Preprocessing

1. Remove user handles, stop words

Removing usernames and stopwords from the text data during preprocessing in emotion detection or any natural language processing (NLP) task serves several purposes:

Data Noise Reduction: Usernames or handles in text data, especially from social media or online forums, can be noisy and may not carry much relevance to the emotions expressed in the text. Removing usernames can help reduce noise in the data and make it cleaner for analysis, allowing the model to focus on the actual text content that conveys emotions.

Data Privacy: Usernames or handles may contain personal information, and removing them from the text data can help protect the privacy of individuals whose data is being analyzed. This is important to comply with data protection regulations and maintain ethical considerations in handling sensitive information.

Generalization: Usernames are typically unique identifiers associated with specific users, and their presence in the text data may not contribute to the generalization of the emotion detection model. By removing usernames, the model can learn patterns and emotions from the content of the text itself, rather than relying on user-specific information that may not be applicable in other contexts.

Stopword Removal: Stopwords are common words such as "a," "an," "the," "and," "in," etc., that do not carry much meaning and are often used for grammatical purposes. Removing stopwords can help reduce the noise in the text data and improve the efficiency of feature extraction techniques such as bag-of-words or TF-IDF, as these techniques often rely on word frequency or occurrence patterns. Removing stopwords can also help the model focus on more meaningful words that carry more emotional content.

Dataflow

Data Collection: we gathered a dataset of text samples that are labeled with their corresponding emotions. This dataset will be used to train and evaluate your emotion detection model.

Data Preprocessing: Clean and preprocess the text data by removing any irrelevant information, such as special characters or punctuation marks, and converting the text to a standardized format. You may also need to handle common language processing tasks like tokenization, stemming, and stopword removal.

Feature Extraction: Convert the preprocessed text data into numerical features that can be used as input for a machine learning algorithm. Common techniques for feature extraction include word embedding, bag-of-words, and term frequency-inverse document frequency (TF-IDF).

Model Selection: Choosing an appropriate machine learning or deep learning model in important for your emotion detection task. Naive Bayes, Support Vector Machines, Linear Regression.

And Bert based model

Model Training: Split your dataset into training and validation sets. Use the training set to train your selected model by feeding it the preprocessed text data and their corresponding emotion labels. Adjust hyperparameters and experiment with different model architectures to optimize the performance of your model.

Model Evaluation: Evaluate your trained model using the validation set to measure its accuracy, and F1 score. Adjust your model or experiment with different techniques if needed to improve its performance.

Model Deployment: Once you are satisfied with your model's performance, deploy it in a production environment to start making predictions on new, unseen text data.

Model Monitoring and Maintenance: Regularly monitor your deployed model's performance and update it as needed to ensure accurate and reliable emotion detection results.

Vectorrizers used

vectorizers are used to convert textual data into numerical representations, which can then be used as input for machine learning algorithms. Vectorizers transform raw text data into a format that can be easily processed by machine learning models, which typically require numerical input.

We used 2 gram

1. Count vectorizers
2. CountVectorizer is used for converting text data into a matrix of token counts. document is represented by a vector where each entry represents the count of a particular word or term in that document. The resulting matrix of token counts can then be used as input to machine learning algorithms for various tasks, such as text classification, sentiment analysis, and topic modeling.
3. Tfidf vectorizers
4. TF-IDF is a numerical statistic that reflects the importance of a term in a document within a collection of documents. It is a commonly used technique for text feature extraction that takes into account both the frequency of a term in a document (term frequency) and the rarity of the term across the entire document collection (inverse document frequency).

Why to use BERt ?

The vectorizer used in Ml-based approaches, such as count vectorizer and TF-IDF, do not account for word sense disambiguation. They generate the same vectors for words with different meanings and same spelling, resulting in embeddings that do not capture the sense of the word. On the other hand, BERT can generate contextualized embeddings that capture the sense of the word. This is because BERT takes into account the context of the word in the sentence, providing embeddings that are sensitive to the meaning of the word in its given contex.

What is called Bidirectional?/

BERT is called "bidirectional" because it leverages information from both the left and right context of each word in the input text during pre-training. This is in contrast to traditional language models, such as the unidirectional models, like the original Transformer model, which only consider the left or right context of words during training.

What are the disadvantages of using BERT?

CountVectorizer:

TF-IDF (Term Frequency-Inverse Document Frequency):