Tweet Analysis and Prediction

Data Exploration

To explore the training data, we displayed the training dataset. We can see from the data below that we have text and their target in the data frame. Here is the list of attributes in the training data.

1. id
2. Keyword
3. location
4. text
5. Target

To explore the test data, we displayed the training dataset. We can see from the data below that we all feature as the training data frame except for the target feature. Here is the list of attributes in the test data.

1. id
2. Keyword
3. location
4. text

So, our task is to predict these labels for the tweets in test data.

Train/Test Data Distribution

Let's take a moment to examine the sizes of our training and test datasets. The training data consists of a total of 7613 tweets, while the test data comprises 3263 tweets. This information provides an overview of the quantity of data available for both training our model and evaluating its performance on unseen test samples.

Class Distribution

Let's take a closer look at how the different classes are spread out in our training data. From the bar chart, it's clear that out of around 7600 tweets, roughly 4300 belong to class 0, and about 3300 are in class 1. While the classes aren't perfectly balanced, with more tweets in class 0, it's still good enough for classifying without needing to tweak the distribution. In simpler terms, we can effectively work with the data as it is without making any adjustments.

[6]:

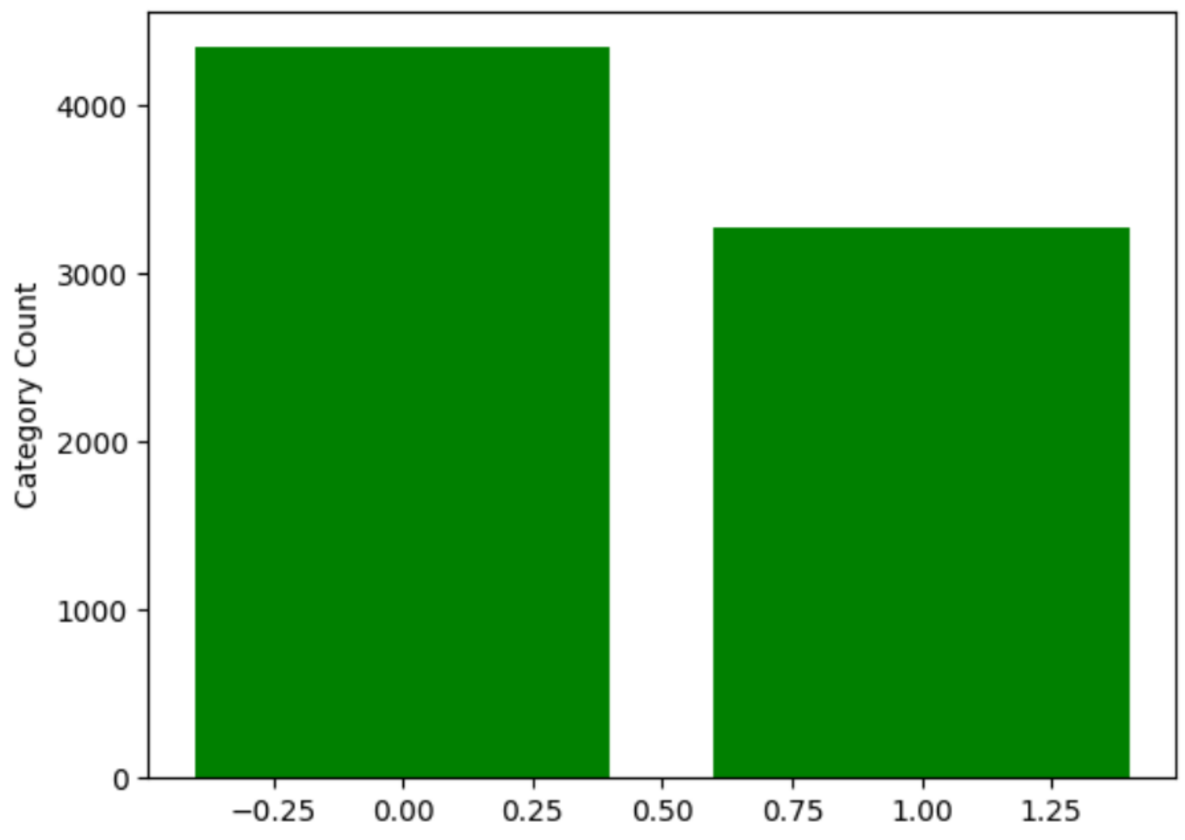

Figure 1 Class Label Distribution in Training Data

Missing value detection and removal

Let's analyze the presence of missing values in both the training and test datasets. In the training data, we observe that there are no missing values in the 'id,' 'text,' and 'target' columns. However, the 'keyword' column has 61 missing values, and the 'location' column has 2533 missing values.

Moving on to the test data, we find a similar pattern of no missing values in the 'id,' 'text,' and 'keyword' columns. However, the 'location' column in the test data has 1105 missing values.

To fix the missing information in both the training and test datasets, we've decided to replace the empty spots with nothing. This way, we keep the data complete and tidy for further analysis and model training.

In the training data, we're filling in the missing values in the 'keyword' and 'location' columns with empty spaces. Similarly, in the test data, we're putting empty spaces in place of missing values in the 'location' column.

This straightforward approach helps maintain the datasets in good shape for future steps, making sure the missing values don't cause any issues as we move forward with our analysis and machine learning model training.

Tweet Length Analysis

In both the test and training data, we checked the lengths of tweets. In the training data:

1. The shortest tweet has 7 characters.
2. The longest tweet has 157 characters.
3. On average, tweets are about 101 characters long.

In test data

1. The shortest tweet has 5 characters.
2. The longest tweet has 151 characters.
3. On average, tweets are about 102 characters long.

So, tweets in both the training and test datasets are about the same length on average, around 101 characters. So, they're similar in terms of tweet length.

Data Cleaning

It is important to clean data before applying an embedding technique to it. The following steps were performed to clean the data:

1. Tokenization is like breaking down a sentence into smaller pieces, like words or parts of words. We extracted the tokens from each of the tweets.

2. All characters in the text were converted to lowercase. This ensures that the same words with different cases are not treated differently.

3. Stop words were removed to make sure that they do not contribute to the model.

Word Embedding

GloVe embeddings are word representations that understand the meaning of words based on how they appear together in sentences. We're using GloVe embedding on the tweets for the following reasons:

1. GloVe helps models understand words and their meanings, making them better at tasks like understanding sentiments in tweets.

2. These models are pre-trained on very large datasets of words. So they are reliable and ready to use.

3. GloVe embeddings are used for lots of positive or negative classifications of text-based data.

4. Adding GloVe to models makes them smarter. They can guess the relationships between words.

In summary, GloVe embeddings improve language understanding in models, making them more effective for various text-based tasks.

Model

We created a model for understanding and predicting whether a piece of text is related to a particular topic or not. The model has three main parts:

Embedding Layer: This part helps the model understand the meaning of words in our text. It takes the words in our data and turns them into numbers, making it easier for the computer to work with them.

LSTM Layer: This part of the model learns patterns and relationships in the data. It's like the brain of the model that remembers important things from the past while processing new information. We used the size of the LSTM layer to 8.

Dense Layer: This is the output layer that makes the final prediction. It tells us if the text is related to the topic or not.

We used the Adam optimizer, a special algorithm that helps the model learn better from the data, and we set a learning rate to control how fast the model learns. We used learning rate 0.0001.