**Project Overview**

Sentiment analysis, a sub-field of Natural Language Processing, is one of the most popular topics and research fields in data science. We will be working on social media sentiment analysis. We aim to be able to classify tweets, reviews and comments from social media as positive, negative or neutral.

Most of the open-source datasets that we found on the internet are properly labeled and structured. Data collected by ourselves need to be properly labeled. Then, we will go through the cleaning, preprocessing and separation of test and training data steps.

We searched for some tools for our project and found some popular and powerful open-source NLP frameworks in Python. We will probably use Natural Language Toolkit (NLTK) . It comes with all the pieces you need to get started on sentiment analysis.

**Data**

This dataset contains over than 400,000 tweets extracted using the twitter api. The tweets have been classified from 0 (negative) to 4 (positive). The dataset contains 6 fields which are target as integer, ids as integer, date as date, flag as string, user as string and text as string.These 6 fields are shown below.

Dataset is data is csv file contain :

● target: The polarity of the tweet (0 - negative, 4 - positive)

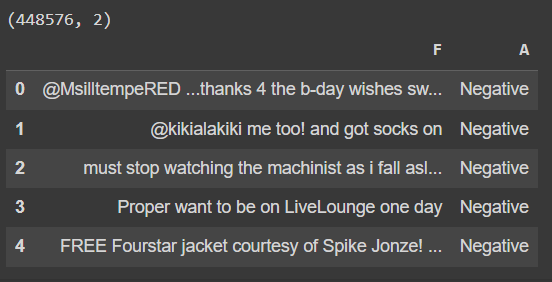
● ids: The id of the tweet.

● date: The date of the tweet.

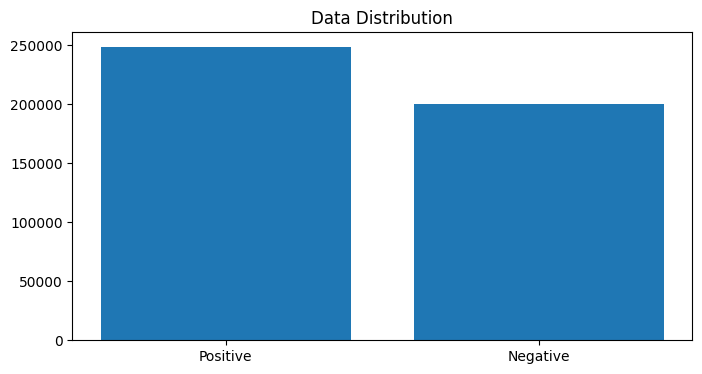
● flag: The query. If there is no query, then this value is NO\_QUERY.

● user: The user that tweeted.

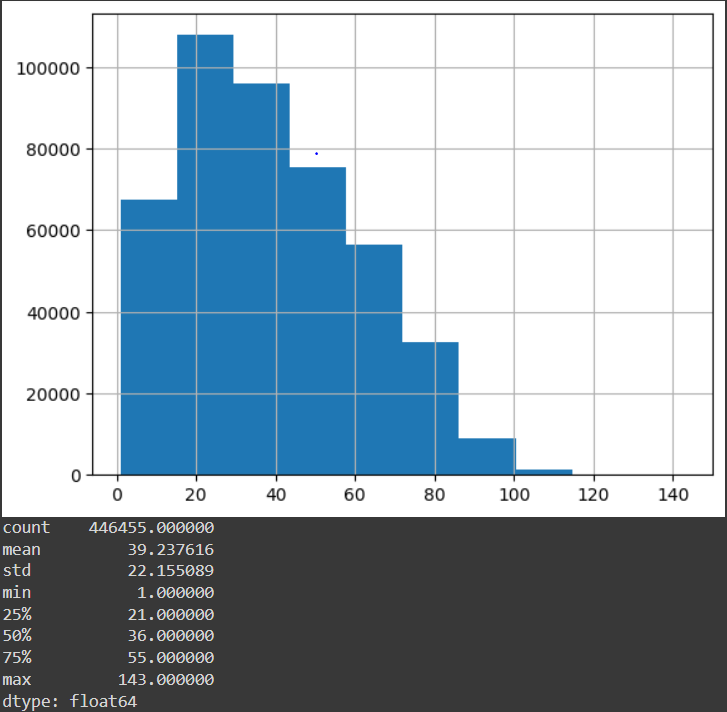
● text: The text of the tweet



Positive and negative samples are equal. The dataset distribution has not any skewness

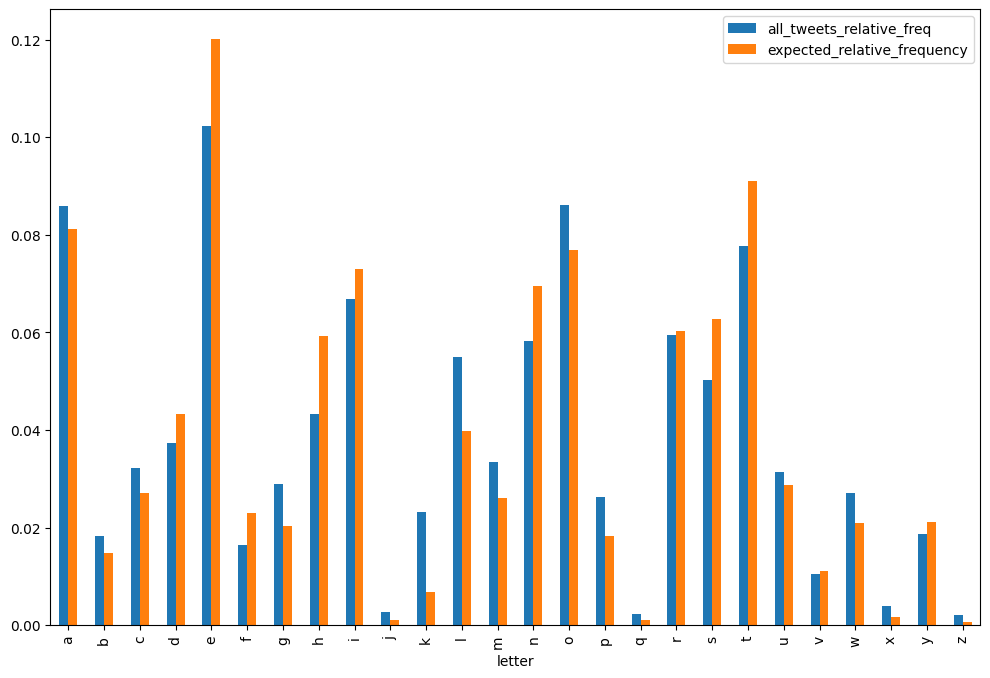


we remove tweets for zero length and make some processing by using stemmer to convert each word into original shape in English and remove all punctuation by using regex expression

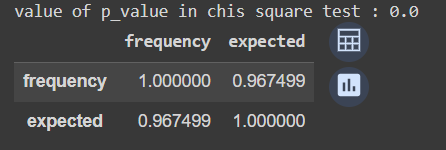


**Number of Letters**

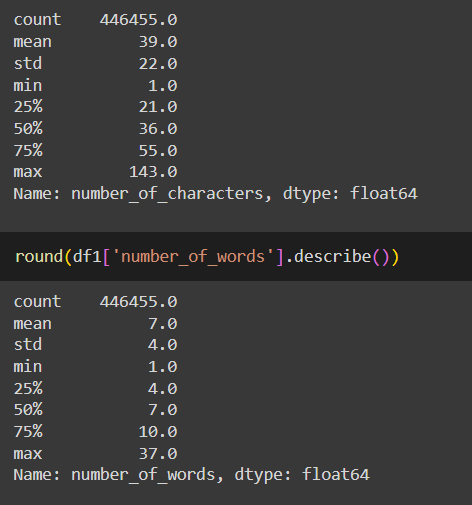
We provide the frequency and the relative frequency of the letters of the whole tweets. Finally, we will apply a chi-square test to test if the distribution of the letters in tweets is the same with what we see in English texts.

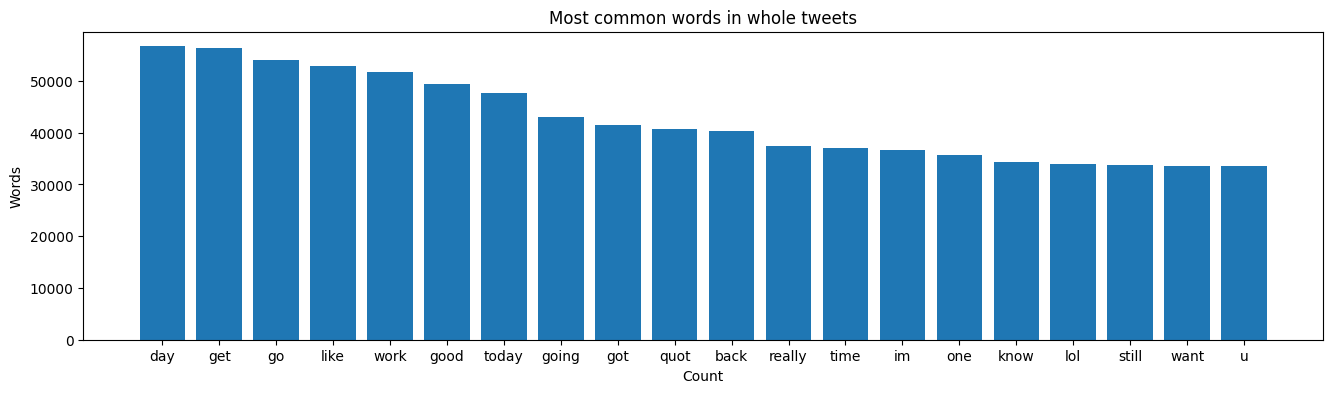


We got the p-value (p) as 0 which implies that the letter frequency does not follow the same distribution with what we see in English tests, although the Pearson correlation is too high (~96.7%) as shown in

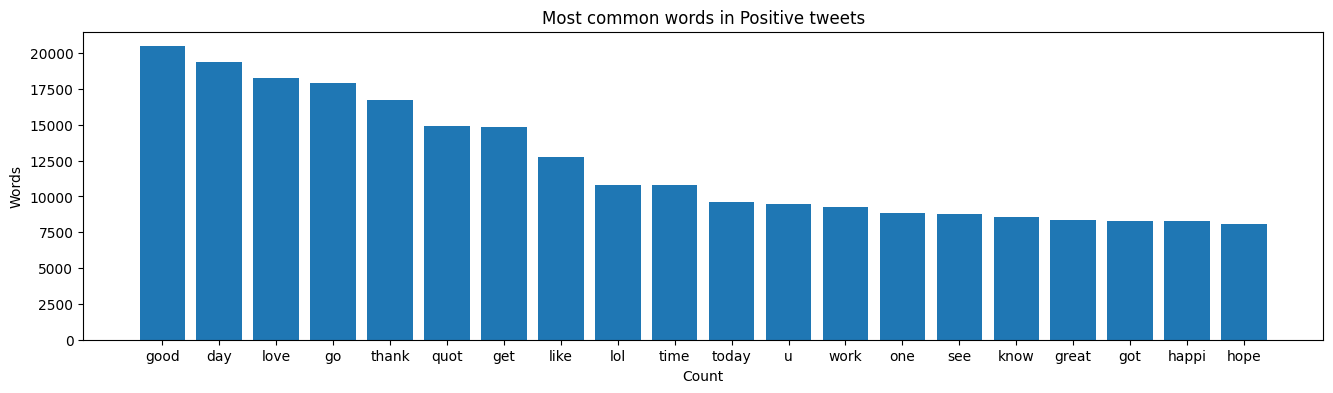


We counted the number of characters and words for each tweet and analyzed the data frame according to maximum number of characters and words , minimum number of characters and words , mean of the number of characters and words and theirs standard deviation.



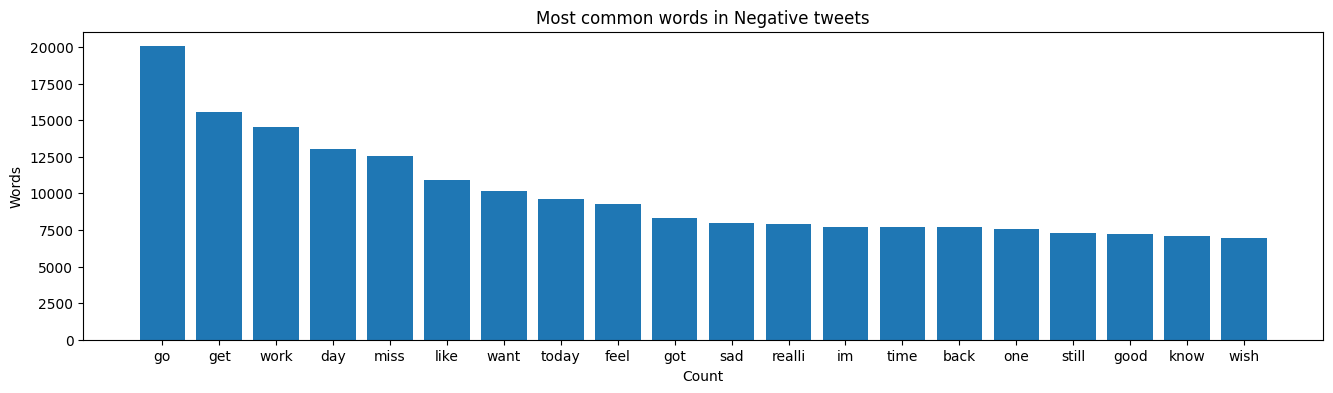


**Positive Tweets**





**Negative Tweets**





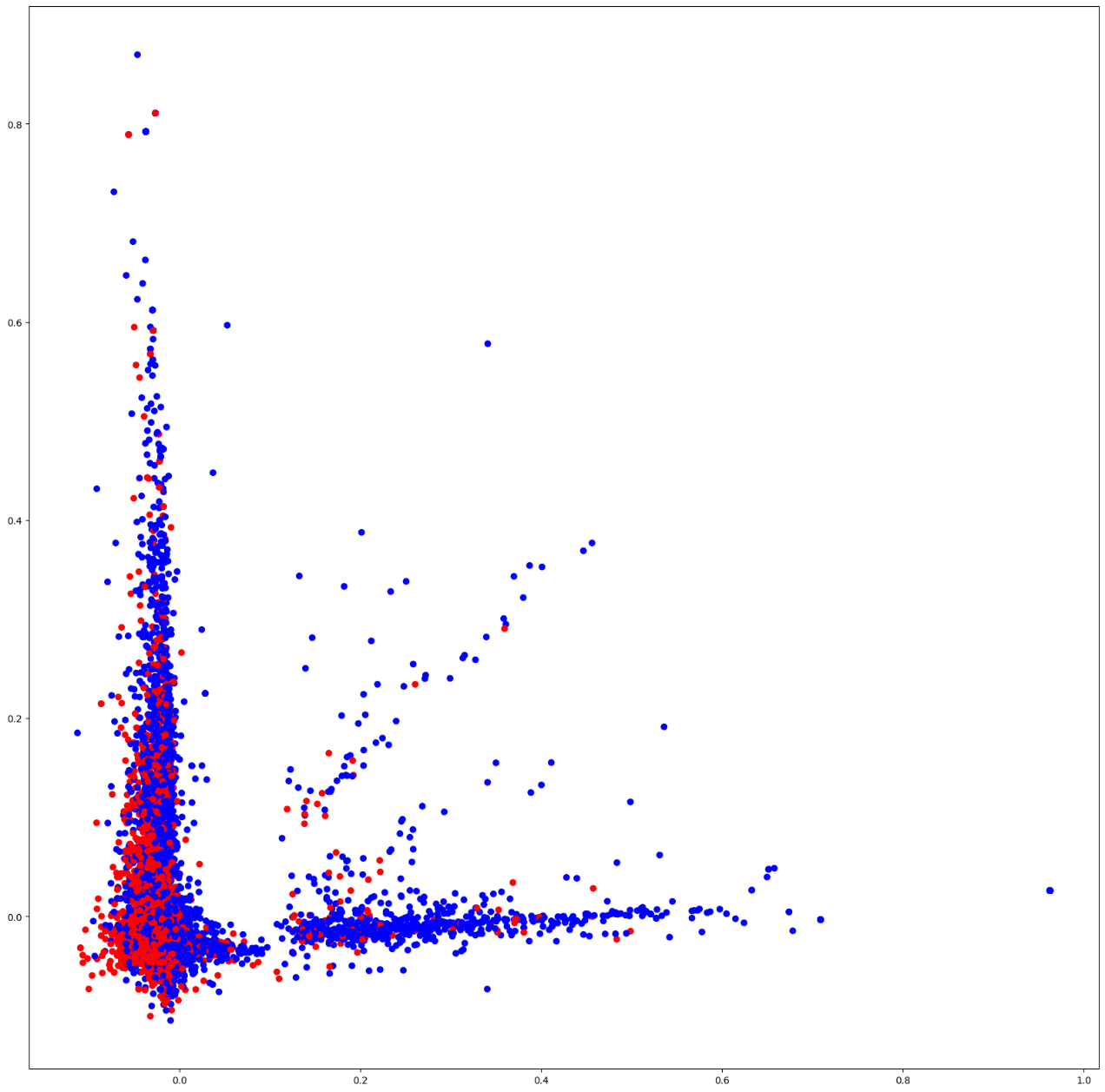
We are using tokenization to know unique words we use in dataset and the total number of unique words are 210343

**Embedding Glove**

We can train the embedding ourselves. However, that approach can take a long time to train. So, we use transfer learning technique, and we use GloVe Global Vectors for Word Representation. The Global Vectors for Word Representation, or GloVe, algorithm is an extension to the word2vec method for efficiently learning word vectors, developed by Pennington.

It is an unsupervised learning algorithm for obtaining vector epresentations for words. Training is performed on aggregated global word-word co occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. We download the GloVe. Then, we initialize an embedding index that has 400000 word vectors, and embedding matrix.

**Scatter Plot**



We used feature extraction methods, bag-of-words, and word embedding. Bag of words with TF-IDF is a common and simple way of feature extraction. Bag-of-Words is a representation model of text data and TF-IDF is a calculation method to score the importance of words in a document. After applying bag-of-words with TF-IDF, we create the scatter plot according to these results

**Model**

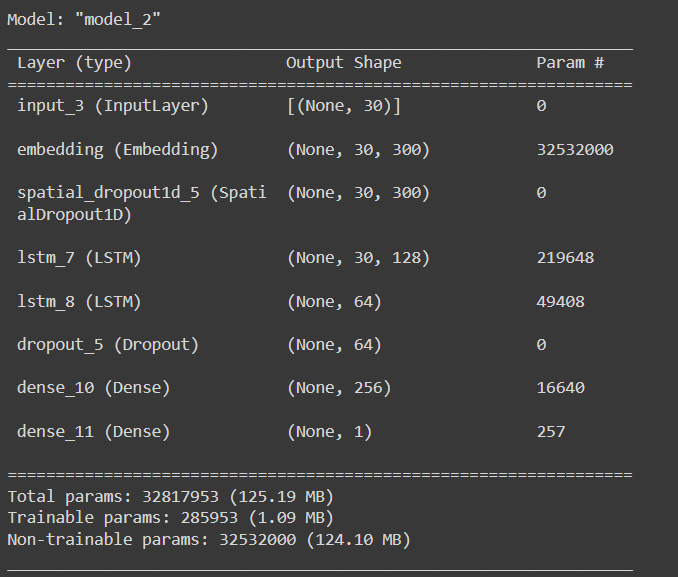
I have used embedding index and embedding vector to convert each word into matrix of values to ease using words with dealing with algorithm and use LSTM

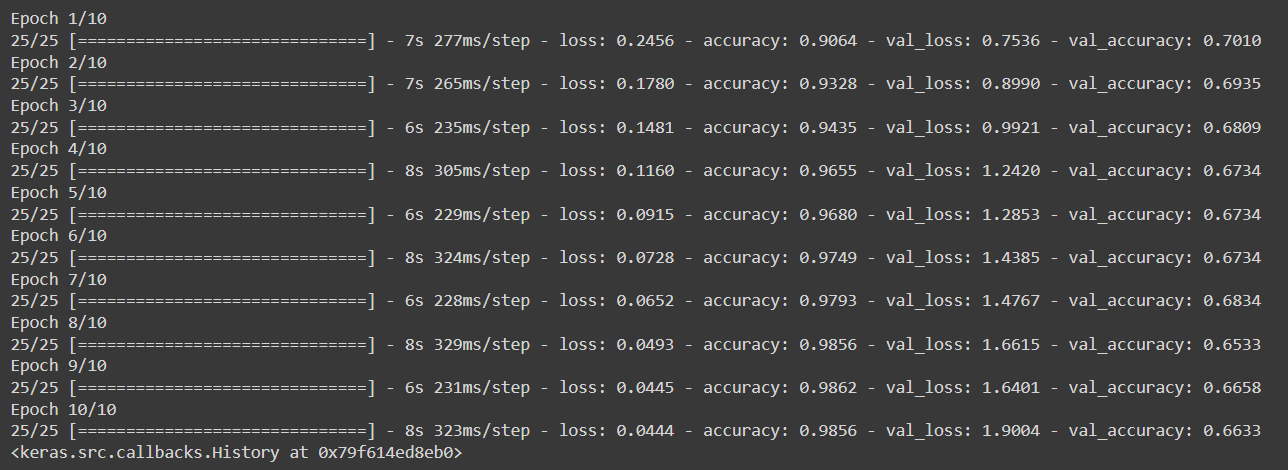
Conv1D Layer: Increase the number of filters to 128, which can potentially capture more complex patterns in the data. Also, decrease the kernel size to 3 to capture more localized features.

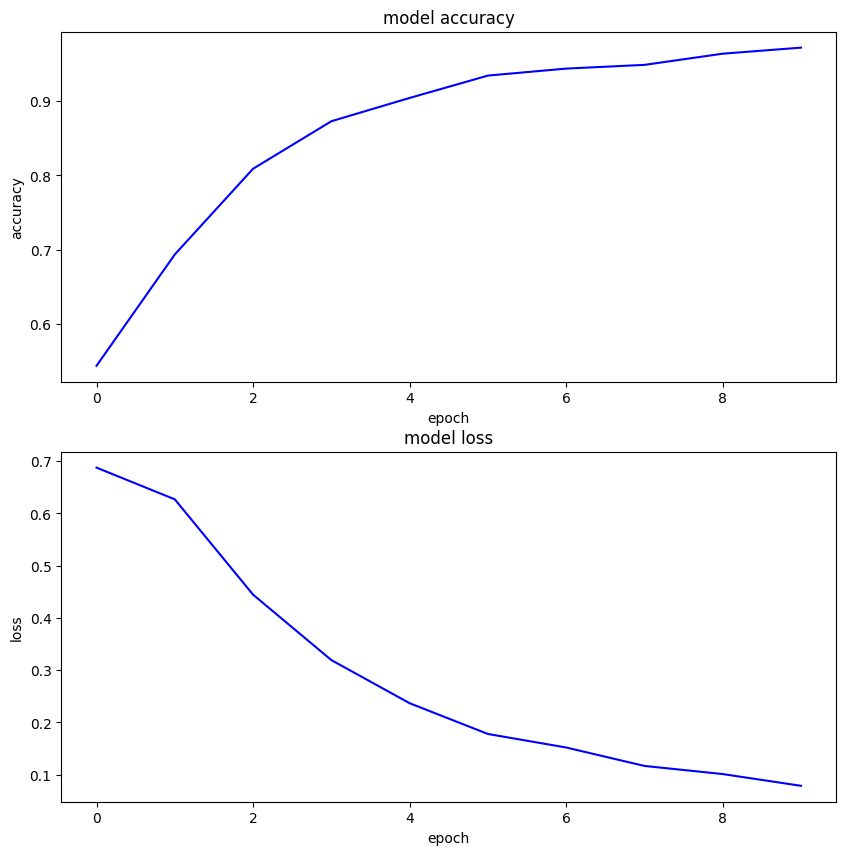
Bidirectional LSTM Layer: Increase the number of units to 128, which can allow the model to learn more complex temporal relationships.

Dense Layers: Decrease the number of units in the dense layers. In this example, the number of units has been changed to 256 and 128, respectively.

These adjustments aim to increase the model's capacity to capture and learn more intricate patterns and relationships in the data







**Sample of input tweets**

when I enter two tweets and I want to know what are the output of thiers

I enter (Starbucks coffee was not as good as i thought , we boycott Starbucks)

And the output are ['Positive', 'Negative']