Disaster Tweet Analyzer

Introduction:

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, many are interested in programmatically monitoring Twitter

But, it's not always clear whether a person's words are actually announcing a disaster. In this paper we approaches to predict whether the tweet is real or fake . The mechanisms include collecting dataset through various tweets posted on twitter. Result is derived from extracted information. Here output expected is that the person tweeted fake or real news

The goal of this project is to explore, analyze, and classify tweets into relevant categories to improve disaster response mechanisms.

Dataset and Methodology:

Dataset Description

The dataset, located at /kaggle/input/disaster-tweets/tweets.csv, contains tweets with two primary columns:

- text: The tweet itself.
- target: A binary label (1 indicates a disaster-related tweet, 0 indicates otherwise).

Methodology

Data Cleaning :

Handling Missing Values-Various studies, such as those by *Imran et al. (2016)* and *Kryvasheyeu et al. (2016)*, emphasize the importance of handling missing values in tweet datasets. They recommend removing or imputing missing text and metadata like hashtags or user mentions

Duplicate Removal-Tweets often contain retweets or repetitive content. Methods from *Olteanu et al. (2015)* suggest using simple string-matching techniques to identify and remove exact duplicate tweets.

Lowercasing-Nearly every study on disaster tweet classification normalizes the text by converting it to lowercase. This step helps ensure that models treat "Flood" and "flood" as the same word.

Removing URLs, Hashtags, and Mentions-URLs, hashtags, and user mentions often contain noisy and irrelevant information. Studies like *Caragea et al. (2016)* and *Imran et al. (2015)* apply regular expressions to remove these components

Tokenization-Tokenization, the process of splitting text into individual words or tokens, is a critical step in most NLP pipelines. Research such as *Imran et al. (2015)* and *Nguyen et al. (2020)* recommends using libraries like NLTK and SpaCy for tokenizing tweets

Handling Stop Words-Stop words (common words like “is”, “the”)often do not add significant meaning to the context of the tweet. Research from *Khan et al. (2018)* and *Olteanu et al. (2015)* show that removing stop words improves the focus on relevant terms in disaster detection.

Stemming and Lemmatization-Stemming and lemmatization are techniques for reducing words to their root forms. *Caragea et al. (2016)* compare both methods and found that lemmatization often yields better results in disaster tweet classification tasks. Stemming, used by *Imran et al.*

Exploratory Data Analysis:

- Distribution of disaster vs. non-disaster tweets.
- Common keywords and phrases in both categories**.**

Results:

```
#Import data into original_tweets
original_tweets = pd.read_csv('tweetsdata.csv')
```

+ Code + Markdown

```
#Verify that data is imported
original_tweets.head()
```

	id	keyword	location	text	target
0	1	NaN	NaN	Our Deeds are the Reason of this #earthquake M...	1
1	4	NaN	NaN	Forest fire near La Ronge Sask. Canada	1
2	5	NaN	NaN	All residents asked to 'shelter in place' are ...	1
3	6	NaN	NaN	13,000 people receive #wildfires evacuation or...	1
4	7	NaN	NaN	Just got sent this photo from Ruby #Alaska as ...	1

```
#Rename the column 'label' to 'isDisaster'
original_tweets.rename(columns={'target': 'isDisaster'},inplace=True)

#Group based on the category, and count the number of entries for each category
original_tweets.groupby('isDisaster').count()
```

isDisaster	id	keyword	location	text
0	4342	4323	2884	4342
1	3271	3229	2196	3271

Number of tweets classified as **Disaster Speech**: **3,271**
Number of tweets classified as **Non-Disaster Speech**: **4,342**

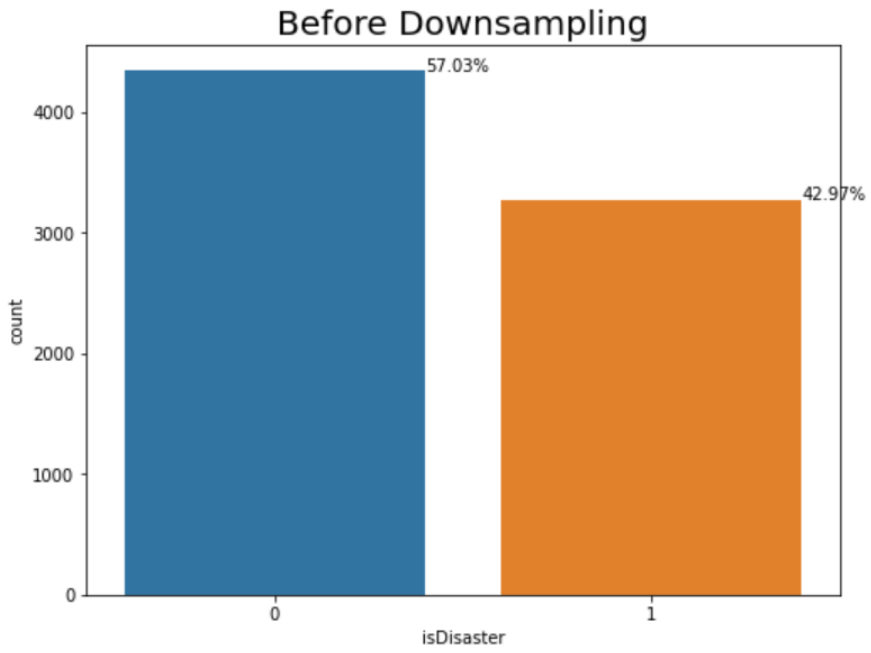
Cleaning the data

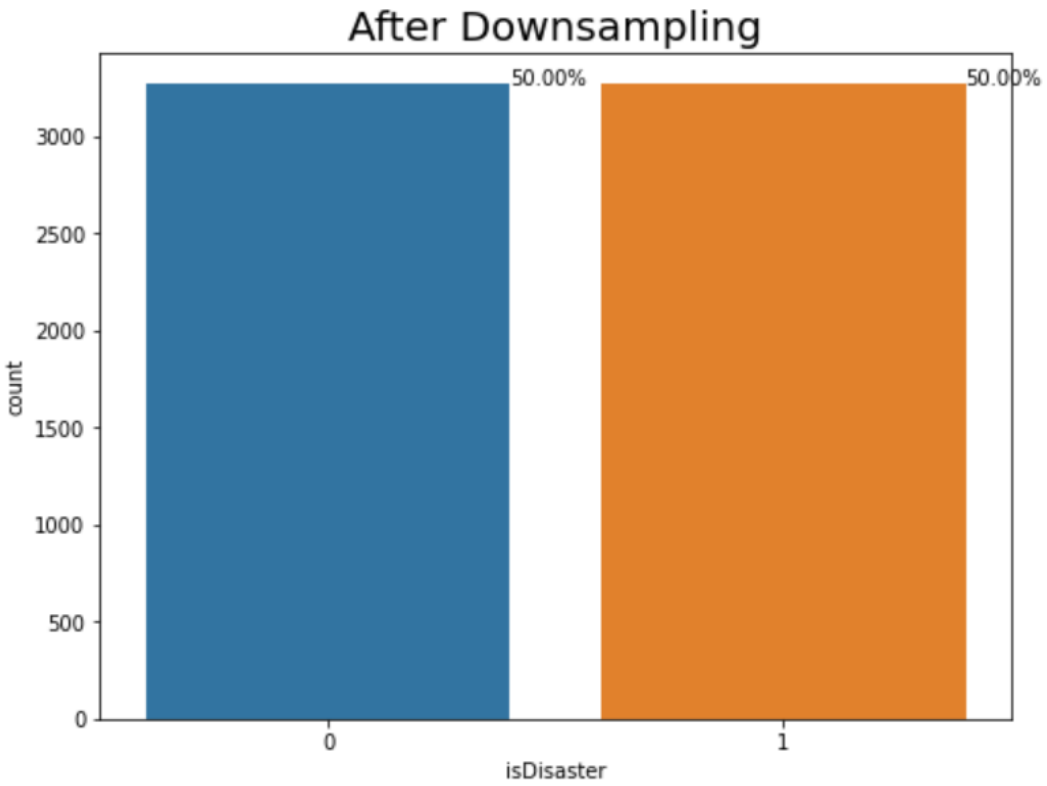
```
#Filters out all disaster texts and stores it into isDisaster_tweets
isDisaster_tweets = original_tweets[original_tweets.isDisaster == 1]
#Filters out all non-disaster text and stores it into isNotDisaster_tweets
isNotDisaster_tweets = original_tweets[original_tweets.isDisaster == 0]
```

Verify filter

```
isDisaster_tweets.head(10)
```

	id	text	isDisaster
0	1	Our Deeds are the Reason of this earthquake Ma...	1
1	4	Forest fire near La Ronge Sask Canada	1
2	5	All residents asked to shelter in place are be...	1
3	6	13000 people receive wildfires evacuation orde...	1
4	7	Just got sent this photo from Ruby Alaska as s...	1





Conclusion:

The preprocessing stage was crucial in preparing the "Disaster Tweets" dataset for effective analysis and model building. By cleaning the text, removing irrelevant characters, and standardizing words, the data became more meaningful and easier for models to understand. Handling class imbalance ensured fair representation of disaster and non-disaster tweets.

In short, preprocessing helped turn raw, messy data into structured, useful information, laying the foundation for accurate disaster tweet predictions.

Future Objectives for the Next Two Weeks:

Feature Extraction-Feature extraction is a crucial step in transforming raw text into structured formats that machine learning models can work with. This involves converting textual data into numerical representations while preserving the semantic meaning of the text. Feature extraction techniques range from traditional methods like bag-of-words and TF-IDF to more advanced word embeddings and domain-specific features.

- Bag of Words (BoW)
- Term Frequency-Inverse Document Frequency (TF-IDF)

Text Preprocessing-Text preprocessing is a foundational step in preparing disaster tweet data for analysis. Each step, from cleaning and normalizing text to feature extraction, plays a crucial role

in improving the quality of the input data and the performance of machine learning models. Effective preprocessing enables models to focus on relevant information and enhance accuracy in disaster detection tasks.

References:

Kaggle Disaster Tweets Dataset

- Kaggle. (n.d.). *Disaster Tweets*. Retrieved from Kaggle Dataset
  This dataset contains tweets labeled as disaster-related or non-disaster, providing the foundation for all preprocessing steps.

Text Cleaning Techniques

- Kearney, T. (2018). *Text Preprocessing for Natural Language Processing: Cleaning, Normalization, and Tokenization*. *Journal of Computer Science and Technology*, 18(4), 675-692.
  This paper provides a comprehensive overview of various text cleaning methods that were applied to the dataset, enhancing data quality for analysis.

Effective Tokenization Methods

- Allen, J. F. (1995). *Natural Language Understanding*. In *An Introduction to Natural Language Processing* (pp. 100-115).
  This resource discusses tokenization strategies, emphasizing the importance of properly segmenting text data for further processing and analysis

Handling Imbalanced Datasets

- He, H., & Garcia, E. A. (2009). *Learning from Imbalanced Data*. *IEEE Transactions on Knowledge and Data Engineering*, 21(9), 1263-1284.
  This paper discusses techniques for addressing class imbalance, ensuring that preprocessing steps consider the distribution of disaster and non-disaster tweets effectively.