Tweet Sentiment Extraction

Brull Borràs, Pere Miquel

April 4, 2020

1 Domain Background

Data is all around us, and with the introduction of Social Networks, the most straight forward to share this data is via text. This means that we need to find ways to interpret unstructured and free information such as **language**. When being introduced into **Natural Language Processing**, Sentiment Analysis is the go-to example one would try to tackle in the form of *tweets*. The statement behind it revolves around inferring if a rather short message could be classified as positive, negative or neutral, based on the subjective information of the language: is the author using "*good* or "*bad*" words? Can we try to predict the emotional state of the author based on such small piece of information?

What we are going to do, however, is look at this same problem from a different perspective. Given the sentiment of the author, can we actually know which words make a message being positive or negative?

2 Problem Statement

Our main objective is explore different NLP techniques that could aid us into finding which parts of the message transmit the *sentiment* information and classify those into positive, negative or neutral. This study is based on the Kaggle's Tweet Sentiment Extraction competition.

3 Datasets and Inputs

Data can be obtained in Kaggle. We are going to use the *train.csv* available, as we won't be covering the submission of the results to the platform.

The train data is composed by 27.486 rows with the following information:

- textID - Unique identifier of the row
- text - Complete tweet
- selected text - The text that supports the tweet's sentiment. It is a subset of the *text* feature containing one extract from a sentence. I.e., it does not contain independent words from different parts of the tweet.
- sentiment - The generated sentiment of the text.

4 Solution Statement

We will explore different NLP techniques based on extracting information, explain and apply them.

For the data preprocessing, we will apply the usual tools as stemming, removing stopwords and splitting the free text into a clean and normalized list of words. After exploring the selected text as our target, we will discuss if we could remove punctuation marks or they are sometimes selected.

Models might be trained separatedly: one for each of the possible sentiments.

5 Benchmark Model

Our starting point will be Named Entity Recognition as the simplest approach into extracting information from free text. With the same processed data, we will apply different models and compare their results against NER to see if we can obtain better results.

6 Evaluation Metrics

The Metric used will be the Jaccard index:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

In our case, the sets are the two sentences which we are comparing after lemmatization, so that words such as *friend* and *friendly* are treated as equal.

7 Project Design

We are going to follow the usual approach for Machine Learning tasks:

- Preprocess data as discussed above: Remove stopwords, apply stemmers and convert free text into a list of clean and homogeneous words.
- Split data into train and validation.
- Apply models and compare their results

In the end, we will train and deploy the best obtained model using AWS Sagemaker.