The dataset for this project can be found here.

Project Submission: 8th Dec, 2024

NLP Project for Disaster Tweet Classification

Background:

In today&#39;s digital age, social media platforms like Twitter play a crucial role in disseminating
information, especially during emergencies and disasters. However, distinguishing between tweets that genuinely indicate a disaster and those that do not can be challenging, even for humans. As a result, there is growing interest from various organizations, such as disaster relief agencies and news outlets, in developing automated methods to monitor Twitter for real-time disaster alerts.

Challenge:

The challenge is to build a machine learning model capable of accurately classifying tweets as either related to real disasters or not. While some tweets may contain explicit keywords like &quot;fire&quot; or &quot;flood,&quot; others may use such terms metaphorically or in a non-disaster context.

Thus, the model needs to discern the true intent behind the language used in tweets, which can be nuanced and context-dependent.

Objectives:

Classification Model Development: Develop a machine learning model that can predict whether a tweet is about a real disaster or not.

Accuracy and Precision: Aim for high accuracy and precision in classification to minimize false alarms and ensure timely and accurate disaster detection.

Robustness: Create a model that can handle variations in language, including slang, abbreviations, and cultural nuances, to ensure effectiveness across diverse tweet datasets.

Scalability: Build a scalable solution capable of processing large volumes of tweets in real-time to support continuous monitoring efforts.

Key Tasks:

Part 1: Data Exploration and Preparation

Task: Data Exploration

How to Achieve:

- Load the dataset of 10,000 hand-classified tweets.
- Explore the dataset&#39;s structure using Python libraries like Pandas to understand the columns and data types.
- Visualize the distribution of classes (disaster vs. non-disaster tweets) using histograms or bar plots.
- Analyze the frequency of keywords and phrases associated with disaster tweets.

Task: Data Preparation

How to Achieve:

- Clean the text data by removing special characters, URLs, and punctuation marks.
- Tokenize the text into individual words or tokens.
- Convert text labels into numerical format (e.g., 0 for non-disaster, 1 for disaster).
- Split the dataset into training and testing sets for model development and evaluation.

Part 2: Feature Engineering and Model Selection

Task: Feature Engineering

How to Achieve:

- Extract relevant features from the text data, such as word frequencies, TF-IDF scores, and sentiment analysis.
- Consider using pre-trained word embeddings like Word2Vec or GloVe to capture semantic meanings.
- Experiment with additional features like tweet length, presence of hashtags, or user mentions.

Task: Model Selection and Training

How to Achieve:

- Choose a set of candidate classification models suitable for text classification, such as logistic regression, random forests, or neural networks.
- Train each model using the training data and evaluate their performance using cross-

validation techniques.
- Optimize hyperparameters of the selected models using techniques like grid search or random search.

Part 3: Model Evaluation and Validation

Task: Model Evaluation

How to Achieve:

- Evaluate the trained models using appropriate evaluation metrics for binary classification tasks, such as accuracy, precision, recall, and F1-score.
- Visualize the performance metrics using confusion matrices, ROC curves, and precision-recall curves.
- Compare the performance of different models to select the best-performing one for deployment.

Task: Model Validation

How to Achieve:

- Validate the selected model on the testing dataset to assess its generalization ability.
- Check for potential issues like overfitting or underfitting and adjust the model if necessary.
- Ensure that the model performs well on unseen data and is robust to variations in the tweet text.

Part 4: Deployment with Web Interface

Task: Model Deployment

How to Achieve:

- Serialize the trained model into a format suitable for deployment, such as a pickle file.
- Develop a web application using a framework like Flask or Django to create the user interface.
- Integrate the trained model into the web application backend to handle incoming tweet data and make predictions.

- Deploy the web application on a server or cloud platform like Heroku or AWS.

Task: User Interface Design

How to Achieve:
- Design a user-friendly interface where users can input text or paste tweet content.
- Provide clear instructions and feedback on the classification result.
- Add additional features like visualizations or real-time updates if needed.

Conclusion:
By successfully developing and deploying a machine learning model for classifying disaster-related tweets, we can enhance the ability to detect and respond to emergencies more efficiently. This project represents a valuable opportunity to leverage NLP techniques for the greater good, contributing to the advancement of disaster management and public safety efforts.

Focus on ensuring the model interprets sarcasm and metaphor in tweets accurately. some noise.