COSC522 - Twitter Disaster Classification

Nikhil Narayane (nnaraya2@vols.utk.edu)
Abdullah Salau (asalau@vols.utk.edu)

December 2023

1 Abstract

Twitter (or more recently known as 'X'), widely recognized as a vital real-time information hub, particularly in disaster scenarios, is the focal point of this study. The main objective of this project is to distinguish tweets related to disasters from those that are not, employing a range of sophisticated machine learning models and algorithms. The significance of this task cannot be overstated, especially for government bodies, public safety organizations, and emergency response teams, as it directly influences decision-making processes aimed at minimizing harm to both life and property.

In the realm of digital communication, Twitter has become an indispensable source for immediate news and critical updates during emergencies. This research leverages cutting-edge advancements in Natural Language Processing (NLP) to categorize the extensive corpus of Twitter data efficiently. The motivation for this study is rooted in the increasingly crucial role of Twitter as a primary source for urgent information in times of crisis. However, the endeavor is not without challenges, primarily due to the intricate and colloquial nature of language on social media platforms. This complexity often impedes the ability of machine learning models to accurately interpret the intent and relevance of tweets in the context of real-time disaster situations. Other challenges present include multilingual content and the ambiguous or lack of context underlying the tweets.

Despite these obstacles, our investigation tackles these challenges head-on, utilizing a carefully compiled curated dataset of 10,746 tweets for both training and testing purposes. The dataset primarily comprises textual data, offering limited auxiliary information for classification. Nevertheless, the remarkable advancements in NLP tools and techniques have significantly augmented our capability to extract meaningful features from text data. This report not only details the application of state-of-the-art NLP packages, such as SpaCy, Gensim, and HuggingFace, for generating insightful embeddings but also discusses their utilization as features in various classification models. The article provides a comprehensive analysis of different models employed, their respective accuracies, and the diverse methodologies adopted for vector generation, underscoring the pivotal role of feature extraction in this project.

2 Introduction

Social media platforms have evolved into indispensable tools for disseminating real-time information, playing a crucial role during emergencies and disasters. Twitter, as a dynamic microblogging platform, has emerged as a prominent source of immediate updates, providing a unique opportunity to harness its vast user-generated content for disaster identification and classification.

The surge in user interaction on social media, particularly Twitter, has prompted an exploration of its potential for disaster management and response. Initially designed for short messages, Twitter now supports 280-character tweets with multimedia, making it a key source for instant news and emergency updates. Recognizing this, our research delves into the innovative application of Natural Language Processing (NLP) and machine learning techniques to address the Twitter Disaster Classification problem.

The primary objective of this project is to develop a robust machine learning model capable of accurately classifying disaster-related tweets from the vast pool of social media data. The impact of achieving this objective is multifaceted. It significantly enhances the efficiency of emergency response teams, government bodies, and the public in differentiating critical information from regular content during crises. In addition to optimizing the usage of public goods and resources, by harnessing the power of machine learning for disaster identification on Twitter, we aim to contribute to the advancement of public safety and effective disaster management.

The motivation for undertaking this project is grounded in the growing dependence of disaster management and news agencies on Twitter as a primary source of immediate information during emergencies. As the digital landscape evolves, leveraging social media platforms for real-time disaster identification becomes increasingly imperative. The choice of this project is driven by the potential societal impact it holds, aligning with the critical need for efficient and accurate methods to sift through the massive volume of social media data generated during disasters.

As mentioned in the abstract, the main challenge inherent in this project lies in the complex and nuanced nature of language used on social media, where words and phrases can take on varied meanings based on context. Solving this problem, to a certain extent, contributes towards the development of a sophisticated machine learning model capable of discerning the intent and relevance of tweets in real-time disaster scenarios.

3 Preliminary Analysis

This report presents an analysis of a dataset comprising four primary attributes: a unique identifier, keyword, location, and tweet text, alongside a target variable that categorizes tweets into 'disaster' and 'non-disaster' classes. An initial examination of the dataset underscores the potential significance of the 'location' attribute in pinpointing disaster-prone regions. However, a notable challenge arises due to the frequent absence of specific location data. We propose methods to extrapolate location details from the tweet text, aiming to augment the predictive accuracy of our model.

Prior to delving into detailed research, a foundational layer is established through basic descriptive statistics. This groundwork steers the trajectory of our subsequent investigative efforts.

The training dataset encompasses 7613 entries, each featuring four distinct aspects: id, keyword, location, and text. The 'id' serves as a unique identifier, potentially functioning as a primary key. The 'keyword' is associated with the tweet content, ostensibly aiding in discerning whether a tweet is disaster-related. Among the dataset, there are 221 unique keywords.

A meticulous analysis reveals that the majority of keywords span across both disaster and non-disaster categories. The target variable bifurcates tweets into two groups: '1' for disaster-related and '0' for non-disaster-related tweets. The dataset comprises 3271 (43%) disaster-related and 4342 (57%) non-disaster-related tweets. This distribution, with a ratio of approximately 0.75 (43/57), suggests a moderately balanced dataset, conducive to analysis.

Focusing on the 'keyword' feature, we observe that 217 keywords are common to both categories. When juxtaposed with the total of 220 keywords in disaster tweets and 218 in non-disaster tweets, the high overlap diminishes the utility of 'keywords' as a discriminative feature. Consequently, we elect to exclude 'keyword' from our analysis.

Turning our attention to 'location', we encounter that 2533 out of 7613 entries (34%) are null. This significant proportion of missing data, coupled with the observation that tweet texts often lack explicit location mentions, challenges the usefulness of 'location' as a feature. A manual review of a random sample of 100 tweets further confirms the rarity of location-specific keywords.

Thus, we resolve to omit both ’location’ and ’keyword’ attributes from our analysis, focusing primarily on the ’text’ and ’target’ variables.

An investigation into the ’text’ aspect includes analyzing tweet lengths. While the range of lengths is consistent across both categories, with the inherent limitation of Twitter’s 280-character cap, there is no discernible pattern distinguishing disaster-related tweets from non-disaster ones in terms of length dispersion.

In conclusion, our analysis pivots on leveraging the ’text’ data. Recognizing the impracticality of using raw text in machine learning models, we extract features from the text, laying the groundwork for in-depth analysis and model development.

4 Key Terminologies in (NLP): A Primer for Beginners

Before delving into the methodology of Natural Language Processing (NLP), it’s beneficial to familiarize ourselves with some key terminologies. Understanding these concepts will lay the groundwork for a deeper comprehension of NLP techniques and applications.

Token: A token is essentially a building block of text, akin to a word in a sentence. In NLP, breaking down text into tokens helps in processing and analyzing it. For example, in the sentence ”Learning NLP is exciting”, the tokens would be [”Learning”, ”NLP”, ”is”, ”exciting”].

Corpus: This term refers to a large and structured set of texts. It’s used as a collection of data for NLP models. An example of a corpus could be a compilation of all articles published by a newspaper in a year.

Stopwords: These are commonly used words in a language that are often filtered out during data processing since they offer little to no unique information. Words like ”and”, ”the”, ”is”, and ”in” are typical stopwords. For instance, in the sentence ”The cat is on the mat”, the stopwords could be [”the”, ”is”, ”on”].

Embeddings: Embeddings are advanced representations of words and phrases in vectors. These vectors capture semantic meanings, enabling models to understand word associations and context. For example, in word embeddings, the words ”school” and ”education” would have similar vector representations due to their related meanings.

Importance of Embeddings: Embeddings are fundamental in NLP as they transform text into a format that’s understandable and usable by machine learning models. They enable models to grasp contextual nuances, making them essential for complex tasks like language translation or sentiment analysis.

Understanding these terms is crucial for anyone venturing into NLP, as they form the basis of many advanced processes and models in this dynamic field of study.

5 Experimental Setup and Methodology

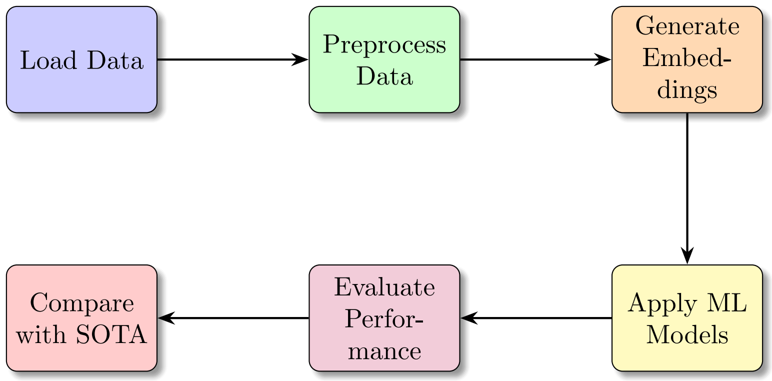

Figure 1: Concise Flowchart of the Twitter Disaster Classification Process

The schematic diagram previously presented delineates the experimental framework utilized in our study. In the preprocessing phase, we elected to exclude the ’keyword’ and ’location’ attributes from our dataset, focusing exclusively on the textual content for feature generation.

A pivotal step in our methodology involved the generation of text embeddings. Contrary to traditional approaches that necessitate the removal of stopwords before embedding, advancements in NLP packages now enable us to directly input raw text into the embedding algorithms. These sophisticated algorithms are adept at deciphering the overarching semantic context of sentences with a high degree of accuracy.

Our experimentation involved feeding the text data into various state-of-the-art embedding algorithms, specifically those provided by the spaCy and Gensim libraries. Each of these packages employs distinct methodologies for embedding generation, resulting in varying levels of efficacy in capturing semantic nuances—a core hypothesis of our study.

We then applied these embeddings as input features across multiple machine learning models, including Support Vector Machine (SVM), Logistic Regression, Gradient Boosting, Random Forest, and XGBoost, all of which are renowned for their classification capabilities.

Given the absence of target variables in the provided testing dataset, we partitioned our training dataset into training, validation, and testing subsets, adhering to a 70:15:15 split ratio. This approach enabled us to conduct a comprehensive evaluation of our models' performance.

Below, we present the results obtained from the various experiments conducted within this framework.

The following bar chart shows the performance of CountVectorizer for different algorithms:-

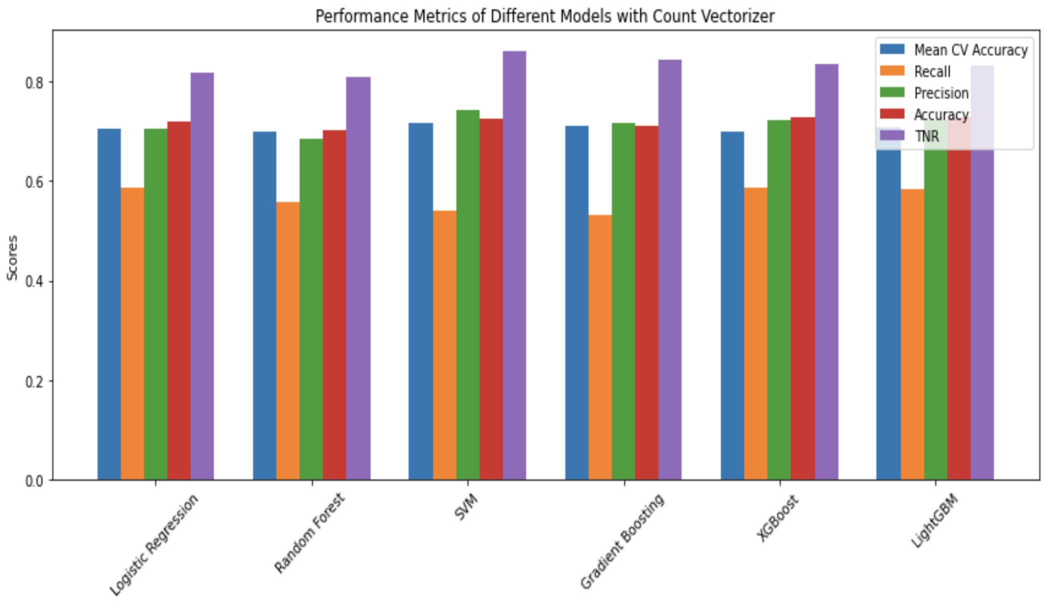

Figure 2: Bar Chart for CountVectorizer for different algoritms

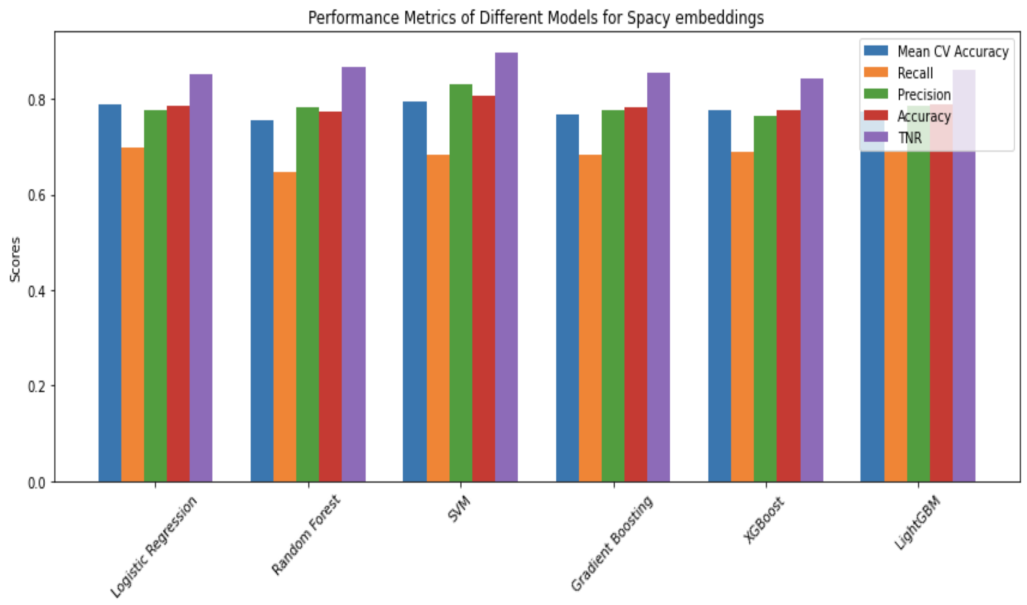

Figure 3: Bar Chart for Spacy embeddings for different algoritms

The embeddings we generated were of length (300,1). Such lengthy embeddings have a lot of information but can sometimes lead to overfitting. Also , processing such high dimensional data can be overwhelming for the model as well. So we tried to see the effect of dimension reduction. We used PCA to reduce the dimension. You wll find the all the results at the end of the table in the Appendix.

5.1 Comprehensive Analysis of Model Performances

- **Overall Base Model Efficacy:** SVM emerged as the top performer in base model settings, showcasing the highest accuracy and True Negative Rate (TNR). Logistic Regression also displayed robust performance across various metrics.
- **Enhancement through Hyperparameter Tuning:** The performance of models like Logistic Regression, Random Forest, and XGBoost significantly improved post-tuning, while SVM consistently maintained its high performance.
- **Count Vectorizer and PCA Integration:** When PCA was applied to the Count Vectorizer data, there was a notable variation in model performances. While the accuracy generally remained above 70%, the True Positive Rate (TPR) and True Negative Rate (TNR) varied, suggesting a nuanced impact of PCA on model efficiency.
- **Hyperparameter Tuning with Count Vectorizer:** The addition of hyperparameter tuning to Count Vectorizer models did not result in substantial performance improvements. In some cases, it even led to reduced model efficiency.

5.2 General Recommendations and Strategic Insights

Based on the analysis, SVM stands out as a consistently reliable model under various conditions, including the integration of PCA with Count Vectorizer. The application of dimension reduction techniques like PCA appears beneficial in refining feature representation, although its impact varies across different models. Careful hyperparameter tuning is essential, as its effectiveness is not uniform across all models and can lead to overfitting, especially in more complex scenarios.

This detailed evaluation provides a holistic view of the model performances with a specific focus on the implications of PCA on Count Vectorizer data, aiding in strategic model selection and optimization for efficient Twitter disaster classification.

6 Appendix A: Additional Results

Table 1: Base Model Performance-setup1

Model	Mean CV Accuracy	Recall	Precision	Accuracy	TNR
Logistic Regression	0.7890	0.6995	0.7761	0.7859	0.8501
Random Forest	0.7542	0.6471	0.7836	0.7735	0.8673
SVM	0.7957	0.6826	0.8296	0.8050	0.8959
Gradient Boosting	0.7681	0.6826	0.7772	0.7814	0.8547
XGBoost	0.7749	0.6903	0.7645	0.7774	0.8421
LightGBM	0.7744	0.6903	0.7859	0.7879	0.8604

Table 2: Table 1: Base Model Performance

Table 3: Hyperparameter Tuned Model Performance- Setup2

Model	Best Parameters	Recall	Precision	Accuracy	TNR
Logistic Regression	C: 0.01, penalty: l2	0.7042	0.7879	0.7932	0.8593
Random Forest	max_depth: 30, min_samples_leaf: 4	0.7119	0.7831	0.7932	0.8535
SVM	C: 1, gamma: scale	0.6826	0.8296	0.8050	0.8959
Gradient Boosting	learning_rate: 0.1, max_depth: 3	0.6872	0.7716	0.7800	0.8490
XGBoost	colsample_bytree: 1, learning_rate: 0.1	0.7119	0.7831	0.7932	0.8535
LightGBM	learning_rate: 0.1, max_depth: 7	0.6949	0.7803	0.7866	0.8547

Table 2: Hyperparameter Tuned Model Performance

Table 4: Count Vectorizer Model Performance-setup3

Model	Mean CV Accuracy	Recall	Precision	Accuracy	TNR
Logistic Regression	0.7061	0.5855	0.7063	0.7196	0.8192
Random Forest	0.6997	0.5578	0.6843	0.7019	0.8089
SVM	0.7158	0.5408	0.7421	0.7242	0.8604
Gradient Boosting	0.7112	0.5331	0.7164	0.7110	0.8432
XGBoost	0.6993	0.5855	0.7238	0.7282	0.8341
LightGBM	0.7082	0.5840	0.7219	0.7269	0.8330

Table 4: Hyperparameter Tuned Model Performance using Count Vectorizer

Table 5: Hyperparameter Tuned Model Performance using Count Vectorizer-setup4

Model	**Best Parameters**	**Recall**	**Precision**	**Accuracy**	**TNR**
Logistic Regression	C: 0.1, penalty: l2	0.5794	0.7002	0.7150	0.8158
Random Forest	max_depth: 30, min_samples_leaf: 1	0.5424	0.7537	0.7295	0.8684
SVM	C: 1, gamma: scale	0.5408	0.7421	0.7242	0.8604
Gradient Boosting	learning_rate: 0.1, max_depth: 3	0.5270	0.7185	0.7104	0.8467
XGBoost	colsample_bytree: 1, learning_rate: 0.1	0.5455	0.7284	0.7196	0.8489
LightGBM	learning_rate: 0.1, max_depth: 7	0.5501	0.7241	0.7189	0.8444

Table 6: Model Performance with Count Vectorizer and PCA (10% Error Rate)-final_setup

Model	**Mean CV Accuracy**	**Recall**	**Precision**	**Accuracy**	**TNR**
Logistic Regression	0.7025	0.5747	0.6895	0.7085	0.8078
Random Forest	0.7090	0.5131	0.7271	0.7104	0.8570
SVM	0.7190	0.5362	0.7484	0.7255	0.8661
Gradient Boosting	0.7067	0.5362	0.7342	0.7196	0.8558
XGBoost	0.6993	0.5794	0.7135	0.7216	0.8272
LightGBM	0.7203	0.5609	0.7208	0.7203	0.8387

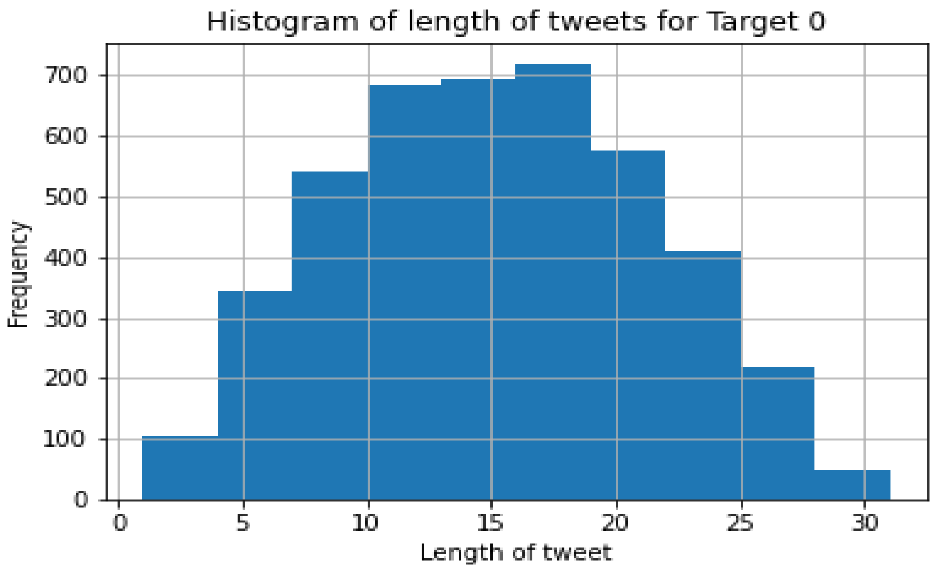

Figure 4: Target 0 tweet length

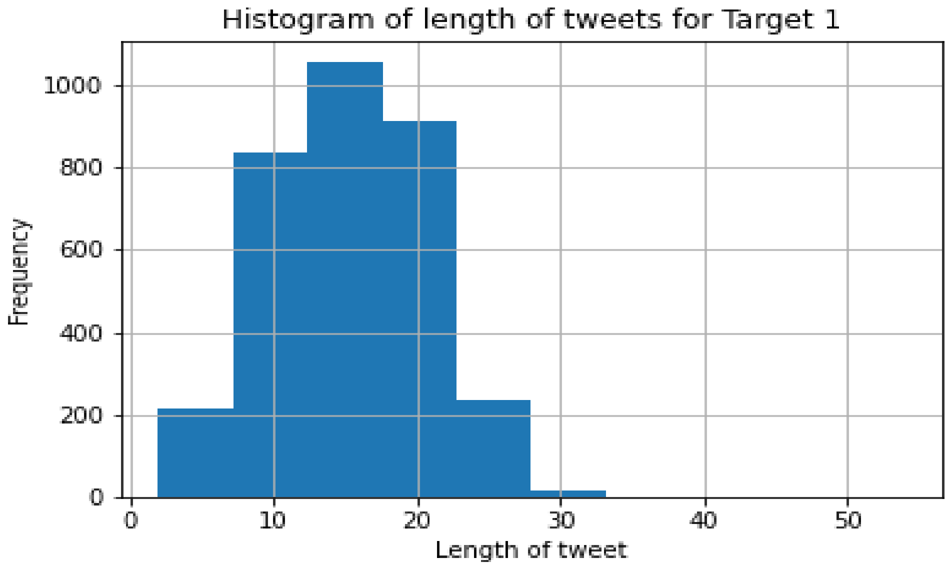

Figure 5: Target 0 tweet length

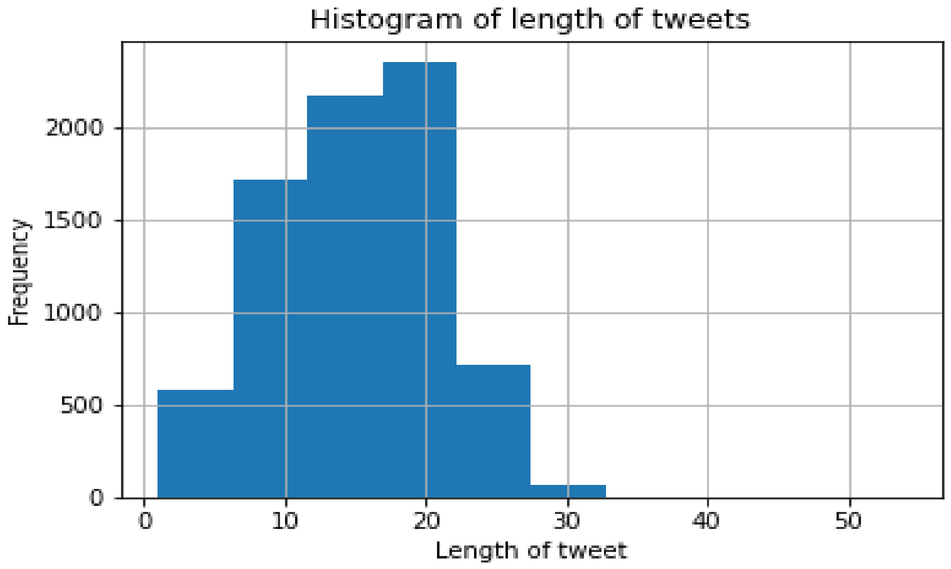

Figure 6: Complete tweet length