Classification of Disaster Tweets

College of Engineering
TENNESSEE TECH

Sharon Colson, Thomas Robertson, Caleb Smith, Tania Perdomo Flores

Introduction

Social media platforms like Twitter provide real-time information during disasters. However, identifying authentic disaster reports among metaphorical or unrelated content is a challenge. This study presents a classification model designed to accurately and efficiently detect disaster-related tweets. The goal is to support emergency response teams by filtering relevant information in real time.

Exploratory Data Analysis

The dataset, sourced from a Kaggle prediction competition, consisted of tweets labeled as disaster-related (43%) or non-disaster (57%). Each tweet included text, a user-provided location, and an optional keyword, with the target variable indicating the presence of a real disaster.

Data Cleaning and Preprocessing -

- **Normalized text** by correcting corrupted character sequences, removing accents, and filtering noise (e.g., URLs, extraneous symbols).
- **Removed unnecessary columns** (e.g., *id, location*), eliminated duplicates and conflicting records, and handled missing values to ensure data consistency.
- **Created 30 dataset variants** by applying three different treatments to the *keyword* feature (prepended, removed, or separate), each combined with ten different preprocessing configurations (e.g., tokenization, stopword removal, stemming, lemmatization). All variants were **vectorized using TF-IDF** with bi-grams and tri-grams.

Disaster-related tweets averaged 101 words and were generally more descriptive, frequently including terms such as *fire*, *disaster*, and *California*, along with increased emoji usage to convey urgency or emotion. After preprocessing, the data was split 80% training and 20% testing sets. These were used to evaluate initial baseline models and guide the deployment of more advanced transformer-based approaches.

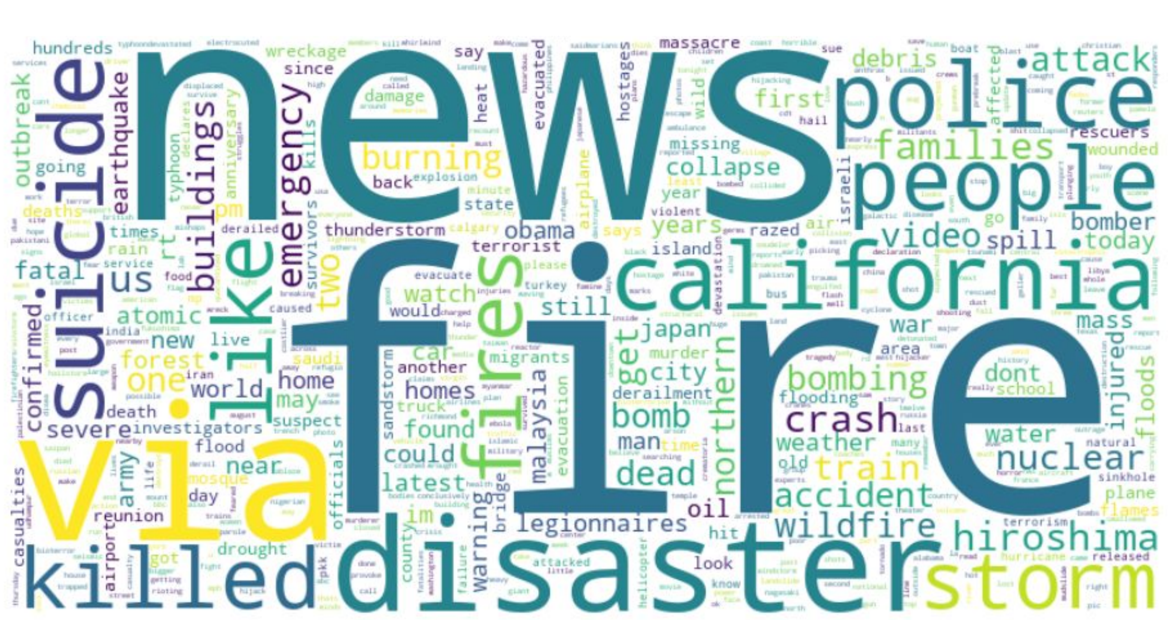

Figure 1: Target Disaster Words Represented on Wordcloud

Methodology

After preprocessing, the data was processed through a standardized pipeline and evaluated using **5-fold cross-validation** to ensure robust performance estimation. For baseline comparison, we implemented a suite of classifiers:

- **Multinomial Naive Bayes (MNB):** Applied probabilistic reasoning well-suited to high-dimensional sparse data.
- **Logistic Regression**: Offered an interpretable linear model for estimating class membership probabilities.
- **Support Vector Machine (SVM)**: Used a linear kernel to maximize the decision margin between classes.
- **Passive Aggressive Classifier**: Dynamically updated its decision boundary with incoming batches of data, making it ideal for streaming scenarios.
- **K-Nearest Neighbors (KNN)**: Employed a non-parametric approach; performance can degrade with large datasets due to its computational intensity.
- **Multi-Layer Perceptron (MLP)**: Featured a single hidden layer with 50 neurons to capture non-linear patterns in text data.

Metric	Multinomial Naive Bayes	Passive Aggressive Classifier	Logistic Regression	Support-Vector Machine	K-Nearest Neighbors	MLP Classifier (NN)
Accuracy	0.7969	0.7846	0.7864	0.6110	0.6849	0.7951
Precision	0.8607	0.7609	0.7609	1.000(?)	0.8775	0.7903
Recall	0.6304	0.7288	0.5885	0.0964	0.3340	0.7136
F1 Score	0.7276	0.7443	0.7034	0.1757	0.4458	0.7499
ROC AUC	0.8481	0.8399	0.8465	0.8524	0.7383	0.8463

Table 1: Baseline Model Results

Each model's performance was evaluated using five key metrics: **Accuracy, Precision, Recall, F1 Score,** and **ROC AUC** metrics.

Best Model (Post Grid Search)

The **Passive Aggressive Classifier** achieved the highest F1 Score among all hyperparameter-tuned models when applied to the *kept_v7_lowercase_words_only* dataset. The preprocessing for this variation involved converting all text to lowercase and restricting content strictly to word tokens. This included removing numerical digits, punctuation, special characters, emojis, and mentions - resulting in clean, consistent input for modeling.

Dataset	Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
kept_v7_lowercase_word_only	Passive Aggressive	0.793	0.765	0.734	0.752	0.859
kept_v2_no_emojis_mentions	Passive Aggressive	0.793	0.765	0.739	0.752	0.856
kept_v9_minimal_processing	Passive Aggressive	0.792	0.764	0.739	0.751	0.861
kept_v1_basic_clean	Passive Aggressive	0.791	0.762	0.737	0.750	0.860
kept_v6_custom_stopwords	Passive Aggressive	0.789	0.759	0.741	0.750	0.864

Table 2: Passive Aggressive Classifier Results with Top Five of Dataset Variations

With an accuracy of 79.26%, the model shows strong overall performance in classifying disaster and non-disaster tweets post-tuning. A precision of 76.50% reflects reliable disaster tweet predictions, while a 73.92% recall highlights improved sensitivity—showcasing the impact of hyperparameter tuning. The 75.18% F1 score strikes a solid balance between precision and recall, confirming it as the best-performing model. A ROC AUC of 85.88% further indicates strong discriminative power and overall robustness.

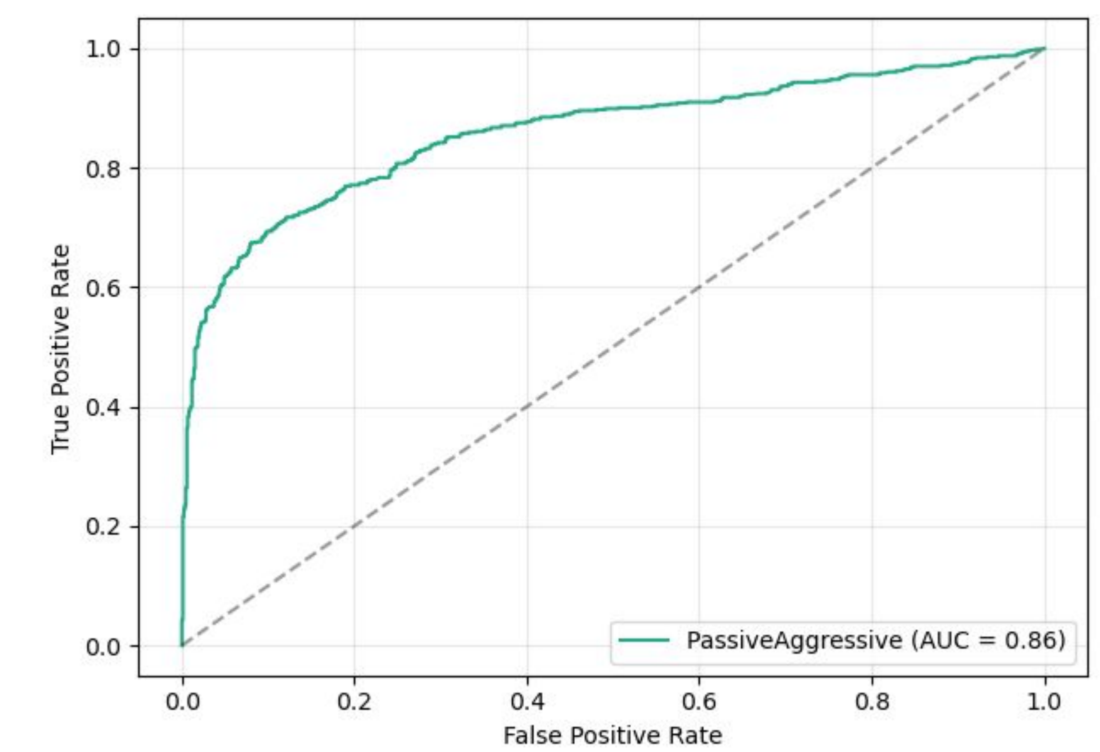

Figure 2: Passive Aggressive Classifier ROC AUC Curve

Tuning key hyperparameters improved generalization, while simple preprocessing outperformed complex methods.

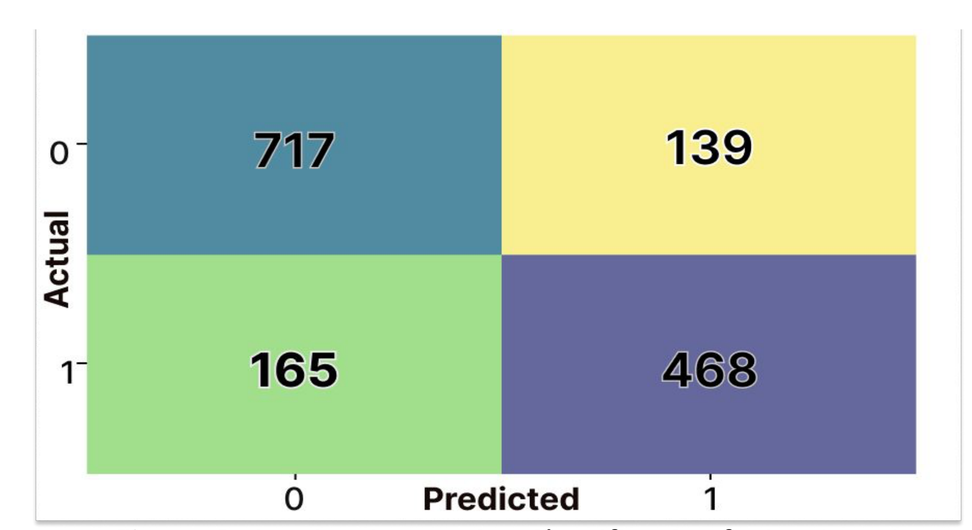

Figure 3: Passive Aggressive Classifier Confusion Matrix

Trained Models vs BERT

We fine-tuned **BERT-Base-Uncased** and **BERTweet-Base** on the top-performing datasets (*kept_v7*, *kept_v2*, *kept_v9*). Both pretrained models outperformed our custom classifiers across all metrics. **BERTweet** led in most cases, achieving up to **0.08 higher precision** than the Passive Aggressive Classifier on *kept_v9*, with only modest gains in F1 and accuracy (~0.05–0.06).
While the performance lift was consistent, the **computational cost** of BERT models raises practical concerns. The Passive Aggressive Classifier offers a **faster, more efficient alternative** with only a small trade-off in accuracy.

Results

We evaluated six classifiers using cross-validation and hyperparameter-tuning via GridSearchCV. The **Passive Aggressive Classifier** emerged as the top performer on the *kept_v7_lowercase_words_only* dataset, achieving **79.26% accuracy**, **76.50% precision**, **73.92% recall**, an **F1 score of 75.18%**, and a **ROC AUC of 85.88%**. While all models improved over their baseline counterparts, the **MLPClassifier** required significantly **more computational resources** without proportional performance gains. Models like **Multinomial Naive Bayes** and **K-Nearest Neighbors** also **underperformed** in recall, increasing the risk of missed disaster detections.

Discussion - Model Tuning

Grid search notably improved **Logistic Regression**, **SVM**, and **Neural Network** models, enhancing F1 scores and generalization. Simpler preprocessing consistently outperformed more complex pipelines—likely due to the brevity and informality of tweet data. **MLPClassifier Neural Network** consumed the overwhelming majority of computation resources with lower average payoff. Even if the neural network had performed marginally better, the benefits would not have outweighed the costs.

Conclusions

This project demonstrated the effectiveness of **systematic tuning and preprocessing** for disaster tweet classification. The Passive Aggressive Classifier proved reliable, interpretable, and resource-efficient.These results suggest that simpler models, when carefully optimized, can offer strong real-world performance.

Future Work

Future work will explore **model ensembling, advanced embeddings** (e.g., Word2Vec, FastText), and more scalable tuning techniques like **RandomizedSearchCV** or **Bayesian Optimization** to further improve recall and reduce training time.

References

Our dataset was collected from kaggle.com
https://www.kaggle.com/competitions/nlp-getting-started/data
Our visualizations and results were collected from our Github repository:
https://github.com/CSC-4260-Advanced-Data-Science-Project/NLP_Disaster_Tweets

Acknowledgements

We gratefully acknowledge **Tennessee Tech ITS Services and the *Warp 1* HPC Facility** (NSF Award #2127188) for its use in training and evaluation of the multitude of models utilized in this project. Special thanks to Dr. William Eberle for his guidance in CSC 4260.

Acknowledgments: We acknowledge that this project was completed for CSC-4260/5260-001 at Tennessee Technological University.