Comp 562 Final Project Report

Anita Murali, Srividya Ramesh, Soumya Mahavadi, Ximena Colopy

I. Introduction

A. Problem

Social media platforms such as Twitter have become an important mode of communication in times of emergency due to smartphones. Monitoring the rapid dissemination of information on these platforms is of great interest to authorities, disaster relief organizations, and news agencies among others. On Twitter, this involves sifting through numerous tweets related to ongoing disasters. During such events, it becomes crucial to differentiate between tweets that are actual reports of real disasters and those that are not.

B. Objective

The objective of this project is to develop a machine learning model that can classify a Twitter tweet as being about a real disaster or not, based on its content. This project will look at 3 different approaches to accomplishing this task. We will also evaluate the accuracy at which these models perform and provide insights into how reliable these models can be for real-world applications like disaster response.

C. Motivation

During emergencies and disasters, the accuracy of information shared on social media can significantly impact response and rescue efforts. False reports can lead to unnecessary panic, misallocation of resources, or even harm. Accurate identification of real disaster events from tweets can enhance situational awareness and aid in efficient response and resource deployment by authorities.

Manual classification of tweets during high-volume periods, such as during a disaster, is impractical due to the sheer speed and volume of data. Automating this process can improve and quicken real-time processing of data, which is vital for timely and effective decision-making during a crisis.

This project also examines how different models and approaches learn to handle the nuanced language like slang and abbreviations found in Twitter tweets.

D. The Dataset

The dataset we will be using for this project is a collection of 10,000 Twitter tweets that have been classified by hand and posted on Kaggle. This dataset was created by the company Figure-Eight and originally shared on their 'Data For Everyone' website.

II. Related Work

The utilization of social media data, particularly Twitter, for disaster response has gained significant attention in recent research due to its potential to provide real-time insights and assist in emergency situations. A variety of approaches and methodologies have been explored to effectively classify and use Twitter data during disasters.

Raguram et al. [4] (2023) developed an enhanced machine learning framework to classify disaster-related tweets by employing sophisticated algorithms that aim to improve the responsiveness of disaster management teams. Their work emphasizes the utility of supervised learning techniques to accurately categorize tweets as disaster-related or not, underscoring the challenges of managing voluminous unstructured data generated during emergencies.

Kumar and Singh [2] [1] (2024) explored the real-time classification of disaster-related tweets by implementing machine learning models that address the challenges posed by the limited context and character constraints of tweets. Their research also tested the efficiency of different machine learning algorithms, including a comparison with the BERT model, highlighting the complexity of processing and understanding sparse textual data effectively.

Goswami and Raychaudhuri [3] (2020) focused on identifying disaster-related tweets using natural language processing to analyze the unstructured format of Twitter data. Their study utilized decision tree classifiers to differentiate between disaster-related and unrelated tweets, illustrating the effectiveness of NLP techniques in extracting meaningful information from social media during crises.

Together, these studies demonstrate a growing trend toward leveraging advanced computational techniques and machine learning models to enhance the accuracy and timeliness of social media analytics in disaster situations. These efforts are critical in harnessing the full potential of real-time data for improving emergency response and preparedness.

III. Approaches

A. BERT base model (uncased)

1) Background: Google's BERT is a transformers model that was pre-trained on a large corpus of English data in a self-supervised manner. This approach involves the raw texts, with no human labeling, and an automatic process to generate inputs and labels from those texts. Here, we are using the base (uncased) version, which does not differentiate between uppercase and lowercase letters.

2) Model Architecture: BERT is based off the Transformers Language Model, a deep learning model, which is currently considered state-of-the-art in the field of machine learning. In this model, every output element is connected to every input element. The key component of the transformers language model is the self-attention mechanism, which assigns greater importance to specific input words over others based on relevance. This enables BERT and other Transformer-based models to deepen their understanding of context.

3) Training Process and Fine-tuning: Initially, the text dataset is loaded and split into training, validation, and test sets. The words are then encoded using a BERT tokenizer from the bert-base-uncased model, which converts text into a format suitable for input into the BERT model. The encoded texts are transformed into TensorFlow datasets for the training and validation sets. These datasets are used to feed data to the pre-trained BERT model (TFBertForSequenceClassification) for fine-tuning. The model is compiled with an Adam optimizer and a cross-entropy loss function. Training is done over three epochs, where the model learns from the training data while periodically being validated on a separate validation set to monitor its performance and generalize its ability beyond the training data.

4) Performance Evaluation: Performance of the model was based off five metrics: F1 Score, Accuracy Score, Confusion Matrix, AUC-ROC curve, and Precision-Recall Curve.

5) Advantages & Limitations: One of the biggest advantages of the BERT model is its ability to understand the context in a sentence. It does this by looking at the entire sequence of words at once, unlike other previous models. Additionally, the model is easily adaptable to a variety of NLP tasks with efficient results and requires less fine-tuning on specific tasks as it is a pre-trained model.

A limitation of the BERT model is that in very small datasets, it tends to overfit to the training data, and thus, fails to generalize. Also, due to the size and complexity of BERT, it can be difficult to deploy on devices with limited resources. In response, the DistilBERT model was created to address these challenges.

B. Naive-Bayes Classification

1) Background: Naive-Bayes is a simple but powerful supervised machine learning algorithm used for classification. This model assumes features are mutually independent given the class. There are three types of Naive-Bayes classifiers. The Gaussian classifier assumes that features follow a normal distribution, Bernoulli assumes features are binary-valued, and Multinomial is an extension of Bernoulli where features represent frequency counts. This project implements the Multinomial Naive-Bayes classifier because it is the most suitable for word classification (Kavlakoglu, 2024).

2) Preprocessing and Implementation: The provided text files are initially split into training, validation, and test sets. Then, the respective feature datasets are preprocessed and cleaned. First, each string or line of text is broken down into word tokens by means of the word_tokenize() function from the Natural Language Toolkit (NLTK) library. Then, tokens that are recognized as punctuation marks or stopwords—common filler words that usually contribute no significant meaning to the actual text—are removed. Finally, the remaining tokens are joined back into a single string, which represents a cleaned version of the original string.

The cleaned feature datasets are then vectorized into document-term matrices (DTMs), which describe the frequency of word tokens occurring in a collection of text documents. Once this process is done, the Multinomial Naive-Bayes classifier is trained using the DTM training data and its corresponding target labels in order to predict probabilities on the validation and testing sets.

3) Performance Evaluation: Performance of the Multinomial Naive-Bayes classifier was evaluated based on five metrics: the model earned an F1 score of 0.7416, an accuracy score of 0.7873, a ROC-AUC score of 0.7782, and an average Precision-Recall score of 0.6715.

4) Advantages & Limitations: Advantages:

Naive-Bayes is computationally efficient and simple; this makes it very easy to implement. Due to this, Naive-Bayes runs very quickly, so it is ideal for larger datasets. Since the model computes probabilities independently among features, it handles missing data values well. Furthermore, Naive-Bayes does very well with categorical data, which makes it incredibly popular for text classification.

Limitations:

Naive-Bayes assumes strong independence between features. In the real world features are rarely independent. This can lead to inaccuracies if there are strong dependencies between features. Additionally, if the dataset isn't diverse enough to represent real-world distributions, there will likely be inaccuracy in the predictions.

C. Logistic Regression

1) Background: Logistic Regression is a statistical method for binary classification that can be extended for multinomial classification (Jurafsky, 2024). Unlike linear regression which is used to predict a continuous dependent variable, logistical regression predicts the probability of a binary event occurring.

Logistic regression is a discriminative classifier, while Naive-Bayes is a generative classifier. A discriminative model, as opposed to a generative model, is only trying to distinguish the different classes, without learning much about them (Jurafsky, 2024). For example, if you give a discriminative model pictures of cats and dogs and ask it to classify each animal's picture, it would find one difference between the two animals and be satisfied. If all of the cats were wearing collars but none of the dogs were wearing collars in the training set, it would mark all animals wearing collars in the test set as cats.

Using a threshold probability (0.5 for our model), the model will assign a boolean to each data entry.

2) Training Process & Validation: Logistic regression is a probabilistic classifier that makes use of supervised machine learning. It has four components: A feature representation of the input, a classification function that computes the estimated class, an objective function for learning, and an algorithm for optimizing the objective function (Jurafsky, 2024). The data is split into a training set and testing set, using the traintestsplit function in sklearn. The data was first vectorized to change the tweets from a string format into one that is easier for the model to work with. A threshold of 0.5 was used for the model.

3) Performance Evaluation: This Logistic Regression model used here uses five metrics to evaluate its performance: F1 Score, Accuracy Score, Confusion Matrix, AUC-ROC curve, and a Precision-Recall Curve. It received an F1 score of 0.7531, Accuracy score of 0.8050, ROC-AUC score of 0.8561, and Precision-Recall score of 0.8514.

4) Advantages & Limitations: Advantages:

Logistic regression is very easy to interpret. Every coefficient represents a the impact of a particular feature which makes it easily explainable.

Because logistic regression outputs the probability that a binary event occurs, you can adjust the threshold probability depending on your needs.

Logistic regression models have low variance which means they are less likely to over fit the training data.

Limitations:

Logistic regression assumes a linear relationship between features and the response variable. This can lead to inaccuracy if there is a non-linear relationship

Logistic regression is very sensitive to outliers

Its very important to choose the right features in logistic regression. Irrelevant or highly correlated features can degrade the model's performance.

IV. Results and Comparison

To solve this classification problem, we used three different machine learning models: BERT, Naive-Bayes, and Logistic Regression. We measured the performance of all three models using five different performance metrics: F1 Score, Accuracy Score, Confusion Matrix, ROC-AUC Curve, and Precision-Recall Curve.

A. Model Comparison

Looking at the performance metrics for each model, we can see that the BERT model is best to use for our data for maximum performance, while Logistic Regression is second and Naive-Bayes is last. We will provide evidence for this analysis now.

B. Error Analysis

A quick description of each performance metric:

The F1 score is a measure of the harmonic mean of precision and recall, and is a score between 0 and 1. Accuracy measures the number of times that a model was able to correctly identify the data class across the entire dataset (Sharma, 2023). A Confusion Matrix displays the True Positive, False Positive, True Negative, and False Negative values in a 2x2 matrix. The Receiver Operating Characteristic curve displays the trade off between the true-positive rate and the false-positive rate. The Area Under Curve (AUC) summarizes how good a test is regardless of the threshold. The Precision-Recall curve displays the trade-off between precision (positive predictive value) and recall (sensitivity) (Erickson, 2023).

Listed below are the performance metrics for each model.

TABLE I
Error Analysis for each Model

Model Type	F1 Score	Accuracy	ROC-AUC	Precision-Recall
BERT-base-uncased	0.7690	0.7945	0.8788	0.8834
Naive-Bayes	0.7416	0.7873	0.7782	0.6715
Logistic Regression	0.7531	0.8050	0.8561	0.8514

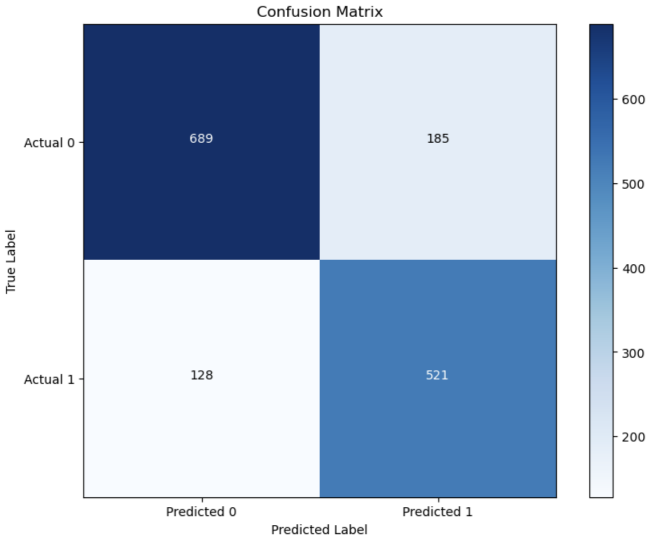

Fig. 1. Confusion matrix for BERT-base-uncased model

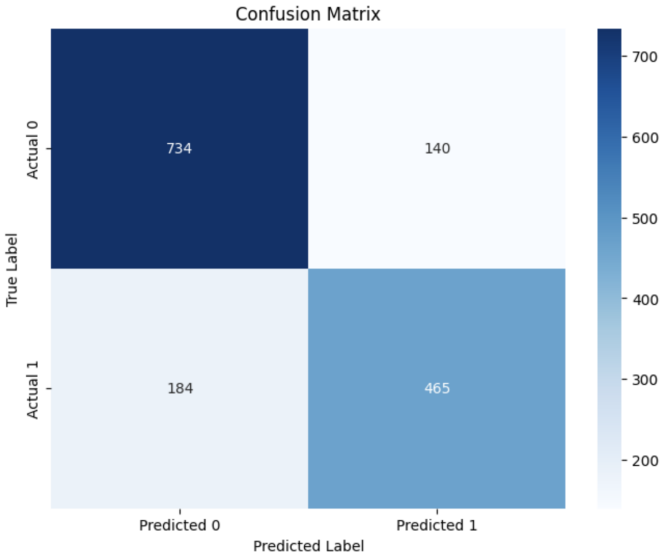

Fig. 2. Confusion matrix for Multinomial Naive-Bayes classifier

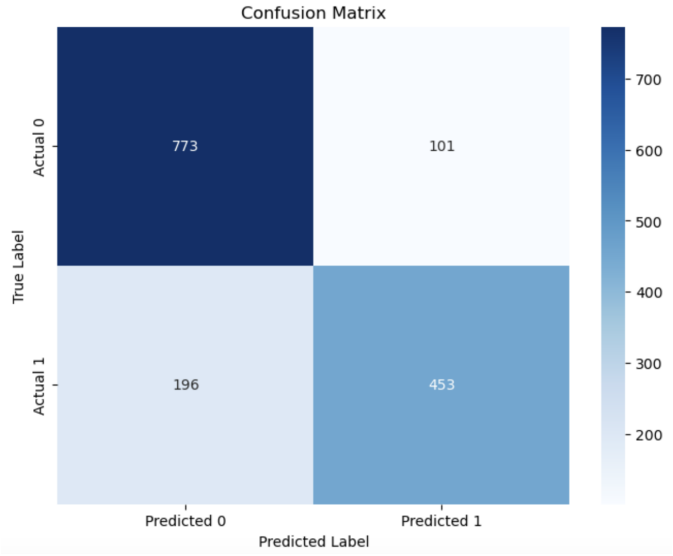

Fig. 3. Confusion matrix for Logistic Regression model

REFERENCES

[1] A. Singh, P. Soni, A. Singh, I. Potle and A. Singh, "Enhancing Disaster Tweet Classification with Ensemble Models and Multiple Embeddings," 2023 4th IEEE Global Conference for Advancement in Technology (GCAT), Bangalore, India, 2023, pp. 1-7, doi: 10.1109/GCAT59970.2023.10353510.

[2] A. Kumar and J. Singh, "Relevant Tweets Identification from Disaster-related Tweets," 2023 International Conference on Computer, Electronics & Electrical Engineering & their Applications (IC2E3), Srinagar Garhwal, India, 2023, pp. 1-6, doi: 10.1109/IC2E357697.2023.10262754.

[3] Goswami, Shriya and Raychaudhuri, Debaditya, Identification of Disaster-Related Tweets Using Natural Language Processing: International Conference on Recent Trends in Artificial Intelligence, IOT, Smart Cities & Applications (ICAISC-2020) (May 26, 2020)

[4] R. M, S. M, N. OS and E. T, "An Enhanced Framework for Disaster-Related Tweet Classification using Machine Learning Techniques," 2023 International Conference on Inventive Computation Technologies (ICICT), Lalitpur, Nepal, 2023, pp. 108-111, doi: 10.1109/ICICT57646.2023.10134249.

[5] google-bert/bert-base-uncased · Hugging Face. (2024, January 18). Huggingface.co. https://huggingface.co/google-bert/bert-base-uncased

[6] Natural Language Processing with Disaster Tweets. (n.d.). Kaggle.com. https://www.kaggle.com/competitions/nlp-getting-started/data.

[7] Erickson, B. J., & Kitamura, F. (2021). Magician's Corner: 9. Performance Metrics for Machine Learning Models. Radiology. Artificial intelligence, 3(3), e200126. https://doi.org/10.1148/ryai.2021200126

[8] Sharma, N. (2023, June 6). Understanding and applying F1 score: Ai Evaluation Essentials with hands-on coding example. Arize AI. https://arize.com/blog-course/f1-score/

[9] Jurafsky, D., & Martin, J. H. (2024, February 3). Logistic regression. Stanford.edu. https://web.stanford.edu/ jurafsky/slp3/5.pdf

[10] Kibriya, A.M., Frank, E., Pfahringer, B., Holmes, G. (2004). Multinomial Naive Bayes for Text Categorization Revisited. In: Webb, G.I., Yu, X. (eds) AI 2004: Advances in Artificial Intelligence. AI 2004. Lecture Notes in Computer Science(), vol 3339. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-30549-1_43

[11] Kavlakoglu, E. (2024, March 20). Classifying data using the Multinomial Naive Bayes algorithm. IBM Developer. https://developer.ibm.com/tutorials/awb-classifying-data-multinomial-naive-bayes-algorithm/