Disaster Tweets

Milestone 3

Taylor Anderson

**INTRODUCTION**

Twitter is a place people go to talk about things that are going on in their life. If an emergency or disaster happens, Twitter will often be the first place that they go to comment on it. It allows people to report that emergency in real time. There are many reasons that it would be useful to be able to quickly identify and filter out tweets that relate to a disaster. We could receive immediate information from primary sources. The tweets could include pictures, videos, or descriptions of what happened. After the fact, we could reach out to twitter users that posted about the disaster for follow-up information. The list goes on and on. We will be using Natural Language Processing techniques to try and identify these tweets that are related to a disaster.

**BACKGROUND**

The data comes from a Kaggle competition (<https://www.kaggle.com/c/nlp-getting-started/data>) and was originally created by the machine intelligence company, Figure Eight. The data contains a training set with 10,000 tweets that were hand classified and a test set with 10,000 more tweets that are not labeled.

The goal of the competition was to identify which tweets are about real disasters and which one’s are not. This is the same goal of our project. We have analyzed this dataset by determining the structure of the tweets, cleaning them to account for special symbols, stopwords, and any other content that we believe might misconstrue the context of the tweet, and tokenizing the phrases to obtain the meaning. Our research questions include:

* Can we tell the difference between a tweet that involves a natural disaster, and one that is a figure of speech?
* Which words or phrases are helpful in understanding the context?
* Which words or phrases are detrimental to understanding the context?
* Do special symbols contribute to the meaning of the sentence, or steer the meaning off course?
* Does the length of a tweet have any correlation to its meaning?
* Does punctuation (‘!’ vs ‘.’) make a difference?

**DATA CLEANING**

We started by inspecting the data to see what we were working with. The dataset has the following fields:

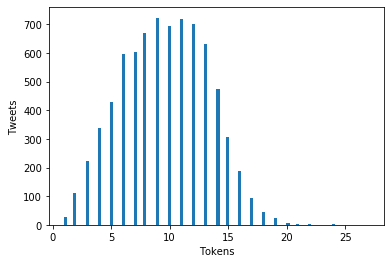
* id: a unique identifier for reach tweet
* text: the text of the tweet
* location: the location the tweet was sent from (may be blank)
* keyword: a particular keyword from the tweet (may be blank)
* target: in train.csv only, this denotes whether a tweet is about a real disaster (1) or not (0)

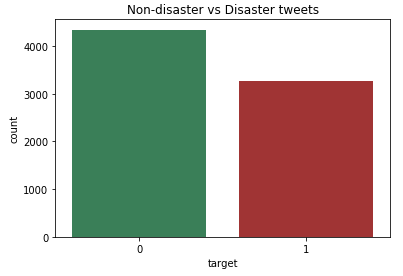


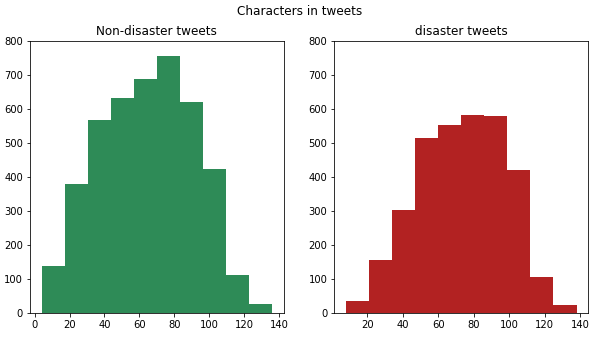
We removed the id, location, and keyword features because we wanted to focus on only the text of the tweet.

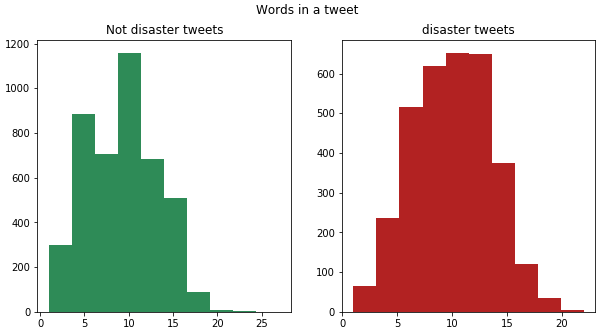
Next, we had to clean the tweets. To do this, we created a function that ran the text through five different functions which cleaned the data in different ways. The first function used BeautifulSoup to remove html language that may exist in the tweets, the second function removes stopwords and punctuation, the third removes brackets, the fourth removes URLs, and the final function removes hashtags. We apply each of these functions to the data which then returns the cleaned text which we add as a new column to the dataset.

**EDA (Visualizations)**









**MODELING**

Our hypothesis assumes that words in a tweet link directly to whether or not that tweet is about an actual disaster. This implies a linear connection between the words and the result, so we need a linear model. We have a huge number of parameters (tokens) for this model and not enough data points to accurately estimate them all. We won't want to use a standard regression algorithm because smaller datasets tend to have poor Least Squares Estimates which can result in overfitting. Instead, we want to shrink (or regularize) the coefficients so that the algorithm will produce low bias and low variance.

We recall that bias is the is the difference in the average prediction of our model and the correct value we are trying to predict, while variance is the difference in fits between data sets.

Ridge Regression is one example of a machine learning algorithm that uses regularization. The main idea behind Ridge Regression is to find a model that doesn't fit the training data too well by introducing a small amount of bias. This causes the variance to be consistently lower when testing on new data because the prediction will be less sensitive to each individual token. It does this by shrinking (regularizing) coefficients, pushing them towards '0' values so they work better on new datasets. Ridge Regression uses Cross Validation to determine the appropriate amount of bias to add based on the lowest calculated variance, which is exactly what we needed.

**PREDICTIONS**

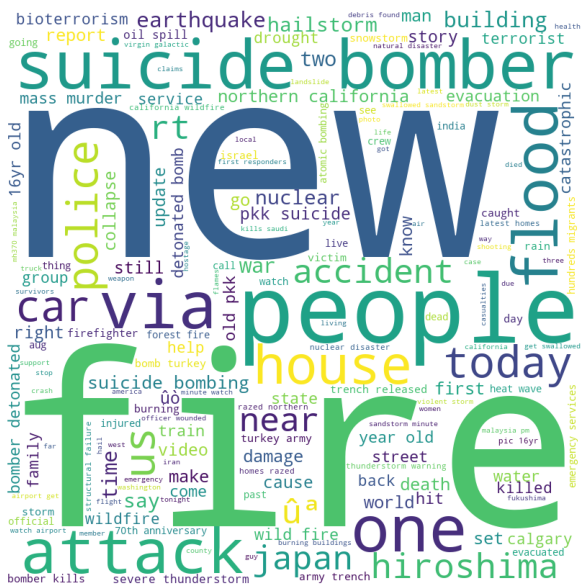
From this point, we could have just run the model and considered the output, but we wanted to ensure that the model was not experiencing moments where the accuracy would be thrown off considerably, in other words, where the model would run well in some cases and poorly in others. We decided to use Cross Validation to test the model.

We ran cross validation on only the training data because we wanted to keep the test set separated from the training set which we had used to train the model. Because the model had never been exposed to the test data, we were able to get an unbiased estimate of its performance.

The specific type of cross validation we chose was called k-fold cross validation, with five folds selected. This meant that the training data would be partitioned into five subsets and the model would be trained with each subset and validated against the test data one subset at a time. This test would provide us with the F1 score which is a weighted average of the precision and recall. This value ranges from 0 being to lowest to 1 being the highest. After running this test for all five subsets we were presented with a different F1 score for each:

0.58951175, 0.45111492, 0.5498008 , 0.48909091, 0.67164179

Taking the highest value, we can expect the accuracy to average around 67%. This accuracy could be improved in the future, but for now we are comfortable with that percentage.

We ran the model and had it introduce a new variable “target” into the data next to each tweet. If the value was ‘1’ then the model believed the tweet was about an actual disaster. If it was a ‘0’, then it was not a disaster. Using this new labeling, we tokenized each word from a tweet that the model believed to be about a real disaster, then output the words into a word cloud. We figured a word cloud would reveal the important words in the predictions while offering an engaging, or eye-catching, visual. This would give us a glimpse into what words within the tweets that the model considered indicative of a 'disaster'.

**CONCLUSIONS**

**ACKNOWLEDGEMENTS**

We are indebted to the communities behind the multiple open-source software packages and statistical reporting on which we depend.

**REFERENCES**