Catastrophe Posts Genuinity Prediction

Shriya Sandilya

October 25, 2025

Problem Statement Understanding

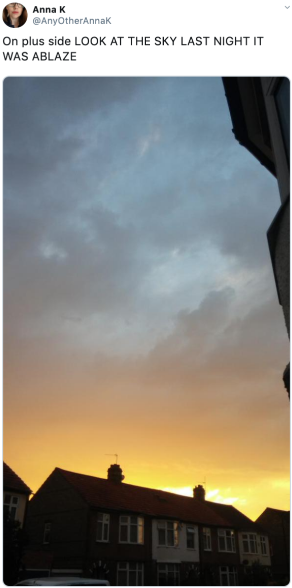
Problem Statement

Catastrophe posts on social media
Can be genuinely about catastrophes like

- Landslide
- Earthquake
- Fire etc.

Or they can be exaggerations and hyperboles to express some other sentiment

Example Posts



Dataset

- 10,000 posts
- Hand-Classified as Genuine or Fake

id	text	location	keyword	target
a unique identifier for each post	the text of the post	the location the post was sent from (may be blank)	a particular keyword from the post (may be blank)	denotes whether a post is about a real disaster (1) or not (0)

Table: Dataset Columns

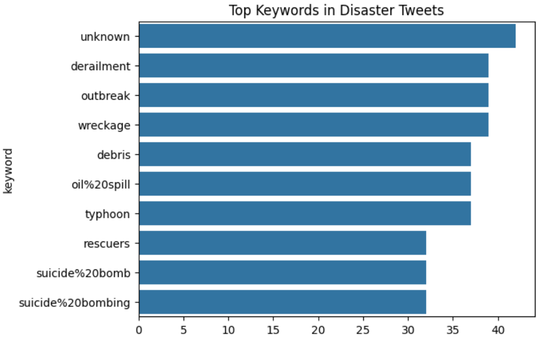
Dataset Analysis

Null Values

column	percentage
id	0.00
text	0.00
location	33.27
keyword	0.80
target	0.00

Class Distribution

class	count	percentage
0	4342	57.03
1	3271	42.97



Approach and Methodology

Tech Stack Used

Languages & Libraries

- Python
- NumPy, Pandas, Scikit-learn
- NLTK, Emoji
- Streamlit for deployment

Tools & Environment

- Jupyter Notebook for development
- Overleaf for presentation
- Git + GitHub for version control
- Joblib for model persistence

Approach and Methodology

1. **Data Preprocessing**
   - Removed URLs, mentions, hashtags, emojis.
   - Lowercased text, removed stopwords, applied stemming.
2. **Feature Engineering**
   - TF-IDF vectorization of cleaned text (max_features=8000, bigrams included).
   - Added 3 numeric features:
     - Character count
     - Word count
     - Punctuation count
3. **Feature Combination**
   - Combined sparse TF-IDF and scaled numeric features using `scipy.hstack()`.
4. **Model Training**
   - Logistic Regression with 5-fold cross-validation.
   - Tuned using GridSearchCV for regularization strength. (Not used in end)
5. **Evaluation Metrics**
   - Accuracy, F1-score.

Model Selection

Compared 4 NLP Focused ML Models

- Logistic Regression
- Naive Bayes
- Random Forest
- XGBoost
- SVM

Compared their Accuracy and F1 Scores

Model Comparison:

	Model	Accuracy	F1-Score
4	SVM	0.820092	0.775410
0	Logistic Regression	0.821405	0.774461
1	Naive Bayes	0.815496	0.756288
2	Random Forest	0.798424	0.752220
3	XGBoost	0.787262	0.717277

Model Selection

Tuned Hyperparameters

- Logistic Regression
  - Best params:
    - solver: lbfgs
    - C: 2
  - Evaluated with tuned parameters
    - Worse Performance
    - Accuracy:
    - F1 Score:
- XGBoost
  - Best params:
    - subsample: 0.8
    - n estimators: 400
    - max depth: 4
    - learning rate: 0.1
    - colsample by tree: 0.8

Model Selection

Ensemble Models

Soft Voting Models

Hyperparameter tuned Logistic Regression and XGBoost

- Logistic Regression + Naive Bayes + Random Forest + XGBoost
- Logistic Regression + Naive Bayes
- Logistic Regression + XGBoost

Compared their Accuracy and F1 Scores

Model	Accuracy	F1
LR + NB + RF + XG	0.8221	0.7662
LR + NB	0.8162	0.7701
LR + NB	0.8050	0.7607

Nothing exceeded **Logistic Regression** or **SVM** significantly

Went ahead with Logistic Regression

Project Demo

Project Demo

- Deployed using **Streamlit**.
- Accepts tweet input from user.
- Cleans, vectorizes, and predicts using the trained model.

Example:

Input: "Earthquake in Delhi, buildings shaking!"
Output: Real Disaster Tweet (Confidence: 0.71)

Challenges and Learnings

Challenges

- Figuring out what can be visualised
- Figuring out numeric features
- Trying to improve Accuracy and F1 Score
- Data Augmentation - Could not accomplish properly
- Trying DL techniques
- Feature fit issues in deployment

Learnings

- Practical usage of NLP tools like TF-IDF and text cleaning function
- Learned how to use Ensemble Models

References and Links

- References
  - Kaggle NLP Getting Started
  - NLTK Documentation
  - Scikit-Learn: TF-IDF
  - Scikit-Learn: Logistic Regression
  - Streamlit Deployment
- Links
  - Deployed Streamlit App
  - Github Repository

Thank You