APRIL 19, 2020

disaster tweet analysis

NLTK Binary Classification of Disaster and Non-Disaster tweets

SUBMITTED BY

NORA P CARCAMO ACOSTA
IE-MBD-2

SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

summary

This challenge proposes a binary text classification task: we have to decide whether a tweet is talking about an actual disaster or not. This is a comprehensive programming solution which includes a proper analysis of the data, the hypothesis to use/implement, a detailed explanation of each step, and the analysis of the results. The final model section includes conclusions and reasons for final model evaluation. The notebook is available on this google colab link.

Kaggle Competiton Description

Twitter has become an important communication channel in times of emergency. The ubiquitousness of smartphones enables people to announce an emergency they're observing in real-time. Because of this, more agencies are interested in programmatically monitoring Twitter (i.e. disaster relief organizations and news agencies).But, it's not always clear whether a person's words are actually announcing a disaster. In this competition, the challenge is to build a machine learning model that predicts which Tweets are about real disasters and which one's aren't. The dataset of 10,000 tweets that were hand classified is provided in training and test sets. Acknowledgments This dataset was created by the company figure-eight and originally shared on their 'Data For Everyone' website.

Tweet source:
https://twitter.com/AnyOtherAnnaK/status/629195955506708480

Reference material from:
Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

BERT tutorial:
https://github.com/google-research/bert/blob/master/predicting_movie_reviews_with_bert_on_tf_hub.ipynb

Analysis Approach

In the following report and jupyter notebook I will explain the process for my analysis and recommended solution for this task. First, I will go through data import and preliminary analysis of the text. Next, I will use the data provided to generate new features to better understand the text structure. Then, I will apply text processing techniques such as cleaning, tokenization, and building a TFIDF matrix. Once the text is processed, I will implement multiple algorithms for text classification to determine the approach that provides better results and higher accuracy. Finally, I will present the final recommendation and reasons for the selection along with my kaggle competition submission.

data analysis

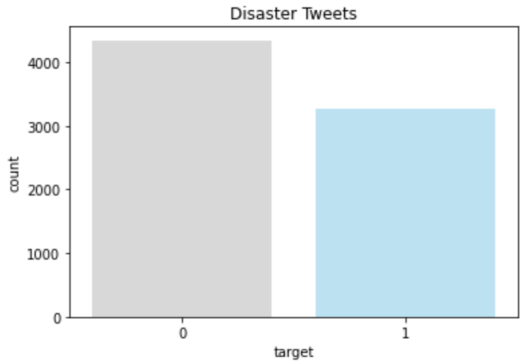
Two datasets are presented, a training set and a test set. The training set contains 7,613 observations and 5 features: id, keyword, location, text, and target label. The test set contains 3,263 observations with the same features except target. Here we will gain better understanding of the text as it pertains to the type of tweet.

Missing Values

The features ***keyword*** and ***location*** on both train and test sets contain missing values. The missing values of ***keyword*** represent approximately 0.8% of training and test sets, and ***location*** represent 33.3% and 33.8% on training and test sets respectively. Location has 3,342 unique values many of which can't be identified on a map, therefore, this feature will not be used in the analysis.

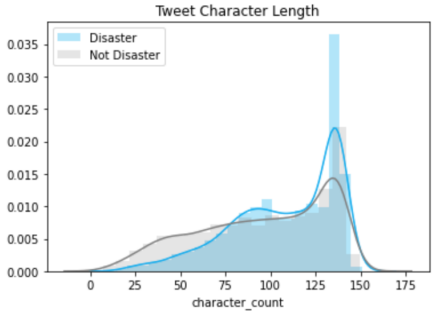
Class Analysis

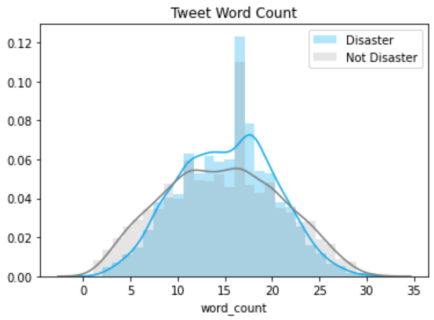
The target class **DISASTER** makes 43% of the training data (3,271 observations), therefore this training set is considered balanced. The following image shows the value count of both classes, where 0 is not a disaster tweet and 1 is a disaster tweet.

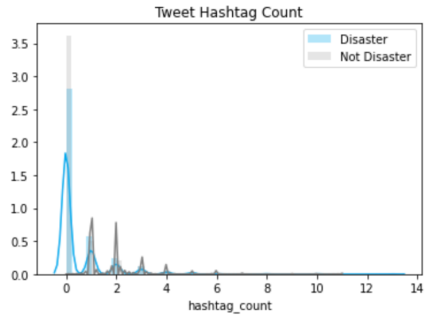


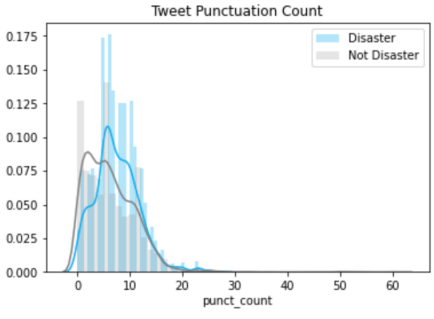
Feature Generation

I generated the following new features for analysis: tweet length, word count, hashtag count, and punctuation count. The graphs show the difference of these features between the two classes.









modeling process

Cleaning

The basic cleaning processing of ***text*** before going to modeling includes: removing html, removing punctuation including hashtags, removing non alpha, lowercasing text, and stemming with snowball stemmer. The result of this cleaning process is stored in the feature ***clean_txt***.

TF-IDF

TF-IDF is a common term weighting scheme in information retrieval used in text classification. It represents the term-frequency times the inverse document-frequency. Using TF-IDF scales down the impact of low frequency tokens. The vectorizer extracts unigrams, bigrams and 3-grams for analysis from the ***clean_text*** as input, and outputs a sparse matrix which is tf-idf-weighted which will be used to test the following algorithms for text classification.

Naive Bayes

While the Naive Bayes model assumes independence among features, it is a simple model with applications to text classification. Naive Bayes typically requires integrer feature counts, we will test the NB model with the fractional counts provided by TF-IDF. The Naive Bayes model provides an recall of **0.6** on class 1 and accuracy of **0.79** on 5-kfold cross validation.

Logistic Regression Classifier

The logistic regression classifier is a simple and good alternative commonly used for binary classification. This model gives recall of **0.54** on class 1 and accuracy score of **0.78** on 5-kfold cross validation.

Random Forest

An ensemble Random Forest classifier is typically powerful in fitting complex datasets using simple classification rules. This model gave us recall of **0.56** on class 1 accuracy of **0.77** on 5-kfold cross validation.

Linear SVC

Linear SVC implements a one-vs-the-rest multiclass strategy for classification. This model results in recall of **0.62** on class 1 an accuracy of **0.79** on 5-kfold cross validation.

SVM with RFB kernel

SVM implements a one-agains-one approach for multiclass classification. I trained an SVM model with the standard RBF kernel which resulted in recall of **0.49** on class 1 and accuracy of **0.75** on 5-kfold cross validation.

Bidirectional Encoder Representation from Transformers (BERT)

BERT, a language representation model created by Google in 2018, is the first deeply bidirectional, unsupervised language representation pre-trained using only a plain text corpus. BERT models require data input to be formatted in a specific manner before feeding into the model. The input builder function takes the training set features and produces a generator, a common design pattern for Tensorflow Estimators. This process is computationally more expensive that the previous models. Using Google Colab, training runtime was over 9 minutes. Evaluation gave recall of **0.78** on class 1 and accuracy score for this model was **0.82**.

solution

Selected Model

The model that provided the best solution for our tweet classification task was **BERT**, providing us an accuracy of **82%**. More importantly, this model had the highest recall score of **78%** with only 139 observations labeled as false negatives. This is an important consideration for the intended use case for this model, which is to give first responders a better lead in identifying true emergency situations through twitter feeds. With this in mind, incorrectly labeling emergencies (false negatives) is more 'expensive' as it results in non-action during a potentially dangerous situation. It is preferable to correctly identify as many true emergencies as possible, even at the expense of mislabeling non-disaster situations as disasters. This model also provides a high AUC metric of **0.82**.

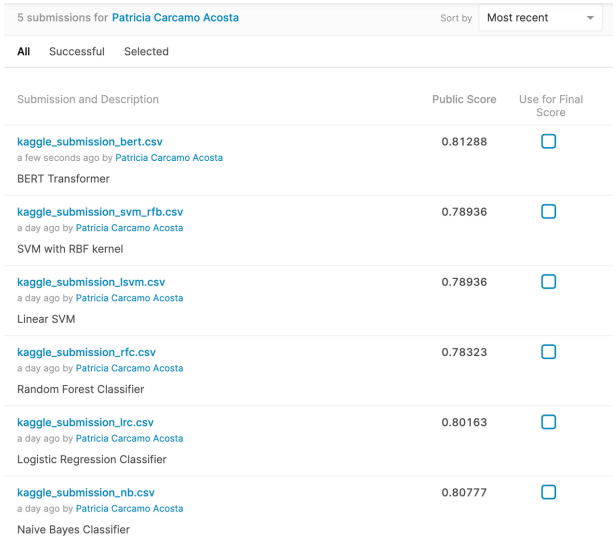
While simpler models like Naive Bayes and Linear SVC provide comparable accuracy scores to BERT, they have significantly lower recall of 0.6 which is a critical consideration in this analysis.

Transformer Models

Transformer models are in fact advancing the field of Natural Language Processing by leaps and bounds. By using transfer learning, they make high performing NLP tasks accessible to anyone. These models still require high technical know-how to implement and fine tune. I tested a BERT model to better understand the bidirectional approach as it is considered an improvement to other transformer models.

Kaggle Competition Submission

The following are my submissions to the Kaggle competition for the models tested:



The performance of this algorithms on the test set is very close to the accuracy scores obtained previously in evaluation. The highest performance comes from the BERT transformer model with 81.3% accuracy which means the model is not overfitting but providing the same performance.

Final Thoughts

Analyzing tweets and predicting their classification has given me a good understanding of the complex nature of natural language processing in this important real life application. It is exciting to see and understand the advances in this field which make understanding and getting meaning from text possible.