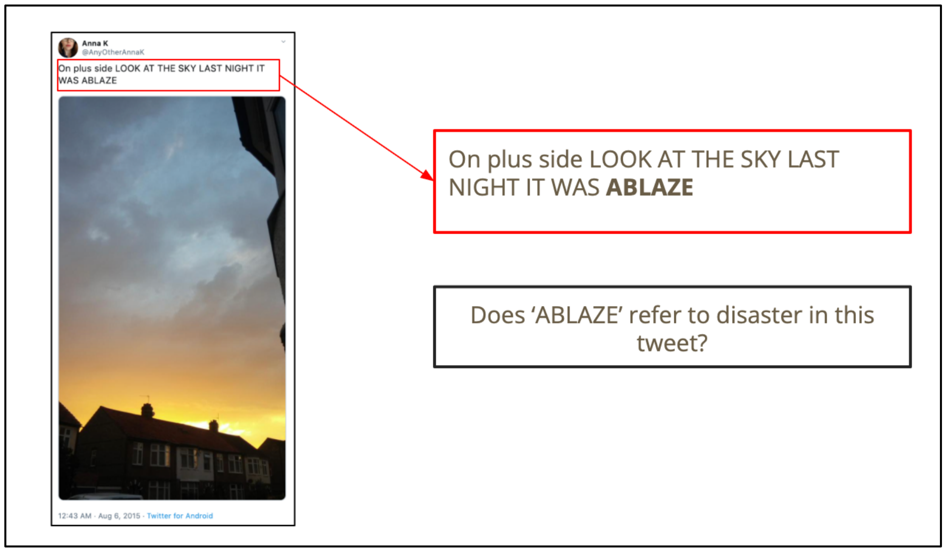
Natural Language Processing with Disaster Tweets

Tarannum Zaki

1 Motivation

Social media is one of the most popular ways to share information these days. Social media is like a virtual social network where people share different news and views, thoughts and ideas, images and videos all over the world. Furthermore, social media applications are easily available on smart phones which make it a more convenient way of sharing content in real-time for the users. There are several widely used social media platforms like Twitter, Facebook, Instagram etc. Though the purpose of these social media platforms is similar, that is, content sharing, different social media platforms are popular for different types of content sharing among the consumers. For instance, a research on Twitter and Facebook regarding news consumption shows that consumers use Twitter more to stay updated on breaking news rather than Facebook [1].

As a result, Twitter has become an important and popular social media platform worldwide for sharing content specially during state of emergency. One of such emergencies are different types of disasters including both natural (earthquake, wildfire, flood etc.) and man-made (mass shooting, chemical explosion etc.) disasters. As social media platforms enable users to share information about such disasters in real-time, different agencies like news agencies and disaster relief organizations are interested to programmatically monitor Twitter. Monitoring Twitter feeds would further help these agencies to take necessary actions during state of emergency. Now, in case of sharing information regarding disasters through textual description, it might not always refer to an actual disaster. For example, Fig. 1 shows a screenshot of a tweet where the author uses the word ‘ABLAZE’ which means ‘burning fiercely’ and can be related to fire related disasters. A human can easily understand that the word ‘ABLAZE’ is used as a metaphor here, and it does not refer to anything that is related to disaster. However, a machine cannot clearly perceive this, that is, programmatically it might be difficult to identify whether such textual description of a tweet refers to an actual disaster related content or not. This is the motivation of the research project which is basically the essence of the Kaggle competition project about predicting programmatically whether the tweets are disaster related or not [2].


Anna K
@AnyOtherAnnaK
On plus side LOOK AT THE SKY LAST NIGHT IT WAS ABLAZE
12:43 AM · Aug 6, 2015 · Twitter for Android

On plus side LOOK AT THE SKY LAST NIGHT IT WAS **ABLAZE**

Does ‘ABLAZE’ refer to disaster in this tweet?

Figure 1. Screenshot of a tweet.

2 Problem

2.1 Problem Definition

The problem definition of this research can be stated as to predict programmatically, to be more precise, by using machine learning models whether tweets are related to disaster or not. The research questions of the project are formulated below:

RQ1: Is the tweet about a real disaster or not?
RQ2: How effectively the prediction is performed using machine learning models?

Additionally, we are required to incorporate some information retrieval method that would retrieve relevant tweets based on a specific query. We define the tasks as follows:

Task1: Search relevant disaster related tweets from the test data based on a query that includes a disaster related keyword.
Task2: Evaluate how effectively the relevant tweets are retrieved.

2.2 Related Works

The data type of tweets is textual in nature. Natural Language Processing (NLP) is a technique to process textual data so that the machine can further process it. There exists some remarkable research works related to natural language processing and machine learning that concerns analysis on textual data. Some of these analyses are – sentiment analysis, topic analysis, intent detection etc. [3]. Among these analyses, several research works concern working with the Twitter dataset provided by the Kaggle competition that is aimed to identify whether a tweet is disaster related or not.

Bikku et al. performed an analysis to detect whether tweets are disaster related or not on the Twitter dataset [4]. They were able to achieve 80.5% accuracy using an optimized SVM model on a reduced part of the dataset. Deepa Lakshmi and Velmurugan also performed an analysis on the same dataset [5]. They used the TF-IDF transformer for feature extraction and concluded that SVM gave better results in terms of precision, accuracy, and F1-score. Alvarado and Solano proposed an emergency chatbot based on RoBERTa model that would detect whether an emergency was real or not [6]. They used the same Twitter dataset and showed that the chatbot resulted in slightly better accuracy compared to a multinomial naïve bayes classifier. Deb and Chanda performed a comparative analysis on contextual and context-free embeddings for predicting disaster tweets from the twitter dataset [7]. They found that contextual embeddings like BERT gives better F1-score and accuracy than context-free embedding methods. Additionally, their qualitative study showed that traditional context-free embedding method predicts disaster related tweet well if the disaster related keyword exists in the tweet. On the contrary, contextual embedding method can predict disaster related tweet just by understanding the context of the words. Bhere et al. experimented with the same dataset and showed that using Word2Vec as feature vector and CNN as classifier, disaster related tweets were predicted with an accuracy of 84% [8].

Related research works show that both traditional machine learning models and deep learning models have been used as a classifier to predict whether tweets are disaster related or not. For the feature extraction step, TF-IDF and pre-trained word embedding like Word2Vec have been used as feature vector. One of the limitations of using TF-IDF as feature vector is that it does not capture semantic meaning of words inherently [9]. Again, word embeddings like Word2Vec can capture semantic relationships between words [10]. Our focus for this project is perform analysis by using traditional algorithms as classifier and using Word2Vec with TF-IDF combinedly as feature vector.

3 Method

The methodology for the project involves several tasks – text pre-processing, text numericalization, text classification, and evaluation. Fig. 2 shows the framework for the tasks. The data is textual description and unstructured, so it must go through some cleaning process. Tweet texts often involve hashtags, URLs, emojis, and other symbols. We clean the text by removing numbers, hashtags, URLs, punctuation, and stop words. Along with the text cleaning, the text pre-processing task involves converting the texts into lowercase and performing tokenization. Next, the machine is unable to process text data, so it is required to transform into a format that is machine-readable. TF-IDF is used for this purpose which provides a calculation of scores in a vector format and highlight each word's relevance in the entire document [11]. Along with the normalized TF-IDF scores, we are using a pre-trained word embedding model Word2Vec with an intent to alleviate the limitation of TF-IDF regarding the encoding of semantic relationship between words. Then these combined vectors are used as feature vectors to train and test machine learning models. We considered some traditional machine learning models for our experiment which are widely used for text classification [12]. These are – logistic regression, random forest, k-nearest neighbor (KNN), multinomial naïve bias, support vector machine (SVM), and gradient boosting. Finally, the classification results are evaluated using F1-score.

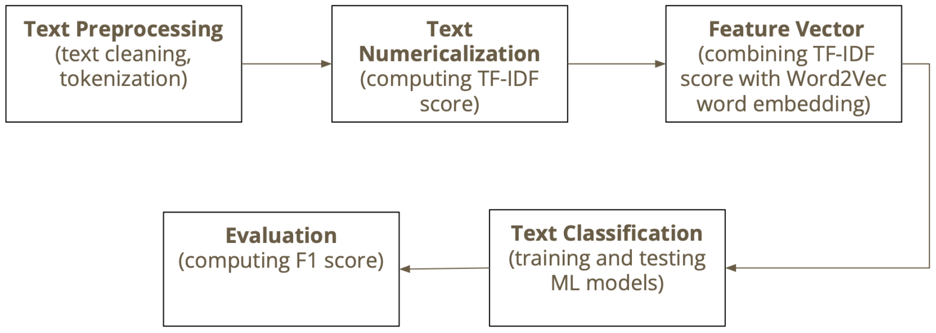


Figure 2. Framework of tasks for classification problem.

For the information retrieval task, we first extract top 20 disaster related keywords (unigrams) from the provided dataset based on normalized TF-IDF scores which are used as search keyword for the query. Next, we filter the train data having disaster related labels only and the query keyword. We also filter the classified test data for the query keyword. Then, we compute the cosine similarity between the each of the combined vectors of test data and a weighted average vector of train data. Lastly, we retrieve the top 20 relevant tweets based on cosine similarity scores and evaluate using F1-score. Fig. 3 shows the framework of task for this information retrieval part.

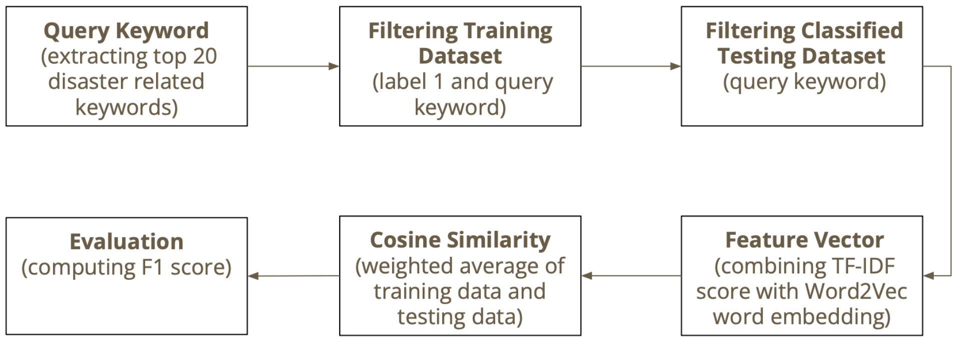
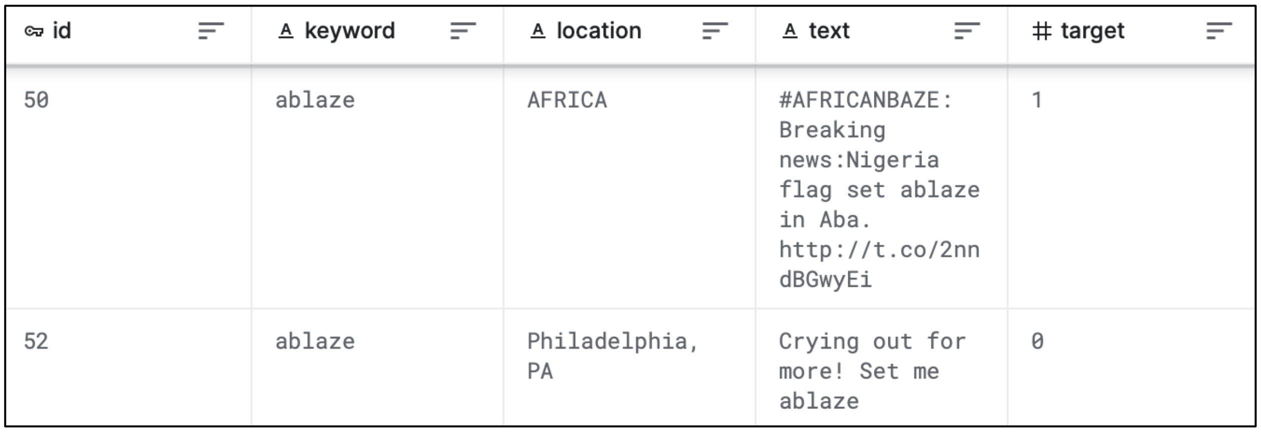

Query Keyword
(extracting top 20 disaster related keywords)
Filtering Training Dataset
(label 1 and query keyword)
Filtering Classified Testing Dataset
(query keyword)
Feature Vector
(combining TF-IDF score with Word2Vec word embedding)
Cosine Similarity
(weighted average of training data and testing data)
Evaluation
(computing F1 score)

Figure 3. Framework of tasks for information retrieval.

4 Data

We use the Twitter dataset provided by the Kaggle competition [2] for this research project. This dataset contains about 10,000 hand-classified tweets. The dataset is split into training and testing data with 7,613 (70%) and 3,263 (30%) data respectively. The data fields for the training data are – ID, keyword, location, tweet text, and target (Fig. 4).



id	keyword	location	text	target
50	ablaze	AFRICA	#AFRICANBAZE: Breaking news:Nigeria flag set ablaze in Aba. http://t.co/2nndBGwyEi	1
52	ablaze	Philadelphia, PA	Crying out for more! Set me ablaze	0

Figure 4. Snap of the training dataset.

We perform an exploratory data analysis (EDA) on the dataset. Both training and testing dataset have some null values for the data fields 'keyword' and 'location.' The target count for the dataset is shown in Fig. 5 which indicates the training set is a moderately balanced dataset having 57% non-disaster related tweets (label 0) and 43% disaster related tweets (label 1).

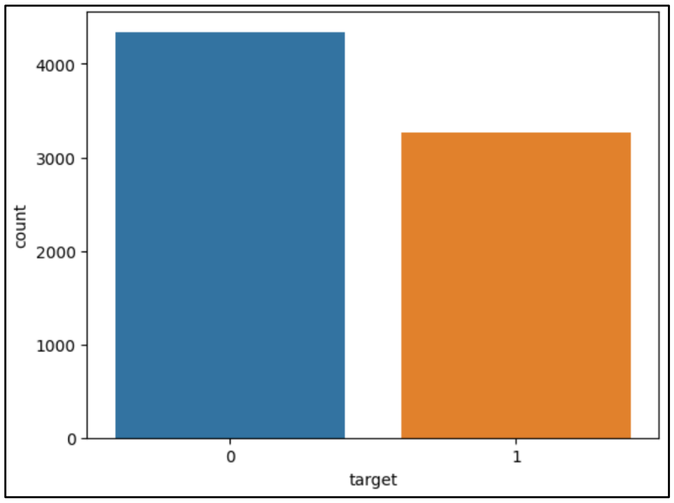


Figure 5. Distribution of labels in the training dataset.

5 Experiments and Evaluation

We focus on working with the 'text' field of the dataset only. As the data type of this column belongs to textual description, it requires some preprocessing. The tweet texts are split into tokens which is termed as the tokenization process. The text cleaning process involves converting all to lowercase; removing URLs, punctuation, hashtags, numbers, and stop words. Table 1 shows an example of a tweet text from the dataset, before and after performing the text preprocessing step.

Table 1. Example of a tweet text, before and after the text preprocessing step.

Text	13,000 people receive #wildfires evacuation orders in California
Cleaned Text	people receive wildfires evacuation orders california

Next, the training dataset is split into 55% training data, 25% validation data, and 20% testing data. The input feature is the text and the target is the label. The feature vector is a combination of the TF-IDF score and the word embeddings. First, a normalized TF-IDF vectorization is computed for each of the datasets. Then, the word embeddings are obtained using the Word2Vec word embedding model. Finally, the TF-IDF features are combined with the word embeddings. The combined vector has a shape having rows same as the input arrays, and the number of columns is the sum of the original columns from TF-IDF features and word embeddings.

Using the combined feature vector, the selected machine learning models are trained and tested. The performance of the ML models is evaluated using F1-score. The equation of F11-score is given as follows:

$$F1\text{-}score = 2 * \frac{precision * recall}{precision + recall}$$

Table 2 shows the precision, recall, and F1-score of the ML models for predicting label 1 and 0.

Table 2. Performance of ML models on training dataset.

Models	Labels	Precision	Recall	F1-score
Logistic Regression	0	0.79	0.87	0.83
	1	0.80	0.69	0.74
Random Forest	0	0.77	0.91	0.84
	1	0.84	0.64	0.73
KNN	0	0.82	0.81	0.81
	1	0.75	0.75	0.75
Multinomial Naïve Bayes	0	0.80	0.88	0.84
	1	0.82	0.75	0.75
SVM	0	0.79	0.88	0.83
	1	0.81	0.69	0.74
Gradient Boosting	0	0.80	0.79	0.79
	1	0.80	0.80	0.80

Figs. 6 and 7 show the bar charts for predicting label 1 and 0 respectively, depicting the F1-score of the machine learning models regarding their performance.

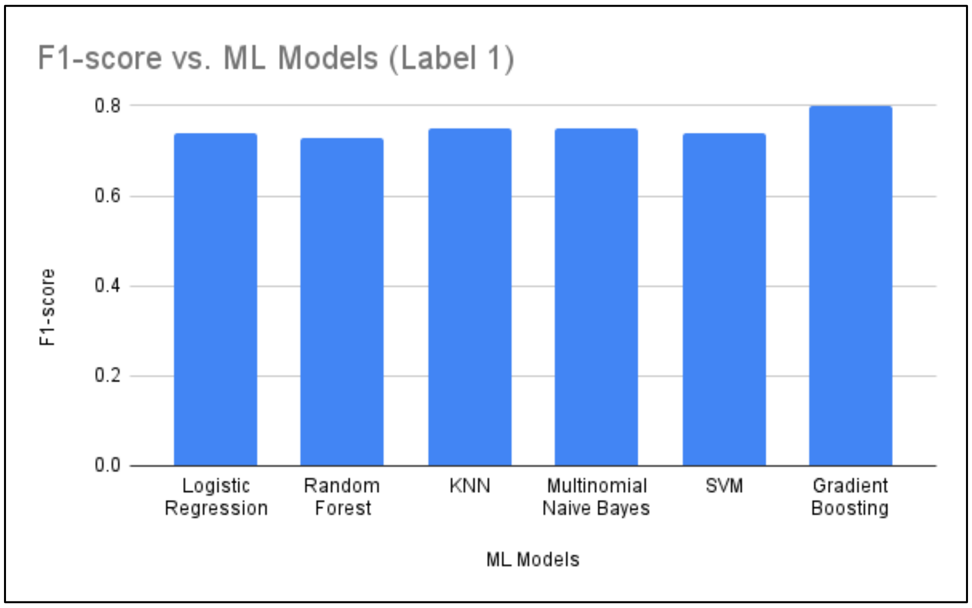


Figure 6. F1-score of the ML models for predicting label 1 on the training dataset.

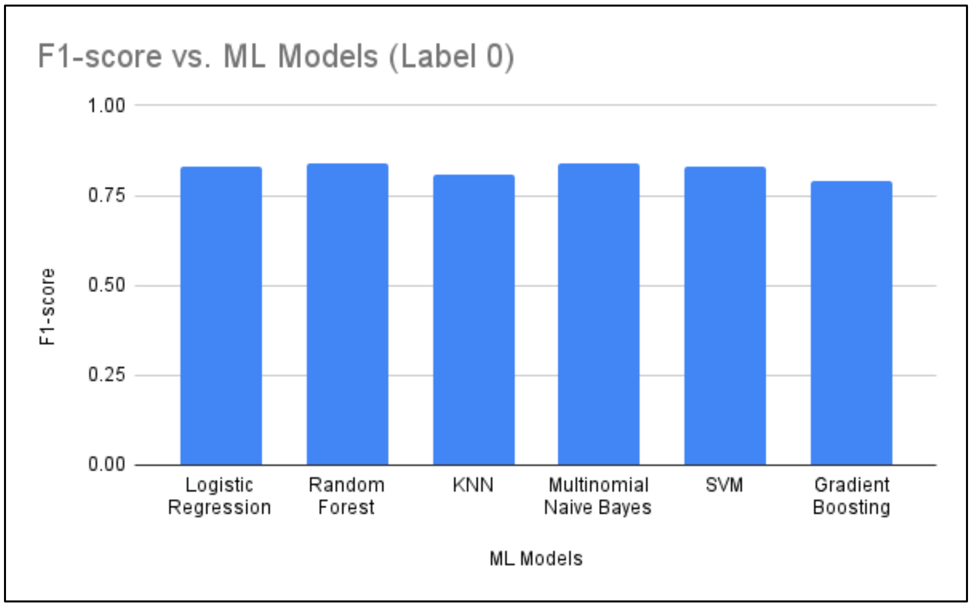


Figure 7. F1-score of the ML models for predicting label 0 on the training dataset.

For the information retrieval task, we first extract the top 20 disaster related keywords (label 1) from the training dataset based on normalized TF-IDF score. The keywords are limited to unigrams. Fig. 8 shows the distribution of the top 20 disaster related keywords obtained from the training dataset.

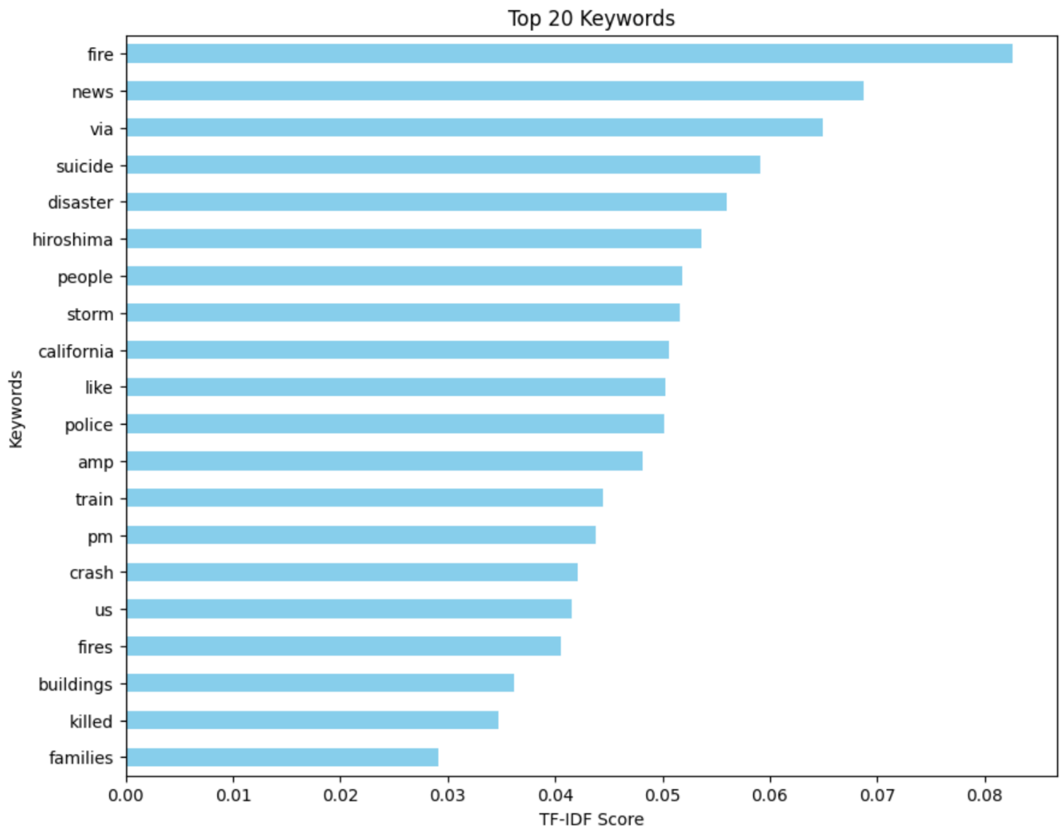
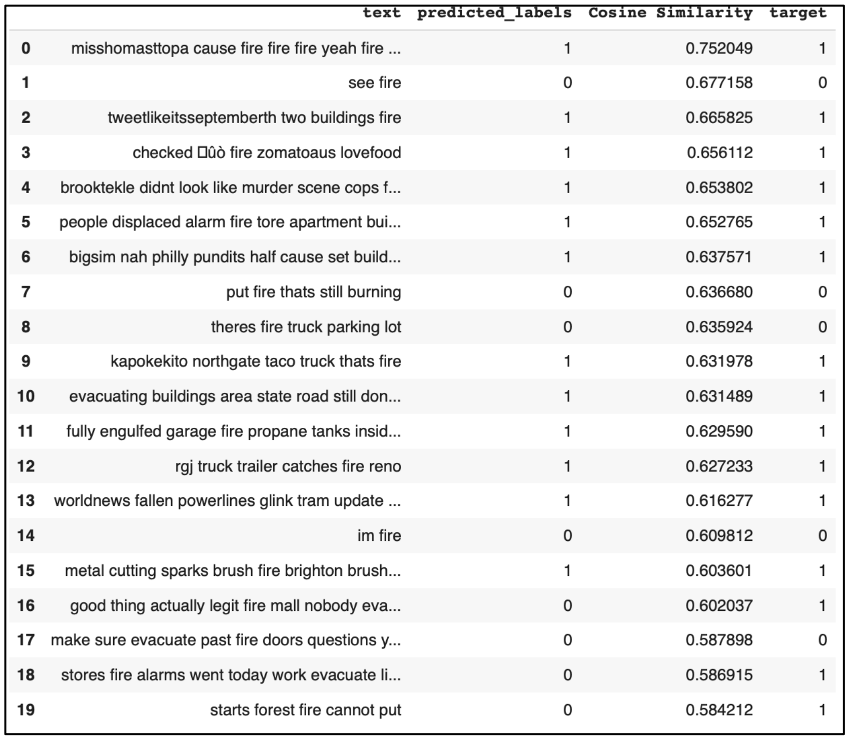


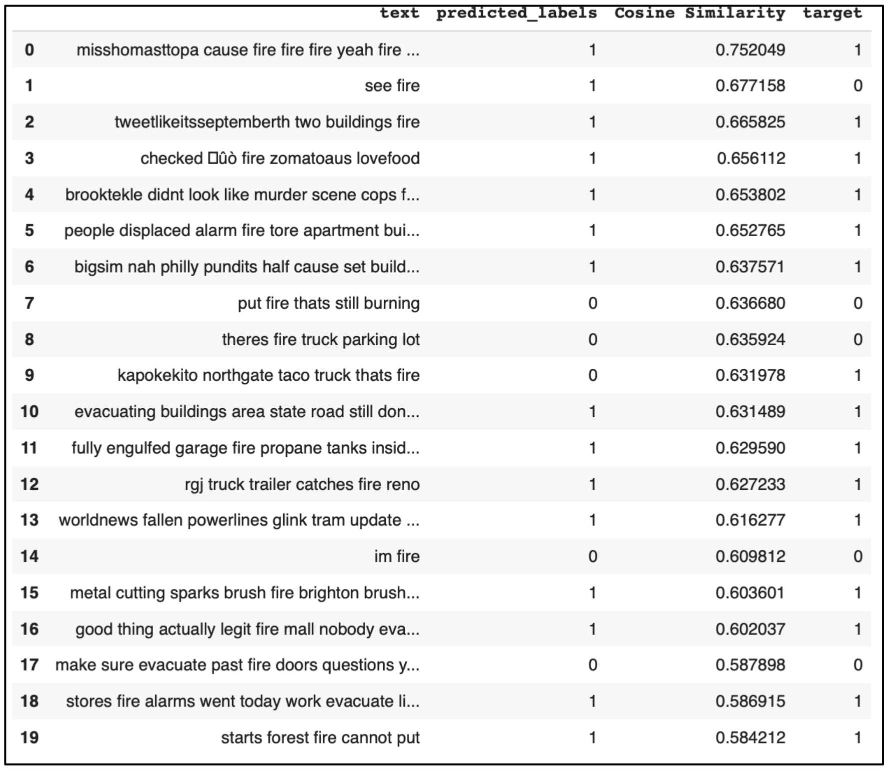
Figure 8. Distribution of top 20 disaster related keywords based on normalized TF-IDF score.

We can see from the distribution that 'fire' is the topmost disaster related keyword used on the tweet text of the training dataset. We performed an experiment using 'fire' as the query keyword on the training dataset (i.e., split into train-validation-test dataset) using the same set of ML models. To retrieve relevant tweet texts based on the query, that is, disaster related tweets having the keyword 'fire' for this experiment, we use the cosine similarity score. The cosine similarity score is computed between the weighted average of the combined vector of the filtered training data and combined vector of each of the filtered testing data. Finally, the evaluation regarding how effectively the information retrieval is performed is done using F1-score for top 20 results. The top 20 retrieved results using multinomial naïve bayes and KNN are shown in Figs. 9 and 10 respectively for example. The performance of the models based on information retrieval task are shown in Table 3.



	text	predicted_labels	Cosine Similarity	target
0	misshomasttopa cause fire fire fire yeah fire ...	1	0.752049	1
1	see fire	0	0.677158	0
2	tweetlikeitsseptemberth two buildings fire	1	0.665825	1
3	checked □ûò fire zomatoaus lovefood	1	0.656112	1
4	brooktekle didnt look like murder scene cops f...	1	0.653802	1
5	people displaced alarm fire tore apartment bui...	1	0.652765	1
6	bigsim nah philly pundits half cause set build...	1	0.637571	1
7	put fire thats still burning	0	0.636680	0
8	theres fire truck parking lot	0	0.635924	0
9	kapokekito northgate taco truck thats fire	1	0.631978	1
10	evacuating buildings area state road still don...	1	0.631489	1
11	fully engulfed garage fire propane tanks insid...	1	0.629590	1
12	rgj truck trailer catches fire reno	1	0.627233	1
13	worldnews fallen powerlines glink tram update ...	1	0.616277	1
14	im fire	0	0.609812	0
15	metal cutting sparks brush fire brighton brush...	1	0.603601	1
16	good thing actually legit fire mall nobody eva...	0	0.602037	1
17	make sure evacuate past fire doors questions y...	0	0.587898	0
18	stores fire alarms went today work evacuate li...	0	0.586915	1
19	starts forest fire cannot put	0	0.584212	1

Figure 9. Top 20 disaster related tweets retrieved using the training dataset (multinomial naïve bayes).



	text	predicted_labels	Cosine Similarity	target
0	misshomasttopa cause fire fire fire yeah fire ...	1	0.752049	1
1	see fire	1	0.677158	0
2	tweetlikeitsseptemberth two buildings fire	1	0.665825	1
3	checked □ûò fire zomatoaus lovefood	1	0.656112	1
4	brooktekle didnt look like murder scene cops f...	1	0.653802	1
5	people displaced alarm fire tore apartment bui...	1	0.652765	1
6	bigsim nah philly pundits half cause set build...	1	0.637571	1
7	put fire thats still burning	0	0.636680	0
8	theres fire truck parking lot	0	0.635924	0
9	kapokekito northgate taco truck thats fire	0	0.631978	1
10	evacuating buildings area state road still don...	1	0.631489	1
11	fully engulfed garage fire propane tanks insid...	1	0.629590	1
12	rgj truck trailer catches fire reno	1	0.627233	1
13	worldnews fallen powerlines glink tram update ...	1	0.616277	1
14	im fire	0	0.609812	0
15	metal cutting sparks brush fire brighton brush...	1	0.603601	1
16	good thing actually legit fire mall nobody eva...	1	0.602037	1
17	make sure evacuate past fire doors questions y...	0	0.587898	0
18	stores fire alarms went today work evacuate li...	1	0.586915	1
19	starts forest fire cannot put	1	0.584212	1

Figure 10. Top 20 disaster related tweets retrieved using the training dataset (KNN).

Table 3. Performance of ML models for top 20 disaster related retrieved tweets on training dataset.

Models	Labels	F1-score
Logistic Regression	1	0.85
Random Forest	1	0.79
KNN	1	0.93
Multinomial Naive Bayes	1	0.89
SVM	1	0.85
Gradient Boosting	1	0.84

After evaluating the ML models on the training dataset, the combined feature vector is used to predict and evaluate the testing dataset. This dataset is the actual testing dataset which is unlabeled. The predicted results were submitted to the Kaggle competition, and the public scores received for the selected machine learning models are shown in Table 4.

Table 4. Public scores of submitted predicted results in Kaggle competition.

Models	Public Score
SVM	0.79619
Multinomial Naive Bayes	0.79589
Logistic Regression	0.79374
Gradient Boosting	0.78854
Random Forest	0.78118
KNN	0.77781

6 Infrastructure

We have used the Google Colab virtual environment on personal computer to perform the programming task as hardware. We have used the Python programming language for writing the scripts for coding. Some of the libraries used to perform necessary tasks for the project are – Pandas to work with data frames, NumPy to work with arrays, NLTK corpus and modules for regular expression and string for text preprocessing purpose, Scikit-learn for computing TF-IDF scores, cosine similarity scores, generating classification report and working with machine learning models, Gensim for working with Word2Vec word embeddings. The word embedding took some time to load using the mentioned infrastructure. Apart from loading the word embedding model, the time and memory requirement were enough to perform the stated tasks.

7 Results and Discussion

Previous works show that while using traditional machine learning models, SVM performed best for the same task of predicting whether tweets are disaster related or not on the same dataset. Another experiment showed transformer model performed with a slightly better accuracy than multinomial naïve bayes for the same task. Our experiment on the training dataset resulted in multinomial naïve bayes being the one with highest F1-score for predicting both label 1 and 0. While incorporating the information retrieval part, KNN followed by multinomial naïve bayes resulted in higher F1-score. In summary, multinomial naïve bayes performed comparatively better for the classification problem of predicting disaster-related tweets and the incorporated information retrieval part as well.

As for the submitted predicted results in the Kaggle competition, the best results were achieved for SVM followed by multinomial naïve bayes. Therefore, comparing our experiment with earlier works on this dataset for the task of predicting whether the tweets are disaster related or not, multinomial naïve bayes seems to be a good choice compared to other traditional ML models. Again, earlier works involved using either TF-IDF score or word embeddings while we use a combination a both as feature vector to capture semantic relationships of words.

Future works that could be added to this project are performing experiments using other word embeddings in combination with the TF-IDF score to see if the performance improves. Current experiment is limited to performance comparison among traditional ML models only, so a performance comparison with different deep learning models could be done. Lastly, the query keyword is limited to unigrams only for the information retrieval task, thus, the query keyword could be extended to retrieve more relevant results.

Project materials are available here: https://github.com/Tarannum123/CS834_ResearchProject.git

8 References

[1] Jaekel,B. Twitter and Facebook serve different purposes for news consumption: Pew Research, https://www.marketingdive.com/ex/mobilemarketer/cms/news/research/20898.html

[2] Kaggle. Natural language processing with disaster tweets, https://www.kaggle.com/competitions/nlp-getting-started/overview

[3] MonkeyLearn, What is text analysis? A beginner's guide, https://monkeylearn.com/text-analysis/

[4] Bikku, T., Chandrika, P., Kanyadari, A., Prathima, V., & Sai, B. B. (2022). Analysis of Disaster Tweets Using Natural Language Processing. In *Intelligent Computing and Applications: Proceedings of ICDIC 2020* (pp. 491-501). Singapore: Springer Nature Singapore.

[5] Lakshmi, S. D., & Velmurugan, T. (2023). Classification of Disaster Tweets Using Natural Language Processing Pipeline. *Acta Scientific COMPUTER SCIENCES Volume*, *5*(3).

[6] Alvarado, B. J. S., & Solano, P. E. C. (2021). Detecting Disaster Tweets using a Natural Language Processing technique.

[7] Deb, S., & Chanda, A. K. (2022). Comparative analysis of contextual and context-free embeddings in disaster prediction from Twitter data. *Machine Learning with Applications*, *7*, 100253.

[8] Bhere, P., Upadhyay, A., Chaudhari, K., & Ghorpade, T. (2020). Classifying Informatory Tweets during Disaster Using Deep Learning. In *ITM Web of Conferences* (Vol. 32, p. 03025). EDP Sciences.

[9] Hyperskill. TF-IDF, https://hyperskill.org/learn/step/31428#

[10] Gomede, E. Word2Vec: Unlocking Semantic Power through Word Embeddings, https://medium.com/@evertongomede/word2vec-unlocking-semantic-power-through-word-embeddings-690df7e7b67d

[11] Geeksforgeeks. Understanding TF-IDF (Term Frequency-Inverse Document Frequency), https://www.geeksforgeeks.org/understanding-tf-idf-term-frequency-inverse-document-frequency/

[12] Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., & Brown, D. (2019). Text classification algorithms: A survey. *Information*, *10*(4), 150, https://doi.org/10.3390/info10040150