (tweets) into sequences of numbers.
    - **Reason**: Since machine learning models cannot process raw text, tokenization transforms the text into a numerical format by assigning a unique index to each word in the dataset. This allows

the model to understand the textual content.

5. **Padding**:
    - **Purpose**: Ensure that all input sequences (tweets) are of the same length by adding padding where necessary.
    - **Reason**: Neural networks, particularly LSTM, require input sequences to have consistent lengths. Padding ensures that shorter tweets are extended to match the length of the longest tweet, allowing the model to process them uniformly.

Model

1. **LSTM (Long Short-Term Memory)**:
    - **Purpose**: Use an LSTM model to analyze sequential data, such as the words in a tweet.
    - **Reason**: LSTMs are a type of recurrent neural network (RNN) that are well-suited for sequential data, as they can capture long-term dependencies between words in a sequence. This is particularly useful in sentiment analysis, where the meaning of a sentence often depends on the relationships between words and their order.
2. **Dropout**:
    - **Purpose**: Regularize the model by randomly dropping out neurons during training.
    - **Reason**: Dropout helps prevent overfitting by making the model less reliant on specific neurons. It forces the model to generalize better by learning from different subsets of the data during each training iteration.
3. **Sparse Categorical Crossentropy Loss**:
    - **Purpose**: Use a loss function that is suitable for multi-class classification tasks, where the output is one of several possible labels.

- **Reason**: Since this is a multi-class classification problem (predicting one of several emotions), sparse categorical crossentropy is an appropriate loss function. It works well with integer-encoded labels and measures how well the predicted probabilities align with the actual labels.

What I Tried

1. **BERT (Bidirectional Encoder Representations from Transformers)**:
    - **Purpose**: I attempted to use BERT, a pre-trained transformer model known for its ability to understand context in text by processing text in both directions (left-to-right and right-to-left). It is particularly powerful for tasks like sentiment analysis, as it captures contextual relationships between words better than traditional models like LSTMs.
    - **Reason for Attempt**: BERT has achieved state-of-the-art results in various natural language processing tasks. I hoped that using it would improve the accuracy of emotion classification by leveraging its pre-trained weights and fine-tuning it on my dataset.
    - **Problem Faced**: Unfortunately, my computer did not have the necessary resources (in terms of RAM, GPU power, or processing capability) to efficiently handle the BERT model. As a result, I could not proceed with training it on my dataset.
2. **Optimization Efforts (Epochs and Optimizer Tuning)**:
    - **Purpose**: To improve the model's performance, I experimented with different configurations of epochs and optimizer settings.
    - **Reason for Adjustment**: Training a model is an iterative process. I initially used default settings for the number of epochs and optimizer. However, I wanted to improve the results, so I adjusted the number of epochs (training iterations) to allow the model to learn better from the data. Additionally, I experimented with

different optimizers like Adam to find the best one for optimizing the learning process.

- **Result**: While I did not achieve the performance of BERT, adjusting the number of epochs and optimizer helped the model converge better during training and improved its overall performance, although there is still room for enhancement

Conclusion

This process of experimenting with different techniques and models is an important step in building a robust machine learning solution. Although I was unable to use BERT due to hardware limitations, tuning the training parameters allowed me to achieve better results with the LSTM model.