Bao Ngoc Dinh| Xueqing Jin| Zhichao Lyu| Haoyan Ou

BYGB 7977 SECTION 2 | FORDHAM UNIVERSITY

Evaluate and Classify Disaster Related Twitter

team 8

**Executive summary**

Disasters can occur at any time around the world. How to respond to disasters in time is always a problem. For example, people need timely rescue in natural disasters such as earthquakes. How to quickly detect the occurrence of disaster is the key issue. Therefore, to spot disasters, many institutions rely on social media like Twitter. Social media users often post about what is happening immediately, which makes social media become faster and more informative than daily news reports. That is, if you know which posts actually are about disasters and which posts are irrelevant, you will have a quicker response.

The dataset we used is a set of tweets that report disasters, hand classified as relevant or irrelevant. Based on the labels of disaster in the train dataset, we are going to build a model to identify if the specific tweet is related to disaster. We will use sentiment analysis to analyze the pattern of labeled text. A topic model will be used as an unsupervised model to classify the original text. Furthermore, we will apply three models to vectorized text data and build a classifier to detect whether a tweet is related to disaster.

**Business Goal**

Twitter is one of the most active social media sites today. People post events on twitter everyday and disaster is included in them. However, not all tweets posted about disasters are useful or reliable. Among thousands of tweets, it is necessary to distinguish the relevance tweets and classify the exact disaster type.

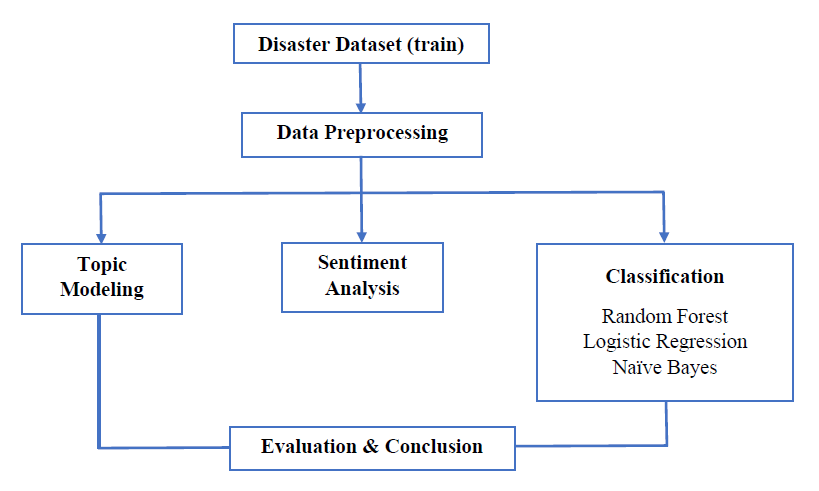
In the study, we want to build models which can correctly detect whether the tweet is related to disaster or is a false alarm and classify the disaster type.

**Data Description**

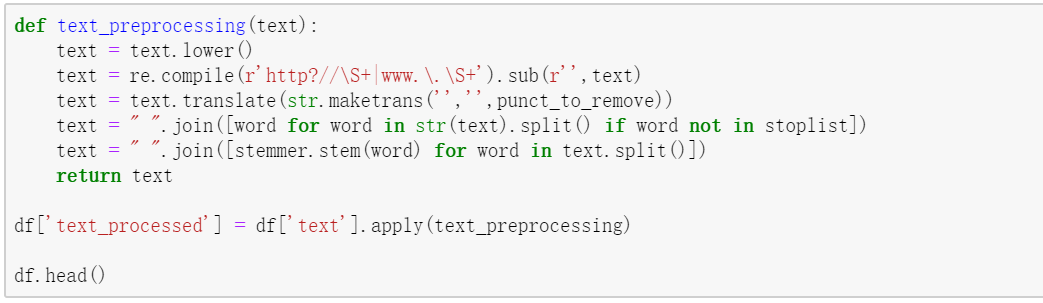
The data in our research was a part of kaggle’s competition. The dataset was split into training and testing. However, since the competition is on-going, only the train dataset has the real ‘target’ value (relevant/not relevant). Thus, we only use the train dataset, which has more than 7000 tweets. The dataset contains 5 columns, which are id, keyword, location, text and target. In our research, we only focused on using keyword, target and text columns.

|  |  |
| --- | --- |
| **Column Name** | **Description** |
| target | Whether the tweet is related to disaster. 0 = Not Relevant, 1 = Relevant |
| keyword | The specific type of disaster |
| text | The tweet content |

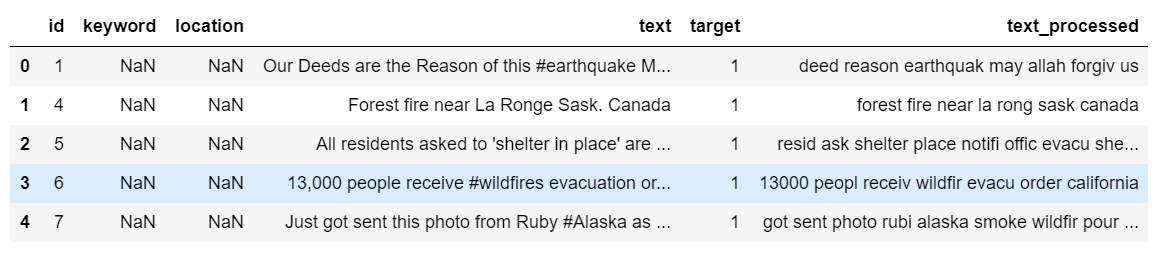
**Methodology**



**Data Preprocessing**

We write python code to preprocess our data.

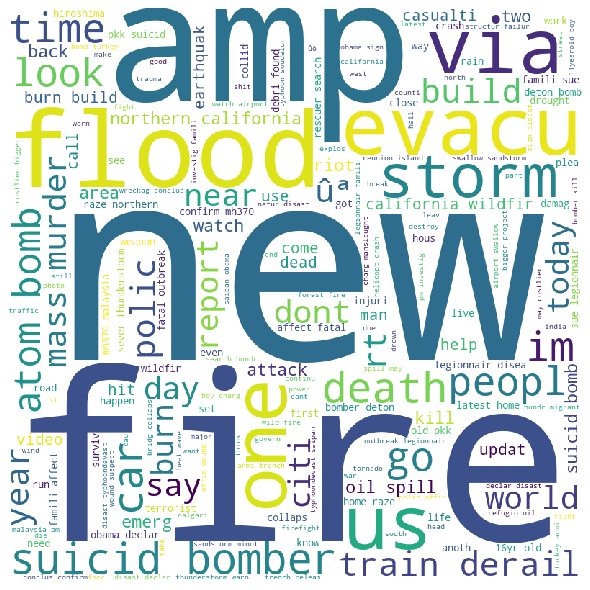
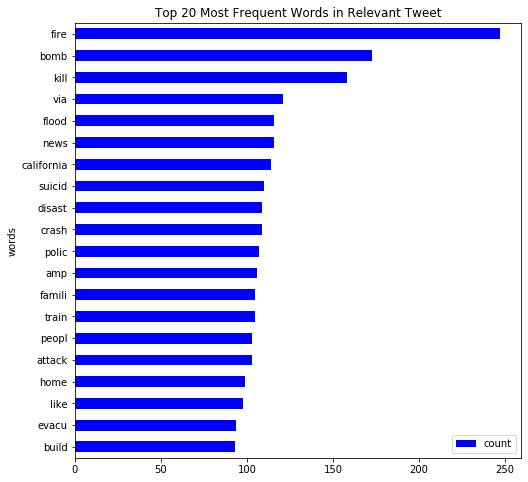
Since the percentage of missing variables are less than 1%, we ignore them as it may not affect the results. In order to analyze the text, we lower all letters and remove unnecessary punctuations and stop words. We also do stemming and here is our processed text data.



In order to evaluate the model we built, we split data into 80% training and 20% testing. 

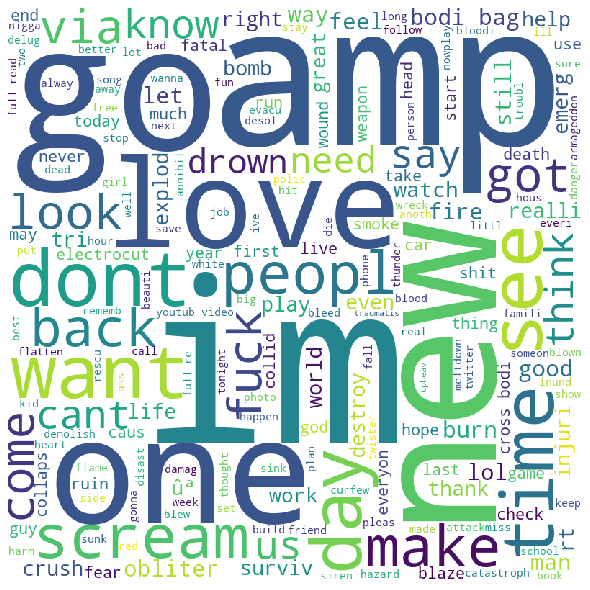
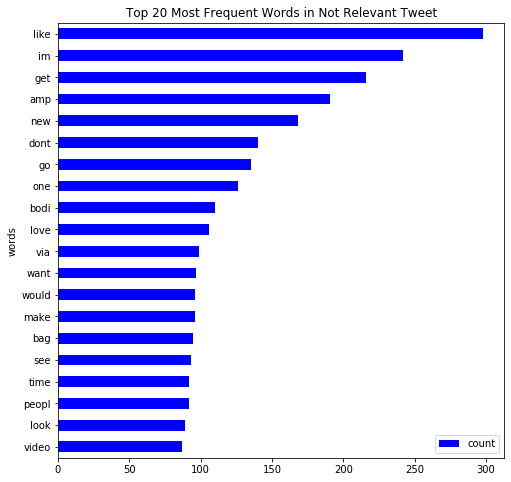
**Descriptive Analysis**

Before analyzing the text, we built WordCloud to visualize the word frequency. We divided the text by **target**, which are Not Relevant and Relevant.

*Figure 1: WordCloud & Word Frequency for Relevant Tweet*

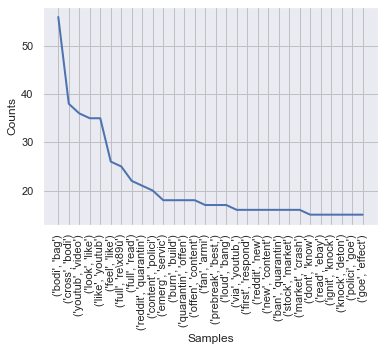
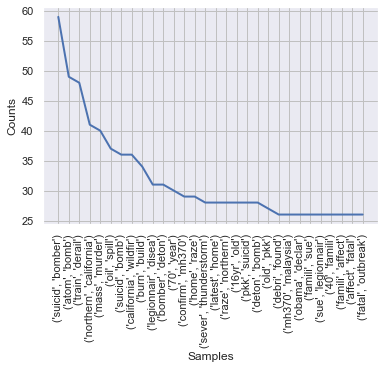
Based on the charts for Relevant tweet, we can see that the most frequent words are mostly related to disaster, such as fire, bomb and flood. Since we use stemming during data preprocessing, words such as ‘disaster’ are shown as ‘disast’, yet this will not affect our analysis.

*Figure 2: WordCloud & Word Frequency for Not Relevant Tweet*

Compare Figure 2 with Figure 1, we can clearly see the difference as WordCloud for Not Relevant tweet does not contain any disaster related word. The word ‘crush’ appeared in Figure 2 WordCloud may sometimes be classified as disaster. There are also some similarities between the two charts, such as the word ‘peopl’.

We also build graphs for visualize the frequent bigrams in tweets and here are the results.

*Figure 3: Graph of 30 most frequent bigrams in 'Not Relevant' and 'Relevant' category*

According to Figure 3, we can find that the most frequent bigrams in ‘Not Relevant’ and ‘Relevant ’is (‘bodi’, ‘bag’) and (‘suicid’, ‘bomber’) respectively. The bigrams in the relevant tweets are diasaster terms while the one in not relevant tweets are scattered in terms of theme.

**Topic Modeling**

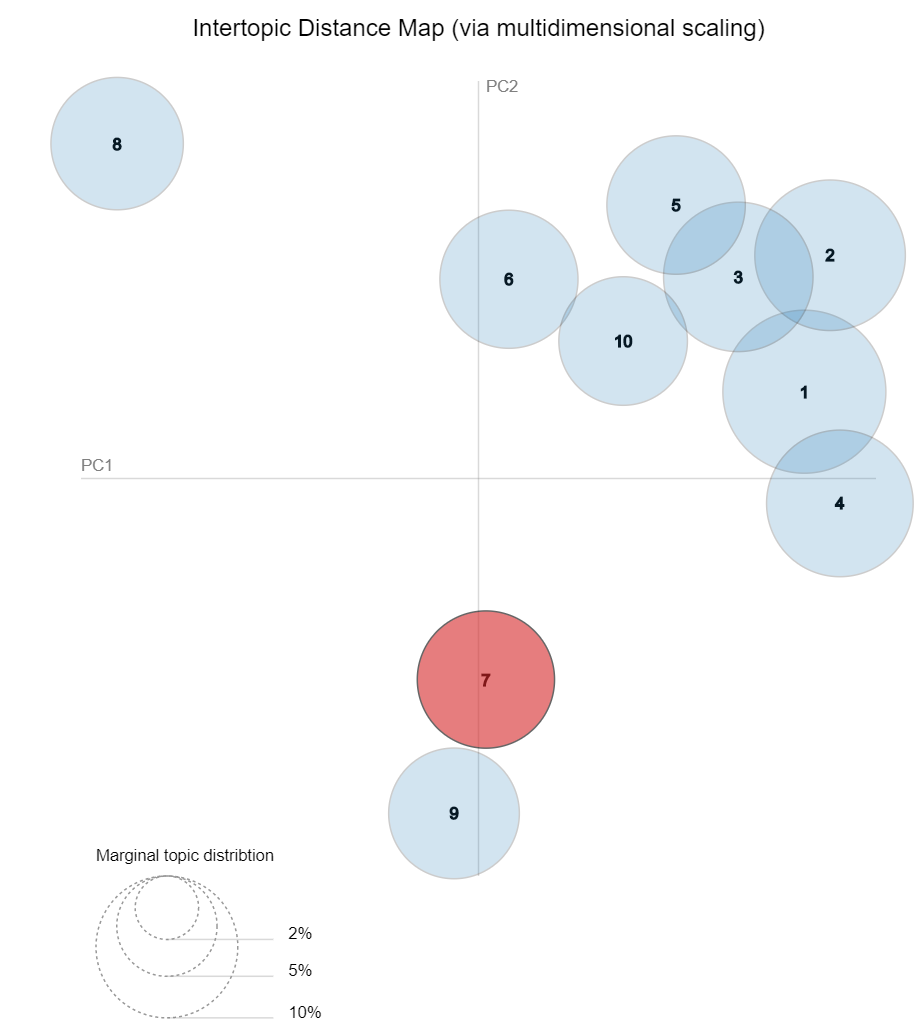
In this part, we are going to analyze the pattern of our text data. Topic modeling is a good idea because this model can generate several topics from the text without pre-classification. Since the LDA model is the most popular model in the topic modeling, we are going to apply the LDA model in our case. The LDA assumes documents are produced from a mixture of topics. Those topics then generate words based on their probability distribution. Therefore, we could use keywords to predict the topic by applying the model. We would like to know if different topics could classify the relevance. Furthermore, we could drill down more patterns by using these topics.

After the tokenization of our text data, we need to decide how many topics we should assign to the LDA model. Originally, we are going to analyze if the specific tweet is relevant to a disaster. Splitting two topics seems work in our case. However, after trying two topics in the LDA model, we found that the LDA model could not classify the text in a proper way. We could not find any unique patterns between two topics. So we decided to assign more topics to the LDA model to see the cluster of different topics. Finally, we chose 10 topics as the input of the LDA model.

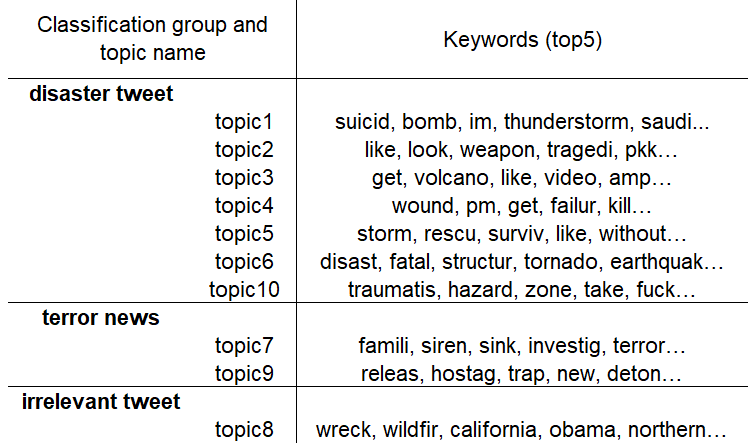
After fitting the text in the LDA model, the keyword result is as the following:



However, these formulas are too hard to read. So we need to visualize the outcome of our LDA model by using the *pyLDAvis* package.



This bubble chart shows the clustering of our topic model. Each bubble represents a topic and there are 10 topics in total. The distance between two bubbles represents the relevance of these two topics. The closer the bubbles are, the stronger the correlation between the two topics. I divided these bubbles into four clusters manually. Here is the result:

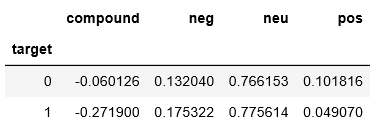


We created that chart according to the distance of bubbles and the keywords under each topic. Furthermore, we also assigned a name to each cluster based on keywords. We categorized the topics into three main primary groups: disaster tweet, terror news, and irrelevant tweets. Therefore, we could use keywords to predict the cluster of each tweet.

**Sentiment Analysis**

Sentiment analysis uses data mining processes and techniques to extract and capture data for analysis in order to discern the subjective opinion of a document or collection of documents.

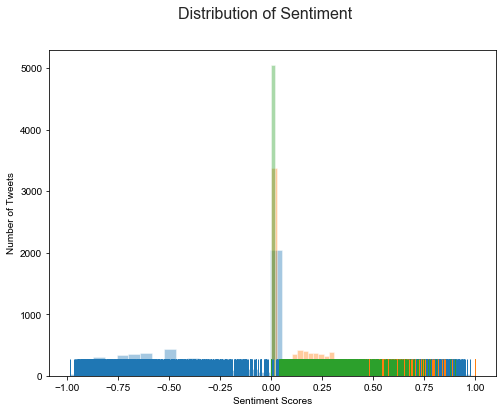
By using NLTK Darth Vader, our Sentiment Analysis result is shown as following:

****

We separate the result into two parts according to the relationship with target: 0 represent not Not Relevant, and 1 represent Relevant.

The Compound score shows a more negative score for relevant tweet (-0.27) versus not relevant tweet (-0.06). It means both relevant tweets and irrelevant tweets are negative, and relevant tweets are more negative than irrelevant tweets.

The negative score and positive score further confirmed this conclusion.

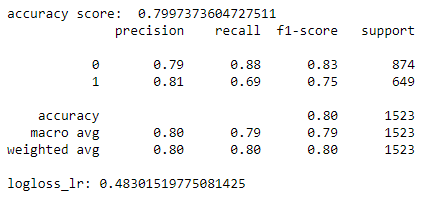
****

This is a simple plot to look into the sentences, the blue represents ‘Compound’, yellow represents ‘Negative’ and green represents ‘Positive’. From the bar chart, we can learn that most Tweets’ sentiment scores are around 0.

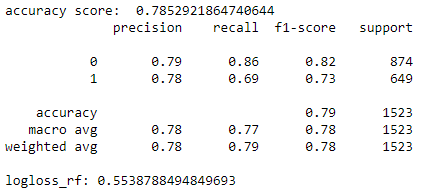
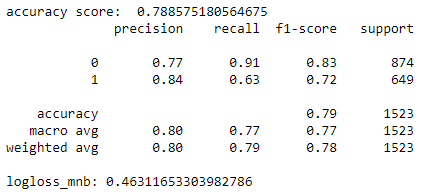
**Classification**

We built three models for classification, which are Logistic Regression (tfidf & word2vec), Random Forest and Naive Bayes. To compare four classifiers and see which model gives the best result, we calculate the accuracy score, precision, recall and F1-score.

***Logistic Regression*  - *TFIDF*** ***Logistic Regression - word2vec***

 A screenshot of a cell phone

Description automatically generated

***Random Forest*   *Naive Bayes*** 

First, we built two logistic regression models to check which kind of vectorization tool (TFIDF and Word2vec) is better. The accuracy score shows that TFIDF is better with 0.7997 accuracy. Thus we are going to use TFIDF to vectorize text data in the following models.

Then we build another two models: Random Forest and Naive Bayes. Comparing the accuracy score, Logistic Regression (TFIDF) has the highest score, which is 0.7997. Further comparing the precision, recall and f1-score, Logistic Regression (TFIDF) still gives the best results among four models for both Relevant and Non Relevant tweet.

Though the Naive Bayes has the lowest Log Loss among these four classifiers, other scores are not as good as Logistic Regression-TFIDF’s.

Overall, we can analyze that Logistic Regression is the best performing model.

**Conclusion**

Our classifiers came back with rather good accuracy across all four. Among them, the Logistic Regression-TFIDF has the highest accuracy score (0.7997). So, we think the Logistic Regression is the best performing model. However, to effectively use this model as a way to detect new, relevant disasters to report on, the accuracy needs to be higher because the cost of false information is high.

According to Sentiment Analysis, the result showed a more negative sentiment versus not relevant tweet. It means both relevant tweets and irrelevant tweets are negative, and relevant tweets are more negative than irrelevant tweets. We can draw the same conclusion by analyzing negative scores and positive scores.

**Scope & Limitation**

The largest accuracy for models we build is only 0.7997, which is not large enough. We may need to find methods to improve the accuracy. We could train on more data, add more language pre-processing or try other tools to boost the result. For example, we could try adding more attributes to the data than just using the ‘text’ of the tweet to see if it could increase accuracy.

Since we use stemming during preprocess data, words such as ‘suicide’ are showed as ‘suicid’, which we may need to fix if we want more precise results.

**Future Direction**

We built a classifier only to identify whether a tweet is related to disaster this time. However, there are some other interesting things in the original dataset. For example, there is a column called “keyword” that records the specific types of disasters associated with each tweet. We could also use this column as the target variable to build another classification model. By applying the new model, we can identify further information about the specific type of disaster. One setback to twitter is that tweets are capped at 140 words. Thus, twitter users tend to write short tweets which could make it hard to mine data from. For example, if a user is referring to something ‘lit’, it could mean a fire or modern slang for ‘fun’. However, there is potential in this model. Perhaps with more fine-tuning news firm could incorporate this into their intelligence resources: finding out new sources about local disaster through local people who shared about it on social media. At the end of the day, this model doesn’t have to be only twitter. To get more validation and sources, other social media sources such as Instagram/ facebook could be used.

# **Works Cited**

Isaienkov, Yaroslav. *Starter NLP: SVM & tf-idf*. January 2020. 22 April 2020. <https://www.kaggle.com/ihelon/starter-nlp-svm-tf-idf/notebook>.

Pangeni, Kamal. *Disaster Tweets with LR,RF..DNN*. January 2020. 22 April 2020. <https://www.kaggle.com/kamalpangeni/disaster-tweets-with-lr-rf-dnn>.

Mokin, Vitalii. “NLP - EDA, Bag of Words, TF IDF, GloVe, BERT.” *Kaggle*, Kaggle, 4 Apr. 2020, www.kaggle.com/vbmokin/nlp-eda-bag-of-words-tf-idf-glove-bert.