Final Report: NLP with disaster tweet

Ting-An Andy Wang, Jie Huang

Introduction

Social media, when disasters strike, becomes a very busy place. People share news, ask for help, or talk about what's happening. This is super fast but sometimes not all true. Our study, called "Natural Language Processing with Disaster Tweets," is about making sure we can find and trust the tweets that are really about disasters. We need this because it helps people who are trying to help others in emergencies. They need good information quickly.

We see that social media can be really helpful but also can cause problems if the information is wrong. That's why our work is important. We want to use what is called NLP, or Natural Language Processing, to look at lots of tweets and tell which ones are really about serious situations. It can help a lot because then the people who are trying to help know what is true and what is just someone talking. It's not easy because the way people write on social media is very different from how we write in books or newspapers. They use short forms, pictures like emojis, and sometimes they don't follow the normal rules of how to write words or sentences.
In our work, we have to think about these things. We have to understand how people use words in different ways when they are in a hurry or scared because of a disaster. Our goal is to make a tool that can read tweets and pick out the ones that are really important for emergency workers. This tool is not just about using computers to read words. It's also about understanding language very well.

We have some good examples from other people's work that we can use to help us. There are some new ways to teach computers how to tell the difference between two things, which is what we need to do with tweets—tell if a tweet is about a real disaster or not. Inspired by recent advancements in binary classification, we choose to refer to two related works in this field. Rayan Wali's research on the "Xtreme Margin" loss function introduces a novel approach to model training that allows for adjustments through its tunable hyperparameters, potentially improving metrics such as precision and AUC score (2022). Additionally, Ala'raj and his team have shown how improvements in binary classification can be achieved using k-NN proximity graphs in 2020. Their work demonstrates the effectiveness of proximity-based filtering and dynamic ensemble selection in increasing classification accuracy across various datasets.

Our contribution is enhancing disaster tweet detection using advanced NLP techniques for emergency response. We have developed a logistic regression model to predict whether a tweet is related to an actual disaster or not. We also visualize our outcomes to increase the reliability and usefulness of information on our model.

Approach

We began our task by importing multiple libraries that we will be using later. We import basic packages such as pandas, numpy, and matplotlib to read in data files and do further analysis. We also import NLTK packages as a preparation because we need to preprocess our text data. Last, we import multiple scikit-learn packages that allow us to apply traditional machine learning models and many other applications. After we import the necessary libraries, we read in the csv data file and check our dataset. Our data consist of five variables, including id, keyword, location, text, and target (see **figure 1**). The id variable contains a list of numbers to identify each tweet, and the keyword variable represents a particular keyword from the tweet. The location is simply the location the tweet was sent from, and the text represents the content of the tweet. Our target consists of 0s and 1s to show whether a tweet is about a real disaster or not.

We then proceed to the next step to check the missing values and balance of the dataset. We found that there are eighty-seven missing values for the keyword category and three thousand six hundred and thirty eight missing values for the location category (see **figure 2**). In order to complete our dataset, we fill in the word "No_data" to each missing value. Moreover, when we created a bar chart to check the balance of the dataset, we found that the total number of actual disaster tweets is larger than the total number of false disaster tweets (see **figure 3**). This imbalanced dataset implies us to focus more on recall and precision score when evaluating the model later on. After a brief look at our dataset, we started preprocessing the text data to ensure it was in a suitable format for analysis. The process involved removing information like stop words and special character, stemming, lemmatizing, and tokenizing. First, we remove punctuation, special characters, URLs and transform all letters to lowercase to make each tweet look more simple. Second, we use the stemming method to make each word shorter to their root form. Third, we use the lemmatizing method to change words to their basic dictionary form. And last, we break down the text into individual tokens by using the tokenization method (see **figure 4**).

Once the text was preprocessed, we split our dataset and try to represent documents with TF-IDF representation. The TF-IDF method, which scales the frequency by how unique a word is to the document, highlights the words that could be pivotal in understanding the content's unique aspects. We use the TfidfVectorizer function under sklearn's feature_extraction function to convert each word into a normalized number, which is between 0 and 1. We also decided to not perform a z-score normalization because TF-IDF itself is already a kind of a normalization method.

Since our task is to predict whether the content of a tweet is truly about disaster or not, the outcome would be shown as 1 or 0. This also fits into the binary classification applications. For our project, we use three different models and compare the best one based on their performance. First, we use k-NN and Use the 5-fold cross-validation to select the hyperparameter K to create the model. The score for the hyperparameter K is 0.6807916181606519. After getting our hyperparameter, we start our training phase and testing phase. Since we have an imbalanced dataset, we not only use accuracy score but also use F1, recall and precision. In addition to a regular recall score, we hope to further compute the probability of predicting 0s correctly. So, we also add a "recallForZero" in our evaluation phase. The next approach we choose to build our model is Logistic Regression. It has a similar procedure as k-NN where we choose 6-fold cross-validation to select the hyperparameter K to create the model. We then train our model by using the LogisticRegression function and evaluate our model by calculating the accuracy, macro F1, micro F1, recall, and recallForZero scores. The last approach we choose is the k-means clustering method. We use it to do clustering and find the 10 most representative words in each cluster. We first choose three clusters to classify and randomly initialize the cluster centroid. Then we compute the distance for each sample point in order to assign each of them to the nearest cluster centroid. After that, we compute the mean values with each cluster and update the cluster centroid. By repeating the process multiple times, we are able to compute the normalized mutual information score of the K-means method and find tenth closest means to the centroids in each cluster.

Result

Precision is about how sure we can be that when our model says a tweet is about a real disaster, it's actually true. It's not good if our model makes many mistakes, telling us a tweet is about a disaster when it's not. It can cause panic or make people worried for no reason. That's why precision matters. There's also a special measure called the F1 score. It combines recall and precision to give us one number that tells us how balanced our model is. The F1 score is like an all-around grade for our model, telling us how well it's doing overall.

First we look at the k-NN model. This model works by looking at tweets and finding which ones are most similar to each other. But it didn't do very well. Its recall was 0.3978378378378378, which is super low, meaning it missed more real disaster tweets. And its F1 was 0.5423728813559322, which was not as high as well, so it was more often confused by tweets that were not really about disasters (see **figure 5**). It's like having another friend who is not as good at the guessing game; they get a few right, but they make many mistakes. Now, logistic

regression had good scores for accuracy, recall, rellForZero, precision, and the macro/micro F1 score (see **figure 6**). It was the best out of the models we have with an average of 0.8 on each score. It found many of the real disaster tweets and didn't get fooled often by the other tweets.

The k-means model was the one that tried to group the tweets into clusters. This was really interesting because it showed us how some words tend to show up together in the tweets. But when it came to finding disaster tweets, k-means wasn't very good. The number that tells us how well it did, the NMI score was very low (see **figure 7**). It's like when you try to sort your clothes into piles, but in the end, you're not sure which pile is for which kind of clothes.

The Recall, which shows how well our model identifies real disaster tweets, and Recall for Zero, which indicates how accurately it identifies non-disaster tweets. High recall means our model is good at spotting disaster tweets, but it's also important to have a good Recall for Zero to avoid mistaking normal tweets for disasters (see **figure 8**). The coefficient plot helps us see which words increase the chances of a tweet being classified as a disaster and which decrease is it. Words like “flood” and “earthquake” for instance have positive weights, making them strong indicators of disaster tweets. Normal everyday words usually have negative weights, showing they are less likely to be in disaster tweets.(see **figure 9**).

Conclusion

In conclusion, our study, "Natural Language Processing with Disaster Tweets," shows how important it is to have correct information predicting disasters. We used NLP techniques to make a logistic regression model that can tell the difference between tweets about real disasters and other tweets. It is very important for people who respond to emergencies because they need quick and reliable information to make good decisions.

Our results show that it is crucial to be precise in disaster response. Our model must avoid false alarms to prevent panic and to use resources wisely. By comparing different machine learning models like k-NN, logistic regression, and k-means clustering, we learned about the strengths and weaknesses of each method. Logistic regression was the best because it was the most accurate and reliable.

This research helps not just in managing disasters but also in using machine learning in real situations. By making our model better at quickly analyzing tweets from social media, we help make information systems stronger for responding to disasters.

References

Ala’raj, Maher et al. (2020, March 5). *Improving binary classification using filtering based on k-NN Proximity Graphs - Journal of Big Data*. J Big Data 7. https://doi.org/10.1186/s40537-020-00297-7.

Machine Learning TV(2019). *NLP - Text Preprocessing and Text Classification (using Python)*. YouTube. Retrieved from https://youtu.be/nxhCyeRR75Q?si=dOctSNsaHcgarbEH**.**

Rana Abdullah (2020, Dec 2). *Natural Language Processing “Disaster Tweets”*. Medium. Retrieved from https://medium.com/swlh/natural-language-processing-disaster-tweets-57b6ceb44b5d**.**

Wali, Rayan (2022, October 31). *Xtreme margin: A tunable loss function for binary classification problems.* Cornell University. Retrieved from https://doi.org/10.48550/arXiv.2211.00176.

Appendix

Figure 1 - check categorical feature

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10876 entries, 0 to 10875
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   id        10876 non-null  int64
 1   keyword   10789 non-null  object
 2   location  7238 non-null   object
 3   text      10876 non-null  object
 4   target    10876 non-null  int64
dtypes: int64(2), object(3)
memory usage: 425.0+ KB
None
```

Figure 2 - check missing values

```
id             0
keyword       87
location    3638
text           0
target         0
dtype: int64
```

Figure 3 - check balance/imbalance

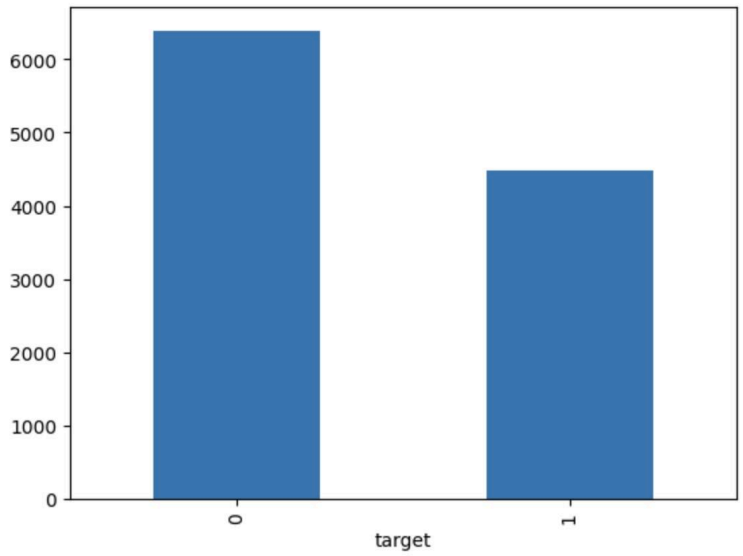

Figure 4 - preprocessed dataset

```
df.head()
```

	id	keyword	location	text	target	text_tokens
0	1	nodata	No_data	our deeds are the reason of this earthquake ma...	1	[o, u, r, , d, e, e, d, s, , a, r, e, , t, ...
1	4	nodata	No_data	forest fire near la ronge sask canada	1	[f, o, r, e, s, t, , f, i, r, e, , n, e, a, ...
2	5	nodata	No_data	all residents asked to shelter in place are be...	1	[a, l, l, , r, e, s, i, d, e, n, t, s, , a, ...
3	6	nodata	No_data	13000 people receive wildfires evacuation orde...	1	[1, 3, 0, 0, 0, , p, e, o, p, l, e, , r, e, ...
4	7	nodata	No_data	just got sent this photo from ruby alaska as s...	1	[j, u, s, t, , g, o, t, , s, e, n, t, , t, ...

Figure 5 - Accuracy, F1, recall, recallForZero and precision score for k-NN model

```
print(f'accuracy = {acc}\nF1 = {f1}\nr
```

```
accuracy = 0.7283464566929134
F1 = 0.5423728813559322
recall_score = 0.3978378378378378
recall_Zero = 0.9529757531227039
precision_score = 0.8518518518518519
```

Figure 6 - Accuracy, macro/micro F1, recall, recallForZero and precision score for logistic regression model

```
{'C': 5}
accuracy = 0.8438320209973753
macro_fi = 0.8354137646674693
micro_fi = 0.8438320209973753
recall_score = 0.7632432432432432
recall_for_Zeros = 0.8986039676708303
precision_score = 0.8364928909952607
```

Figure 7 - The normalized mutual information score and 10th representative words for k-means method

```
(5, 15857)
The normalized mutual information score of the K-means method is 0.0388
```

Cluster: 1	Cluster: 2	Cluster: 3
im	swallowed	suicide
just	minute	bomber
video	airport	detonated
amp	sandstorm	16yr
new	watch	pkk
dont	fahlowcw	saudi
people	fadc	mosque
disaster	faded	old
news	fading	bomb
liked	fag	trench

Figure 8 - Comparison of predicting 1s and 0s

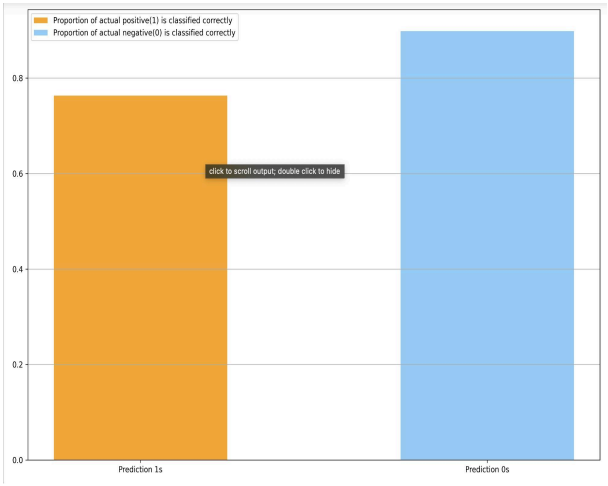

Figure 9 - Influence on parameters

```
[[-0.25224385  0.11500984  0.64699378 -0.30711388 -0.56013076  0.13026355
   0.13026355  0.88044522  1.07655418  0.90453443  0.35894411  0.28021027
  -0.8158896  -0.27330195  0.4248359  -0.21525863  0.33852602 -0.30711388
   0.83905526  0.27507199  0.5043028   0.26258376  0.16326697  0.25965087
   0.38748307 -0.36797605  0.28021027 -1.00294667 -0.38590621 -0.19001921]]
```