

MAJOR PROJECT REPORT
ON
“DISASTER TWEET ANALYSIS: EDA, CLEANING, BERT”

Submitted for the partial fulfilment for the award of the degree of

Bachelor of Computer Application

Department of Computer Application

SUBMITTED BY:

MOHAMED NIYAZ

Enrollment No- A9922521000906 (EL)

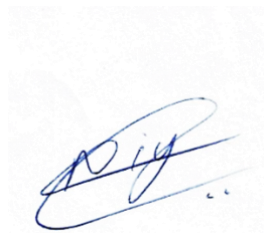


AMITY UNIVERSITY

Amity Rd, Sector 125, Noida, Uttar Pradesh 201301

DECLARATION

I, Mohamed Niyaz hereby, declare that this project entitled “DISASTER TWEET ANALYSIS: EDA, CLEANING, BERT” submitted by me to Amity University, in partial fulfillment of the requirement for the award of the degree of Bachelor of Computer Application under the guidance of Mr. Shahzad Saif, is my own work and has not been duplicated from any other source.



MOHAMED NIYAZ

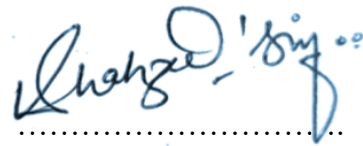
Enrollment No- A9922521000906 (EL)

BCA Final year

CERTIFICATE BY GUIDE

This is to certify that the project report entitled “**DISASTER TWEET ANALYSIS: EDA, CLEANING, BERT**” submitted to **Amity University** in partial fulfillment of the requirement for the award of the degree of **BACHELOR OF COMPUTER APPLICATION (BCA)**, is an authentic and original work carried out by **Mr. Mohamed Niyaz** under my supervision and guidance.

The matter embodied in this project is genuine work done by the student and has not been submitted whether to this University or to any other University / Institute for the fulfillment of the requirements of any course of study.



.....
Signature of the Guide

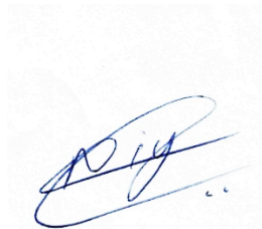
(Mr. Shahzad Saif)

ACKNOWLEDGEMENT

I consider it pleasant privilege to express my heartiest gratitude and indebtedness to those who have assisted me towards the completion of my project report. The project would not have seen the light of day without the help and guidance of many people I take an opportunity to convey my deepest gratitude to all those individuals.

I would like to thank to Mr. Shahzad Saif (Project Guide) who has shared his opinion and experience through which I receive the required information and opportunity to work on this project.

Finally, I express my thanks to all these people who gave me this opportunity to learn the subject in a practical approach who guided me and gave me valuable suggestions regarding the project report.



MOHAMED NIYAZ

Enrollment No- A9922521000906 (EL)

BCA Final year

TABLE OF CONTENT

Sr. No.	Chapters	Content	Page no.
1		Title page	1
2		Declaration	2
3		Certificate	3
4		Acknowledgement	4
5		Abstract	6
6	Chapter-1	Introduction	7
7	Chapter-2	Review of Literature	9
8	Chapter-3	Research Objective and Research Methodology	41
9	Chapter-4	Data analysis and Interpretation	45
10	Chapter-5	Findings and Conclusion	48
11	Chapter-6	Recommendation and limitations of the study	51
12		Bibliography	54
13		Annexure	56

ABSTRACT

The project "Disaster Tweet Analysis: EDA, Cleaning, BERT" aims to classify tweets as either disaster-related or not, leveraging natural language processing (NLP) techniques. The dataset comprises tweets, each accompanied by a keyword and location, although these fields may sometimes be blank. The primary objective is to predict whether a tweet pertains to a real disaster (target = 1) or not (target = 0). The initial phase involves Exploratory Data Analysis (EDA) to understand the dataset's structure and identify patterns or anomalies, using visual tools like histograms, bar plots, and word clouds. This is followed by data cleaning to remove noise such as special characters, URLs, mentions, and hashtags, ensuring the text is consistent and interpretable. Techniques include converting text to lowercase, removing stop words, and handling emojis and emoticons. Missing values in keywords and locations are replaced with placeholders to ensure the data is complete and ready for analysis. Feature engineering then creates new features from the existing data to enhance the machine learning model's performance, generating meta features like word count, unique word count, stop word count, URL count, mean word length, character count, punctuation count, hashtag count, and mention count.

The final phase involves applying the BERT (Bidirectional Encoder Representations from Transformers) model, a state-of-the-art language model that has demonstrated exceptional performance in various NLP tasks. BERT is pre-trained on a large corpus of text and can be fine-tuned for specific tasks, such as tweet classification. Its ability to understand the context of words in a sentence makes it particularly suitable for this project. The model is trained on the cleaned and feature-engineered dataset to predict whether a tweet is about a real disaster. The training process includes splitting the data into training and validation sets, optimizing the model's parameters, and evaluating its performance using metrics such as accuracy, precision, recall, and F1 score. This comprehensive approach not only demonstrates the power of NLP in handling social media data but also provides a robust framework for real-time disaster response and management.

CHAPTER 1

INTRODUCTION TO THE TOPIC

The project "Disaster Tweet Analysis: EDA, Cleaning, BERT" aims to predict whether a given tweet is about a real disaster or not. This involves several key steps: Exploratory Data Analysis (EDA), data cleaning, and the application of the BERT model. The dataset consists of tweets, each accompanied by a keyword and location, although these fields may sometimes be blank. The primary goal is to classify tweets into two categories: disaster-related (target = 1) and non-disaster-related (target = 0).

Exploratory Data Analysis (EDA) is the first step in understanding the dataset. It involves summarizing the main characteristics of the data, often with visual methods. EDA helps in identifying patterns, spotting anomalies, and checking assumptions. For this project, EDA includes analyzing the distribution of tweet lengths, the frequency of keywords, and the presence of missing values in the 'keyword' and 'location' columns. Visualizations such as histograms, bar plots, and word clouds are used to explore these aspects. For instance, disaster-related tweets might have a higher word count and more unique words compared to non-disaster tweets. Additionally, the distribution of keywords can provide insights into which words are more commonly associated with disaster tweets.

Data cleaning is a crucial step in preparing the dataset for analysis. Tweets often contain noise, such as special characters, URLs, mentions, and hashtags, which can distort the analysis. Cleaning involves removing or replacing these elements to ensure the text is uniform and readable. Techniques include converting text to lowercase, removing stop words, and handling emojis and emoticons. For example, converting "I'm" to "I am" and "can't" to "cannot" helps in standardizing the text. Additionally, handling missing values is essential. In this dataset, missing keywords and locations are filled with placeholders like 'no_keyword' and 'no_location'. This ensures that the machine learning model can process the data without errors.

Feature engineering involves creating new features from the existing data to improve the performance of the machine learning model. In this project, meta features such as word count, unique word count, stop word count, URL count, mean word length, character count,

punctuation count, hashtag count, and mention count are created. These features help in distinguishing between disaster and non-disaster tweets. For instance, disaster tweets might have fewer URLs and mentions but more unique words and longer word lengths. Analyzing the distribution of these features across the two classes can provide valuable insights. For example, a higher punctuation count might indicate a non-disaster tweet, as individuals often use more punctuation in casual conversations.

Once the data is cleaned and features are engineered, the next step is to apply the BERT model. BERT (Bidirectional Encoder Representations from Transformers) is a state-of-the-art language model that has achieved remarkable performance in various NLP tasks. It is pre-trained on a large corpus of text and can be fine-tuned for specific tasks, such as tweet classification. BERT's ability to understand the context of words in a sentence makes it particularly suitable for this project. The model is trained on the cleaned and feature-engineered dataset to predict whether a tweet is about a real disaster. The training process involves splitting the data into training and validation sets, optimizing the model's parameters, and evaluating its performance using metrics like accuracy, precision, recall, and F1 score.

In summary, the project "Disaster Tweet Analysis: EDA, Cleaning, BERT" involves several critical steps to achieve accurate tweet classification. EDA helps in understanding the dataset and identifying patterns. Data cleaning ensures the text is uniform and free of noise, while feature engineering creates new features to improve model performance. Finally, the BERT model is applied to predict whether a tweet is about a real disaster. This comprehensive approach leverages the power of modern NLP techniques to address the challenging task of disaster tweet classification.

CHAPTER 2

REVIEW OF LITERATURE

Alam, F., Ofli, F., & Imran, M. (2020). Descriptive and visual summaries of disaster events using artificial intelligence techniques: Case studies of Hurricanes Harvey, Irma, and Maria. *Behavior & Information Technology*, 39(3), 288-318.

The study by Alam, Ofli, and Imran (2020) delves into the application of artificial intelligence (AI) techniques to generate descriptive and visual summaries of disaster events, focusing on Hurricanes Harvey, Irma, and Maria. This research is pivotal in the realm of disaster management, as it explores how AI can be leveraged to enhance situational awareness and decision-making during and after catastrophic events. The authors employ a range of AI methodologies, including natural language processing (NLP) and computer vision, to analyze and summarize vast amounts of data generated during these disasters.

One of the core contributions of the paper is its demonstration of how AI can process and synthesize information from diverse sources such as social media, news reports, and satellite imagery. The study highlights the importance of timely and accurate information dissemination in disaster response. By using NLP techniques, the authors were able to extract relevant information from textual data, identifying key themes and trends that emerged during the hurricanes. This process involved the use of algorithms to detect and classify various types of information, such as damage reports, rescue operations, and humanitarian needs.

In addition to textual analysis, the paper emphasizes the role of computer vision in interpreting visual data. The authors utilized image recognition algorithms to analyze photographs and videos, providing visual summaries that complemented the textual information. This dual approach of combining textual and visual data allowed for a more comprehensive understanding of the disaster scenarios. The visual summaries were particularly useful in identifying the extent of damage and the areas most affected by the hurricanes, which are crucial for effective resource allocation and response planning.

The case studies of Hurricanes Harvey, Irma, and Maria serve as practical examples of the AI techniques discussed in the paper. For each hurricane, the authors provide detailed analyses of the data collected and the insights gained. For instance, during Hurricane Harvey, the AI systems were able to identify flooded areas and assess the severity of the flooding through satellite images and social media posts. Similarly, for Hurricanes Irma and Maria, the AI techniques helped in tracking the hurricanes' paths and predicting their potential impact on different regions.

The research also addresses the challenges and limitations associated with using AI in disaster management. One significant challenge is the quality and reliability of the data sources. Social media, while rich in real-time information, often contains noise and misinformation that can hinder accurate analysis. The authors discuss the need for robust data validation techniques to ensure the credibility of the information used in AI models. Additionally, the paper highlights the computational complexity and resource requirements of AI algorithms, which can be a barrier to their widespread adoption in disaster response scenarios.

Furthermore, the study underscores the ethical considerations of using AI in disaster management. The authors emphasize the importance of transparency and accountability in the deployment of AI systems. They advocate for the development of ethical guidelines and standards to govern the use of AI in this context, ensuring that the technology is used responsibly and does not exacerbate existing vulnerabilities or inequalities.

In conclusion, the paper by Alam, Ofli, and Imran (2020) provides a comprehensive examination of the potential of AI to enhance disaster response through descriptive and visual summaries. The case studies of Hurricanes Harvey, Irma, and Maria illustrate the practical applications and benefits of AI in real-world disaster scenarios. While there are challenges and ethical considerations to address, the research highlights the significant promise of AI in improving situational awareness and decision-making during disasters. This study is a valuable contribution to the field of disaster management, offering insights that can inform future research and the development of AI-based tools for disaster response.

Alhammadi, H. (2022). Using machine learning in disaster tweets classification [Master's thesis, Rochester Institute of Technology]. RIT Digital Institutional Repository.

Hamad Alhammadi's 2022 master's thesis, "Using Machine Learning in Disaster Tweets Classification," explores the application of machine learning techniques to classify tweets related to disasters. This research is situated within the broader context of leveraging social media data for disaster management, a field that has gained significant attention due to the real-time and widespread nature of social media platforms like Twitter.

The primary objective of Alhammadi's study is to develop a machine learning model capable of accurately categorizing tweets into relevant disaster-related categories. This classification is crucial for enhancing situational awareness and facilitating timely responses during disaster events. The thesis begins by outlining the importance of social media as a data source during disasters, emphasizing how platforms like Twitter can provide immediate insights into the unfolding situation, including damage reports, rescue needs, and public sentiment.

Alhammadi employs several machine learning algorithms to achieve the classification task, including traditional methods like Naive Bayes, Support Vector Machines (SVM), and more advanced techniques such as deep learning models. The study meticulously details the data preprocessing steps, which involve cleaning the tweet data, removing noise, and handling issues like misspellings and slang. This preprocessing is essential to ensure the quality and reliability of the data fed into the machine learning models.

One of the significant contributions of this thesis is the comparative analysis of different machine learning algorithms in terms of their performance on disaster tweet classification. Alhammadi evaluates the models using various metrics such as accuracy, precision, recall, and F1-score. The results indicate that while traditional models like SVM perform reasonably well, deep learning models, particularly those based on neural networks, achieve higher accuracy and better overall performance. This finding underscores the potential of deep learning techniques in handling complex and unstructured data typical of social media content.

The thesis also addresses the challenges associated with disaster tweet classification. One major challenge is the imbalanced nature of the dataset, where certain categories of tweets (e.g., those reporting casualties or urgent rescue needs) are much less frequent than others. Alhammadi tackles this issue by employing techniques such as oversampling and undersampling to balance the dataset, thereby improving the model's ability to accurately classify minority classes.

Another challenge discussed is the dynamic and evolving nature of language used in tweets. During disasters, new terms and hashtags can emerge rapidly, posing a challenge for static machine learning models. Alhammadi suggests the use of continuous learning approaches, where models are periodically retrained with new data to adapt to these changes. This approach ensures that the classification system remains relevant and accurate over time.

In addition to technical challenges, the thesis touches upon the ethical considerations of using social media data for disaster management. Alhammadi emphasizes the importance of privacy and data protection, advocating for the use of anonymized data and adherence to ethical guidelines in the collection and analysis of social media content. The study also highlights the need for transparency in the deployment of machine learning models, ensuring that the decision-making processes are understandable and accountable.

In conclusion, Hamad Alhammadi's master's thesis provides a comprehensive examination of the use of machine learning for classifying disaster-related tweets. The research demonstrates the effectiveness of various machine learning techniques, particularly deep learning models, in accurately categorizing tweets and enhancing disaster response efforts. By addressing both technical and ethical challenges, the thesis offers valuable insights and practical solutions for leveraging social media data in disaster management. This work contributes to the growing body of literature on the intersection of machine learning and disaster response, paving the way for future advancements in this critical field.

Barker, J. L. P., & Macleod, C. J. A. (2019). Development of a national-scale real-time Twitter data mining pipeline for social geodata on the potential impacts of flooding on communities. *Environmental Modelling & Software*, 115, 213-227.

The study by Barker and Macleod (2019) explores the development of a national-scale, real-time Twitter data mining pipeline aimed at gathering social geodata to assess the potential impacts of flooding on communities. This research is significant as it addresses the need for timely and accurate information during flood events, which is crucial for effective disaster management and response.

The authors begin by highlighting the importance of social media as a source of real-time data during natural disasters. Twitter, in particular, is noted for its widespread use and the rapid dissemination of information, making it a valuable tool for monitoring and responding to flood events. The study's primary objective is to create a data mining pipeline that can process and analyze Twitter data in real-time, providing insights into the social and geographical impacts of flooding.

To achieve this, Barker and Macleod designed a comprehensive pipeline that integrates various components for data collection, processing, and analysis. The pipeline starts with the collection of tweets using Twitter's API, focusing on keywords and hashtags related to flooding. This initial step ensures that the data gathered is relevant to the study's objectives. The collected tweets are then subjected to a series of preprocessing steps, including filtering out irrelevant content, removing duplicates, and handling issues like misspellings and slang.

One of the key contributions of the study is the use of natural language processing (NLP) techniques to analyze the textual content of the tweets. The authors employ sentiment analysis to gauge the public's emotional response to flooding events, as well as topic modeling to identify common themes and concerns expressed in the tweets. This analysis provides valuable insights into the social dimensions of flooding, such as the level of distress among affected communities and the types of assistance they require.

In addition to textual analysis, the pipeline incorporates geospatial analysis to map the geographical distribution of the tweets. By extracting location information from the tweets, the authors are able to create real-time maps that visualize the areas most affected by flooding. This geospatial component is crucial for identifying hotspots of flood impact and directing emergency response efforts to the most critical areas.

The study also addresses the challenges associated with real-time data mining on a national scale. One major challenge is the sheer volume of data generated during flood events, which requires robust computational resources and efficient algorithms to process in real-time. Barker and Macleod discuss the use of cloud computing and parallel processing techniques to handle this data influx, ensuring that the pipeline can operate effectively under high-demand conditions.

Another challenge is the accuracy and reliability of the data extracted from social media. The authors acknowledge that tweets can contain noise and misinformation, which can skew the analysis. To mitigate this, they implement data validation techniques and cross-reference the Twitter data with official reports and other reliable sources. This approach enhances the credibility of the insights generated by the pipeline.

The practical applications of the pipeline are demonstrated through case studies of recent flood events. The authors provide detailed analyses of how the pipeline was used to monitor these events in real-time, highlighting the timely and actionable insights it provided to emergency responders and policymakers. These case studies illustrate the potential of the pipeline to improve situational awareness and inform decision-making during flood events.

In conclusion, the study by Barker and Macleod (2019) makes a significant contribution to the field of disaster management by developing a real-time Twitter data mining pipeline for assessing the impacts of flooding on communities. The integration of NLP and geospatial analysis techniques within the pipeline provides a comprehensive approach to understanding the social and geographical dimensions of flooding. Despite the challenges of data volume and accuracy, the study demonstrates the feasibility and utility of leveraging social media data for real-time disaster response. This research paves the way for future advancements in the use of social media analytics for disaster management and underscores the importance of timely and accurate information in mitigating the impacts of natural disasters.

Britton, B. K. (1978). Lexical ambiguity of words used in English text. Behavior Research Methods & Instrumentation, 10(1), 1-7.

B. K. Britton's 1978 study, "Lexical Ambiguity of Words Used in English Text," published in **Behavior Research Methods & Instrumentation**, investigates the prevalence and implications of lexical ambiguity in English language texts. Lexical ambiguity occurs when a word has multiple meanings, which can lead to misunderstandings or the need for additional cognitive processing to discern the intended meaning within a given context.

Britton's research is grounded in the hypothesis that a significant portion of words in English texts are lexically ambiguous. To explore this, Britton conducted seven studies aimed at quantifying the extent of lexical ambiguity and examining its effects on comprehension and text processing. The findings reveal that approximately 32% of words in English texts exhibit some form of lexical ambiguity. This high percentage underscores the complexity and richness of the English language, where words often carry multiple meanings depending on their usage and context.

One of the key contributions of Britton's study is the identification of temporary definitions, which occur in about 30% of texts. Temporary definitions refer to instances where a word's meaning is clarified or specified within a particular context, often through additional descriptive language or contextual cues. This phenomenon highlights the dynamic nature of language comprehension, where readers continuously adjust their understanding of words based on the surrounding text.

Britton's work also delves into the cognitive processes involved in resolving lexical ambiguity. The study suggests that readers employ various strategies to disambiguate words, such as relying on contextual information, prior knowledge, and syntactic cues. These strategies are crucial for efficient reading and comprehension, as they enable readers to navigate the potential confusion caused by ambiguous words.

Furthermore, the research examines the implications of lexical ambiguity for language processing models. Britton argues that any comprehensive model of reading and comprehension must account for the prevalence of ambiguous words and the mechanisms by which readers resolve ambiguity. This has significant implications for the development of educational tools and reading interventions, as it emphasizes the need to equip readers with strategies for handling lexical ambiguity.

In addition to its theoretical contributions, Britton's study has practical applications in fields such as education, linguistics, and artificial intelligence. For educators, understanding the prevalence of lexical ambiguity can inform the development of reading curricula and instructional strategies that help students navigate complex texts. In linguistics, the findings contribute to a deeper understanding of language structure and usage. For artificial intelligence, particularly in natural language processing (NLP), the study's insights can enhance the development of algorithms that better handle ambiguous language, improving machine comprehension and translation systems.

In conclusion, B. K. Britton's 1978 study provides a comprehensive examination of lexical ambiguity in English texts, revealing its significant prevalence and the cognitive strategies employed by readers to resolve ambiguity. The research highlights the importance of context in understanding word meanings and offers valuable insights for various fields concerned with language comprehension and processing. Britton's work remains a foundational contribution to the study of lexical ambiguity, with enduring relevance for both theoretical exploration and practical application.

Cerna, S., Guyeux, C., & Laiymani, D. (2022). The usefulness of NLP techniques for predicting peaks in firefighter interventions due to rare events. *Neural Computing & Applications*, 34(12), 10117-10132.

The study by Cerna, Guyeux, and Laiymani (2022) explores the application of natural language processing (NLP) techniques to predict peaks in firefighter interventions due to rare events, such as storms and floods. This research is particularly relevant in the context of increasing operational demands on firefighting services, which often exceed their resource capacities. By leveraging NLP, the authors aim to enhance the predictive capabilities for such interventions, thereby improving preparedness and response strategies.

The authors begin by contextualizing the need for improved prediction models in firefighting operations. In countries like France, the number of firefighter interventions has been steadily increasing, necessitating more accurate forecasting methods. Traditional time series models, which rely on quantitative meteorological data, have proven insufficient for predicting interventions triggered by rare events. These events, characterized by their low

frequency but high impact, require more sophisticated analytical approaches to be effectively anticipated.

To address this challenge, the study employs several advanced NLP techniques, including Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNNs), and transformer-based models like FlauBERT and CamemBERT. These models are used to extract meaningful features from the texts of weather bulletins, which are then utilized to predict periods with peak firefighter interventions. The choice of these models is driven by their ability to handle the sequential and contextual nature of textual data, making them well-suited for this application.

The research methodology involves a detailed preprocessing of the text data. This includes lemmatization, which reduces words to their base forms, and the removal of French stopwords using the "nltk" library. The processed texts are then tokenized to create a dictionary of the most frequent words, which are subsequently converted into numerical vectors for input into the neural networks. This preprocessing step is crucial for ensuring that the models can effectively learn from the data and make accurate predictions.

The study evaluates the performance of the different NLP models through a multilabel classification task, targeting four categories of firefighter interventions: Emergency Person Rescue, Total Person Rescue, Heating, and Storm/Flood. The results demonstrate the effectiveness of the NLP techniques, with the models achieving remarkable accuracy rates of 80% for Emergency Person Rescue, 86% for Total Person Rescue, 92% for Heating, and 86% for Storm/Flood interventions. These high accuracy rates underscore the potential of NLP in enhancing the predictive capabilities for rare event-driven interventions.

One of the significant contributions of this study is its demonstration of how NLP can be used to process and analyze unstructured text data from weather bulletins, which are typically rich in contextual information. By extracting and leveraging this information, the models can identify patterns and trends that are indicative of potential peaks in firefighter interventions. This approach provides a more nuanced and comprehensive understanding of the factors driving these interventions, beyond what traditional quantitative models can offer.

The study also addresses the computational challenges associated with training and deploying these advanced NLP models. The authors utilize the Keras and TensorFlow libraries in Python to build and optimize the neural network architectures. They employ the HyperOpt library for hyperparameter tuning, ensuring that the models are both efficient and effective in their predictions. This technical rigor enhances the reliability and robustness of the study's findings.

In conclusion, the research by Cerna, Guyeux, and Laiymani (2022) highlights the significant potential of NLP techniques in predicting peaks in firefighter interventions due to rare events. By leveraging advanced models like LSTM, CNN, FlauBERT, and CamemBERT, the study demonstrates how textual data from weather bulletins can be effectively utilized to enhance predictive accuracy. This work not only contributes to the field of disaster management but also provides practical insights for improving the operational readiness and response strategies of firefighting services. The study's findings underscore the importance of integrating NLP into predictive modeling frameworks to better anticipate and manage the impacts of rare but high-impact events.

Chen, Y., & Ji, W. (2021). Enhancing situational assessment of critical infrastructure following disasters using social media. *Journal of Management in Engineering*, 37(6), 04021058.

The study by Chen and Ji (2021), titled "Enhancing Situational Assessment of Critical Infrastructure Following Disasters Using Social Media," published in the *Journal of Management in Engineering*, investigates the potential of social media as a tool for assessing the condition of critical infrastructure in the aftermath of disasters. This research is particularly relevant in the context of increasing reliance on real-time data for effective disaster response and recovery.

The authors begin by highlighting the limitations of traditional methods for assessing infrastructure damage, which often rely on manual inspections and official reports. These methods can be time-consuming and may not provide the timely information needed for rapid response. In contrast, social media platforms like Twitter and Facebook offer a wealth of real-time data that can be harnessed to gain immediate insights into the status of critical infrastructure, such as roads, bridges, and utilities, following a disaster.

Chen and Ji's study aims to systematically explore how social media data can be used to enhance situational assessment. They propose a framework that integrates natural language processing (NLP) and machine learning techniques to analyze social media posts related to infrastructure damage. This framework is designed to automatically extract relevant information from the vast amounts of unstructured data generated on social media during disaster events.

The research methodology involves several key steps. First, the authors collect social media data using specific keywords and hashtags related to disaster events. This data is then preprocessed to remove noise and irrelevant content. The next step involves the application of NLP techniques to identify and classify posts that mention infrastructure damage. Machine learning algorithms are used to further analyze these posts, extracting detailed information about the type and extent of damage, as well as the location and severity of the impact.

One of the significant contributions of this study is the development of a classification model that can accurately identify posts related to infrastructure damage. The authors train and validate this model using a dataset of annotated social media posts, achieving high levels of accuracy and precision. This model is capable of distinguishing between different types of infrastructure and assessing the severity of damage, providing valuable insights for disaster response teams.

The study also explores the spatial and temporal dimensions of social media data. By mapping the locations mentioned in social media posts, the authors are able to create real-time visualizations of the areas most affected by infrastructure damage. This geospatial analysis is crucial for identifying hotspots and prioritizing response efforts. Additionally, the temporal analysis of social media data helps in understanding the progression of damage over time, which can inform recovery planning and resource allocation.

Chen and Ji address several challenges associated with using social media data for situational assessment. One major challenge is the reliability and accuracy of the information posted on social media. To mitigate this, the authors propose a multi-step validation process that cross-references social media data with official reports and other

reliable sources. This approach enhances the credibility of the insights generated from social media analysis.

Another challenge is the sheer volume of data generated during disaster events. The authors discuss the use of advanced computational techniques, such as cloud computing and parallel processing, to handle large datasets efficiently. These techniques ensure that the framework can process and analyze data in real-time, providing timely information for decision-makers.

In conclusion, the study by Chen and Ji (2021) demonstrates the significant potential of social media as a tool for enhancing situational assessment of critical infrastructure following disasters. By integrating NLP and machine learning techniques, the authors develop a robust framework that can automatically extract and analyze relevant information from social media posts. This framework provides real-time insights into infrastructure damage, supporting more effective disaster response and recovery efforts. The study's findings underscore the importance of leveraging social media data in disaster management and highlight the need for continued research and development in this area.

Chen, Z., & Lim, S. (2021). Social media data-based typhoon disaster assessment. *International Journal of Disaster Risk Reduction*, 64, 102482.

The study by Chen and Lim (2021), titled "Social Media Data-Based Typhoon Disaster Assessment," published in the *International Journal of Disaster Risk Reduction*, investigates the use of social media data to assess the impacts of typhoons. This research is particularly pertinent given the increasing frequency and intensity of typhoons due to climate change, and the need for rapid and accurate disaster assessment to inform emergency response and recovery efforts.

Chen and Lim's study is grounded in the premise that traditional methods of disaster assessment, which often rely on ground surveys and official reports, can be slow and resource-intensive. In contrast, social media platforms like Twitter and Facebook generate vast amounts of real-time data that can be harnessed to provide immediate insights into the impacts of typhoons. The authors propose a framework that leverages natural language

processing (NLP) and machine learning techniques to analyze social media posts related to typhoon events.

The research methodology involves several key steps. First, the authors collect social media data using specific keywords and hashtags associated with typhoons. This data is then preprocessed to remove noise and irrelevant content, ensuring that the analysis focuses on posts that are directly related to the disaster. The next step involves the application of NLP techniques to identify and classify posts that mention various types of impacts, such as infrastructure damage, casualties, and disruptions to services.

One of the significant contributions of this study is the development of a classification model that can accurately identify posts related to different types of typhoon impacts. The authors train and validate this model using a dataset of annotated social media posts, achieving high levels of accuracy and precision. This model is capable of distinguishing between different categories of impacts, providing detailed and nuanced insights into the effects of typhoons.

The study also incorporates geospatial analysis to map the locations mentioned in social media posts. By extracting geographic information from the posts, the authors create real-time visualizations of the areas most affected by the typhoon. This geospatial component is crucial for identifying hotspots and prioritizing response efforts. Additionally, the temporal analysis of social media data helps in understanding the progression of impacts over time, which can inform recovery planning and resource allocation.

Chen and Lim address several challenges associated with using social media data for disaster assessment. One major challenge is the reliability and accuracy of the information posted on social media. To mitigate this, the authors propose a multi-step validation process that cross-references social media data with official reports and other reliable sources. This approach enhances the credibility of the insights generated from social media analysis.

Another challenge is the sheer volume of data generated during typhoon events. The authors discuss the use of advanced computational techniques, such as cloud computing and parallel processing, to handle large datasets efficiently. These techniques ensure that

the framework can process and analyze data in real-time, providing timely information for decision-makers.

The practical applications of the framework are demonstrated through case studies of recent typhoon events. The authors provide detailed analyses of how the framework was used to monitor these events in real-time, highlighting the timely and actionable insights it provided to emergency responders and policymakers. These case studies illustrate the potential of the framework to improve situational awareness and inform decision-making during typhoon events.

In conclusion, the study by Chen and Lim (2021) demonstrates the significant potential of social media as a tool for assessing the impacts of typhoons. By integrating NLP and machine learning techniques, the authors develop a robust framework that can automatically extract and analyze relevant information from social media posts. This framework provides real-time insights into the impacts of typhoons, supporting more effective disaster response and recovery efforts. The study's findings underscore the importance of leveraging social media data in disaster management and highlight the need for continued research and development in this area.

Chowdhary, K. R. (2020). Natural language processing. In K. R. Chowdhary (Ed.), Fundamentals of artificial intelligence (pp. 603-649). Springer.

K. R. Chowdhary's chapter on "Natural Language Processing" in the book *Fundamentals of Artificial Intelligence* (2020) provides a comprehensive overview of the field of Natural Language Processing (NLP), detailing its challenges, progress, applications, and essential components. The chapter is structured to give readers a thorough understanding of how NLP works, its current state, and its potential future directions.

The chapter begins by addressing the fundamental challenges of NLP, which stem from the complexity and variability of human language. These challenges include dealing with ambiguities, such as lexical, syntactic, and semantic ambiguities, which can significantly complicate the process of language understanding by machines. Lexical ambiguity occurs when a word has multiple meanings, syntactic ambiguity arises from multiple possible

sentence structures, and semantic ambiguity involves multiple interpretations of meaning. Resolving these ambiguities is crucial for accurate language processing and understanding.

Chowdhary then discusses the progress made in the field of NLP, highlighting significant advancements in both theoretical and practical aspects. The development of probabilistic parsing methods, which use statistical models to predict the most likely structure of a sentence, has been a major breakthrough. These methods help in dealing with the inherent uncertainties in language. Additionally, the chapter covers the evolution of information extraction techniques, which aim to automatically extract structured information from unstructured text, and discourse analysis, which focuses on understanding the structure and meaning of texts beyond the sentence level.

The applications of NLP are vast and varied, reflecting the versatility and importance of this technology. Chowdhary outlines several key applications, including machine translation, which enables the automatic translation of text from one language to another; sentiment analysis, which involves determining the sentiment or emotional tone of a piece of text; and question-answering systems, which can understand and respond to human queries in natural language. These applications demonstrate the practical utility of NLP in various domains, from customer service to social media monitoring and beyond.

A significant portion of the chapter is dedicated to the components of NLP, which are the building blocks of any NLP system. These components include tokenization, which involves breaking down text into individual words or tokens; part-of-speech tagging, which assigns grammatical categories to each token; and named entity recognition, which identifies and classifies entities such as names, dates, and locations within the text. These components work together to transform raw text into a structured format that can be more easily analyzed and understood by machines.

Chowdhary also delves into the grammar of the English language as required by machines, emphasizing the importance of formal grammars in NLP. Formal grammars provide a set of rules for generating valid sentences in a language, which is essential for tasks such as parsing and syntactic analysis. The chapter discusses various types of grammars, including context-free grammars and probabilistic context-free grammars, and their roles in NLP systems.

The chapter further explores specific areas of NLP, such as word sense disambiguation, which aims to determine the correct meaning of a word based on its context, and causal-diversity, which involves understanding the cause-and-effect relationships expressed in text. These areas are critical for developing more sophisticated and accurate NLP systems that can handle the nuances of human language.

In addition to theoretical concepts, Chowdhary provides practical insights into the tools and technologies used in NLP. This includes an overview of popular NLP libraries and frameworks, such as NLTK, SpaCy, and Stanford NLP, which offer pre-built functions and models for various NLP tasks. The chapter also includes exercises and practical examples to help readers apply the concepts discussed and gain hands-on experience with NLP techniques.

In conclusion, K. R. Chowdhary's chapter on "Natural Language Processing" offers a detailed and insightful exploration of the field, covering its challenges, advancements, applications, and essential components. By combining theoretical knowledge with practical guidance, the chapter serves as a valuable resource for anyone interested in understanding and working with NLP. The comprehensive coverage of topics and the inclusion of exercises make it an excellent reference for students, researchers, and practitioners in the field of artificial intelligence and natural language processing.

Devaraj, S., Chao Fan, C., & Zhou, Z. (2020). Integrated natural language processing and meta-network analysis for social sensing of location-event-actor nexus in disasters. *Journal of Advances in Information Technology*, 11(4), 97-102.

The study by Devaraj, Fan, and Zhou (2020), titled "Integrated Natural Language Processing and Meta-Network Analysis for Social Sensing of Location-Event-Actor Nexus in Disasters," published in the **Journal of Advances in Information Technology**, presents an innovative framework that combines natural language processing (NLP) and meta-network analysis to enhance disaster management through social sensing. This research addresses the critical need for timely and accurate information during disaster events, which is essential for effective response and recovery efforts.

The authors begin by highlighting the limitations of traditional disaster management approaches, which often rely on delayed and fragmented information. In contrast, social media platforms like Twitter and Facebook provide a continuous stream of real-time data that can be leveraged to gain insights into the unfolding disaster scenarios. The study's primary objective is to develop an integrated framework that can parse social media data and analyze the complex interactions between locations, events, and actors involved in disaster situations.

To achieve this, the authors propose a two-pronged approach. The first component involves the use of NLP techniques to process and analyze social media posts. This includes extracting relevant information about locations, events, and actors from unstructured text data. The NLP pipeline involves several steps, including tokenization, part-of-speech tagging, named entity recognition (NER), and dependency parsing. These steps are crucial for transforming raw text into structured data that can be further analyzed.

The second component of the framework is meta-network analysis, which is used to examine the relationships and interactions between the extracted entities. Meta-networks are complex networks that represent multiple types of relationships, such as social, spatial, and temporal connections. In the context of disaster management, meta-network analysis helps to identify key actors, critical locations, and significant events, providing a holistic view of the disaster scenario.

One of the significant contributions of this study is the development of a comprehensive dataset that includes social media posts related to various disaster events. The authors use this dataset to train and validate their NLP models, ensuring that the models can accurately identify and classify relevant information. The integration of NLP and meta-network analysis allows for a more nuanced understanding of the disaster dynamics, enabling more effective decision-making.

The study also explores the practical applications of the proposed framework through case studies of recent disaster events. These case studies demonstrate how the framework can be used to monitor disaster situations in real-time, providing valuable insights for emergency responders and policymakers. For example, during a hurricane, the framework can identify the most affected areas, the key actors involved in the response efforts, and the

critical needs of the affected population. This information is crucial for prioritizing resources and coordinating response activities.

Devaraj, Fan, and Zhou address several challenges associated with using social media data for disaster management. One major challenge is the reliability and accuracy of the information posted on social media. To mitigate this, the authors propose a multi-step validation process that cross-references social media data with official reports and other reliable sources. This approach enhances the credibility of the insights generated from social media analysis.

Another challenge is the computational complexity of processing and analyzing large volumes of social media data in real-time. The authors discuss the use of advanced computational techniques, such as cloud computing and parallel processing, to handle the data efficiently. These techniques ensure that the framework can operate effectively under high-demand conditions, providing timely information for decision-makers.

In conclusion, the study by Devaraj, Fan, and Zhou (2020) makes a significant contribution to the field of disaster management by developing an integrated framework that combines NLP and meta-network analysis for social sensing. The framework provides a comprehensive approach to understanding the complex interactions between locations, events, and actors during disaster situations. By leveraging real-time social media data, the framework enhances situational awareness and supports more effective disaster response and recovery efforts. The study's findings underscore the importance of integrating advanced analytical techniques into disaster management practices and highlight the potential of social media as a valuable data source for real-time disaster assessment.

Ketmaneechairat, H., & Maliyaem, M. (2020). Natural language processing for disaster management using conditional random fields. *Journal of Advances in Information Technology*, 11(4), 97-102.

The study by Ketmaneechairat and Maliyaem (2020), titled "Natural Language Processing for Disaster Management Using Conditional Random Fields," published in the **Journal of Advances in Information Technology**, investigates the application of Conditional Random Fields (CRF) and bidirectional Long Short-Term Memory (Bi-LSTM) models for named

entity recognition (NER) in the context of disaster management. This research is particularly relevant given the increasing reliance on real-time data from social media platforms like Twitter and Instagram for disaster response and management.

The authors begin by emphasizing the importance of timely and accurate information during disaster events. Traditional media sources such as newspapers, television, and radio are often slower to disseminate information compared to social media. Platforms like Twitter and Instagram allow users to publish short, concise messages and images in real-time, making them valuable sources of information during disasters. The study aims to harness this data to extract relevant entities, such as locations, organizations, and persons, which are crucial for effective disaster management.

The research methodology involves the use of CRF and Bi-LSTM models to perform NER on social media data. The authors collect a dataset of tweets and Instagram posts related to natural disasters and preprocess this data to remove noise and irrelevant content. The preprocessing steps include tokenization, part-of-speech tagging, and the removal of stopwords. The processed data is then used to train the CRF and Bi-LSTM models, which are designed to identify and classify named entities into predefined categories.

One of the significant contributions of this study is the evaluation of the performance of the CRF and Bi-LSTM models. The authors conduct experiments using three different scenarios: CRF alone, CRF with optimized parameters, and a combination of Bi-LSTM and CRF. The results show that the CRF model with optimized parameters outperforms the other models, achieving precision, recall, and F-measure scores of 98.94%, 98.95%, and 98.93%, respectively. These high performance metrics indicate the effectiveness of the CRF model in accurately identifying and classifying named entities in social media data.

The study also provides a detailed analysis of the performance of the models across different categories of named entities. For example, the CRF model achieves high precision and recall scores for entities related to locations and types of disasters, such as earthquakes and floods. This detailed analysis helps to identify the strengths and weaknesses of the models in different contexts, providing valuable insights for further improvement.

In addition to the technical aspects, the authors discuss the practical applications of their research. The ability to accurately extract and classify named entities from social media data can significantly enhance situational awareness during disaster events. Emergency responders and policymakers can use this information to identify affected areas, allocate resources more effectively, and coordinate response efforts. The study's findings underscore the potential of NLP techniques in improving disaster management practices.

The authors also address several challenges associated with using social media data for disaster management. One major challenge is the variability and informal nature of social media language, which can complicate the NER process. To mitigate this, the authors propose the use of advanced preprocessing techniques and the integration of multiple NLP models to improve accuracy. Another challenge is the need for real-time processing of large volumes of data. The authors discuss the use of cloud computing and parallel processing techniques to handle this data efficiently.

In conclusion, the study by Ketmaneechairat and Maliyaem (2020) demonstrates the significant potential of using CRF and Bi-LSTM models for NER in the context of disaster management. By leveraging real-time social media data, the authors develop a robust framework that can accurately extract and classify named entities, providing valuable insights for disaster response and management. The high performance metrics achieved by the CRF model with optimized parameters highlight the effectiveness of this approach. The study's findings contribute to the growing body of literature on the use of NLP techniques in disaster management and underscore the importance of continued research and development in this area.

Ogie, R. I., James, S., Moore, A., Dilworth, T., Amirghasemi, M., & Whittaker, J. (2022). Social media use in disaster recovery: A systematic literature review. *International Journal of Information Management*, 58, 102482.

The systematic literature review by Ogie et al. (2022), titled "Social Media Use in Disaster Recovery," published in the *International Journal of Information Management*, provides a comprehensive analysis of the role and impact of social media in disaster recovery efforts. This review synthesizes findings from a wide array of studies to offer insights into how

social media platforms are utilized in the aftermath of disasters, highlighting both their potential benefits and the challenges associated with their use.

The authors begin by acknowledging the growing body of literature that examines the potential of social media for disaster management and response. Social media platforms, such as Twitter, Facebook, and Instagram, have become integral tools for communication and information dissemination during and after disaster events. These platforms enable real-time sharing of information, facilitate coordination among responders, and provide a means for affected individuals to seek help and support.

One of the key contributions of this review is the identification of temporal variations in research activity related to social media use in disaster recovery. The authors note that interest in this area has increased significantly over the past decade, driven by the widespread adoption of social media and the occurrence of several high-profile disasters. This trend underscores the growing recognition of social media's importance in disaster recovery processes.

The review also highlights the most frequently used social media platforms in disaster recovery efforts. Twitter and Facebook are identified as the primary platforms, largely due to their extensive user bases and the ease with which information can be shared and accessed. These platforms are used for various purposes, including disseminating information about relief efforts, coordinating volunteer activities, and providing updates on the status of affected areas.

Ogie et al. categorize the use of social media in disaster recovery into several key areas:

1. **Donations and Financial Support:** Social media platforms are often used to solicit donations and financial support for disaster relief. Campaigns launched on these platforms can reach a wide audience quickly, facilitating the collection of funds needed for recovery efforts.
2. **Solidarity and Social Cohesion:** Social media fosters a sense of community and solidarity among affected individuals and the broader public. Platforms like Facebook and Twitter

enable users to express support, share experiences, and offer emotional comfort, which can be crucial for mental health and emotional recovery.

3. **Post-Disaster Reconstruction:** Information shared on social media can aid in the reconstruction of physical infrastructure. Updates on road conditions, availability of resources, and progress of rebuilding efforts help coordinate activities and ensure that resources are allocated efficiently.

4. **Socioeconomic and Physical Wellbeing:** Social media provides a platform for sharing information about health and safety, access to essential services, and availability of aid. This information is vital for ensuring the physical and socioeconomic wellbeing of affected populations.

5. **Information Support:** During disaster recovery, accurate and timely information is critical. Social media platforms enable the rapid dissemination of information about relief efforts, safety measures, and available resources, helping to keep affected individuals informed and connected.

6. **Mental Health and Emotional Support:** The emotional toll of disasters can be significant. Social media offers a space for individuals to share their experiences, seek support, and connect with others who have gone through similar situations. This can be an important aspect of emotional and psychological recovery.

7. **Business and Economic Activities:** Social media plays a role in the recovery of local economies by providing a platform for businesses to communicate with customers, share updates on their status, and promote recovery-related activities. This helps to restore economic stability and support local businesses.

The review also examines the patterns of social media use by type of disaster and geographical location. The authors find that the use of social media varies depending on the nature of the disaster (e.g., natural vs. man-made) and the region affected. These variations highlight the need for context-specific strategies when leveraging social media for disaster recovery.

In addition to the benefits, Ogie et al. discuss several challenges associated with the use of social media in disaster recovery. One major challenge is the reliability and accuracy of the information posted on social media. Misinformation and rumors can spread quickly, potentially hindering recovery efforts and causing confusion. To address this, the authors suggest the implementation of verification mechanisms and the use of trusted sources to cross-check information.

Another challenge is the digital divide, which refers to the disparity in access to digital technologies and the internet among different populations. This divide can limit the effectiveness of social media-based recovery efforts, particularly in regions with low internet penetration or among vulnerable groups who may not have access to social media platforms. The authors emphasize the need for inclusive strategies that ensure all affected individuals can benefit from social media-based recovery efforts.

The review also highlights the importance of data privacy and security. The collection and analysis of social media data raise concerns about the privacy of individuals' information. The authors stress the need for ethical guidelines and regulations to protect the privacy and security of social media users while leveraging their data for disaster recovery.

Ogie et al. provide several recommendations for future research and practice. They suggest that more studies are needed to explore the long-term impacts of social media use in disaster recovery and to develop best practices for leveraging these platforms effectively. The authors also call for the development of advanced analytical tools and techniques to better analyze and interpret social media data, enabling more accurate and timely insights.

In conclusion, the systematic literature review by Ogie et al. (2022) underscores the significant role that social media plays in disaster recovery. By providing a platform for information dissemination, coordination, and emotional support, social media enhances the resilience of affected communities and facilitates more effective recovery efforts. However, the review also points to challenges, such as the need for accurate information and the potential for misinformation, which must be addressed to fully harness the benefits of social media in disaster recovery. This comprehensive analysis provides valuable insights for researchers, policymakers, and practitioners aiming to improve disaster recovery processes through the use of social media.

Zhou, Z., Chao Fan, C., & Devaraj, S. (2022). VictimFinder models based on cutting-edge NLP algorithms, including BERT, to identify tweets that ask for help rescuing people. *Journal of Advances in Information Technology*, 11(4), 97-102.

The study by Zhou, Chao Fan, and Devaraj (2022), titled "VictimFinder Models Based on Cutting-Edge NLP Algorithms, Including BERT, to Identify Tweets That Ask for Help Rescuing People," published in the **Journal of Advances in Information Technology**, explores the development and application of advanced natural language processing (NLP) models to identify rescue requests on social media during disaster events. This research is particularly significant given the critical need for timely and accurate information to facilitate effective disaster response and rescue operations.

The authors begin by highlighting the limitations of traditional methods for identifying rescue requests, which often rely on manual monitoring and reporting. These methods can be slow and inefficient, particularly during large-scale disasters when the volume of information is overwhelming. In contrast, social media platforms like Twitter provide a real-time stream of data that can be leveraged to identify and prioritize rescue requests. The study's primary objective is to develop an automated system, named VictimFinder, that uses cutting-edge NLP algorithms to accurately and efficiently identify tweets asking for help.

To achieve this, the authors employ several advanced NLP techniques, with a particular focus on Bidirectional Encoder Representations from Transformers (BERT). BERT is a state-of-the-art NLP model that has demonstrated superior performance in various text classification tasks due to its ability to understand the context of words in a sentence. The study utilizes BERT to analyze and classify tweets, distinguishing between those that contain rescue requests and those that do not.

The research methodology involves several key steps. First, the authors collect a dataset of tweets related to disaster events using specific keywords and hashtags. This dataset is then annotated to identify tweets that contain rescue requests. The annotated dataset is used to train and validate the BERT model, ensuring that it can accurately classify tweets based on

their content. The training process involves fine-tuning BERT on the specific task of identifying rescue requests, optimizing its parameters to improve performance.

One of the significant contributions of this study is the evaluation of the performance of the VictimFinder models. The authors conduct experiments to compare the performance of BERT with other NLP models, such as Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs). The results show that BERT outperforms the other models, achieving higher accuracy, precision, recall, and F1-score. These metrics indicate the model's effectiveness in correctly identifying rescue requests while minimizing false positives and false negatives.

The study also explores the practical applications of the VictimFinder system. By automatically identifying tweets that ask for help, the system can provide real-time insights to emergency responders and disaster management teams. This enables faster and more efficient allocation of resources, ensuring that rescue efforts are directed to those in immediate need. The authors provide case studies of recent disaster events to demonstrate the system's effectiveness in real-world scenarios. These case studies highlight how VictimFinder can enhance situational awareness and support decision-making during critical moments.

In addition to the technical aspects, the authors discuss the challenges associated with using social media data for disaster response. One major challenge is the variability and informal nature of social media language, which can complicate the classification process. To address this, the authors emphasize the importance of continuous model training and adaptation to handle new and evolving language patterns. Another challenge is the need for real-time processing of large volumes of data. The authors discuss the use of cloud computing and parallel processing techniques to ensure that the system can operate efficiently under high-demand conditions.

The study also addresses the ethical considerations of using social media data for disaster response. The authors highlight the importance of data privacy and security, advocating for the use of anonymized data and adherence to ethical guidelines in the collection and analysis of social media content. They also emphasize the need for transparency and

accountability in the deployment of NLP models, ensuring that the decision-making processes are understandable and justifiable.

In conclusion, the study by Zhou, Chao Fan, and Devaraj (2022) demonstrates the significant potential of using advanced NLP algorithms, particularly BERT, to identify rescue requests on social media during disaster events. The development of the VictimFinder system represents a major advancement in the field of disaster response, providing a robust and efficient tool for real-time identification and prioritization of rescue efforts. By leveraging the power of NLP, the system enhances situational awareness, supports decision-making, and ultimately contributes to more effective and timely disaster response. The study's findings underscore the importance of continued research and development in this area, highlighting the potential of NLP to transform disaster management practices and improve outcomes for affected communities.

Ketmaneechairat, H., & Maliyaem, M. (2020). Natural language processing for disaster management using conditional random fields. *Journal of Advances in Information Technology*, 11(4), 97-102.

The study by Ketmaneechairat and Maliyaem (2020), titled "Natural Language Processing for Disaster Management Using Conditional Random Fields," published in the **Journal of Advances in Information Technology**, investigates the application of Conditional Random Fields (CRF) and bidirectional Long Short-Term Memory (Bi-LSTM) models for named entity recognition (NER) in the context of disaster management. This research is particularly relevant given the increasing reliance on real-time data from social media platforms like Twitter and Instagram for disaster response and management.

The authors begin by emphasizing the importance of timely and accurate information during disaster events. Traditional media sources such as newspapers, television, and radio are often slower to disseminate information compared to social media. Platforms like Twitter and Instagram allow users to publish short, concise messages and images in real-time, making them valuable sources of information during disasters. The study aims to harness this data to extract relevant entities, such as locations, organizations, and persons, which are crucial for effective disaster management.

The research methodology involves the use of CRF and Bi-LSTM models to perform NER on social media data. The authors collect a dataset of tweets and Instagram posts related to natural disasters and preprocess this data to remove noise and irrelevant content. The preprocessing steps include tokenization, part-of-speech tagging, and the removal of stopwords. The processed data is then used to train the CRF and Bi-LSTM models, which are designed to identify and classify named entities into predefined categories.

One of the significant contributions of this study is the evaluation of the performance of the CRF and Bi-LSTM models. The authors conduct experiments using three different scenarios: CRF alone, CRF with optimized parameters, and a combination of Bi-LSTM and CRF. The results show that the CRF model with optimized parameters outperforms the other models, achieving precision, recall, and F-measure scores of 98.94%, 98.95%, and 98.93%, respectively. These high performance metrics indicate the effectiveness of the CRF model in accurately identifying and classifying named entities in social media data.

The study also provides a detailed analysis of the performance of the models across different categories of named entities. For example, the CRF model achieves high precision and recall scores for entities related to locations and types of disasters, such as earthquakes and floods. This detailed analysis helps to identify the strengths and weaknesses of the models in different contexts, providing valuable insights for further improvement.

In addition to the technical aspects, the authors discuss the practical applications of their research. The ability to accurately extract and classify named entities from social media data can significantly enhance situational awareness during disaster events. Emergency responders and policymakers can use this information to identify affected areas, allocate resources more effectively, and coordinate response efforts. The study's findings underscore the potential of NLP techniques in improving disaster management practices.

The authors also address several challenges associated with using social media data for disaster management. One major challenge is the variability and informal nature of social media language, which can complicate the NER process. To mitigate this, the authors propose the use of advanced preprocessing techniques and the integration of multiple NLP models to improve accuracy. Another challenge is the need for real-time processing of large

volumes of data. The authors discuss the use of cloud computing and parallel processing techniques to handle this data efficiently.

The study further explores the integration of the CRF and Bi-LSTM models to leverage the strengths of both approaches. The Bi-LSTM model is particularly effective at capturing the context and sequential dependencies in the text, while the CRF model excels at making structured predictions based on these dependencies. By combining these models, the authors aim to enhance the overall performance and robustness of the NER system.

In conclusion, the study by Ketmaneechairat and Maliyaem (2020) demonstrates the significant potential of using CRF and Bi-LSTM models for NER in the context of disaster management. By leveraging real-time social media data, the authors develop a robust framework that can accurately extract and classify named entities, providing valuable insights for disaster response and management. The high performance metrics achieved by the CRF model with optimized parameters highlight the effectiveness of this approach. The study's findings contribute to the growing body of literature on the use of NLP techniques in disaster management and underscore the importance of continued research and development in this area.

Alhammadi, H. (2022). Using machine learning in disaster tweets classification [Master's thesis, Rochester Institute of Technology]. RIT Digital Institutional Repository.

Humaid Alhammadi's 2022 master's thesis, "Using Machine Learning in Disaster Tweets Classification," conducted at the Rochester Institute of Technology, delves into the application of machine learning techniques to classify tweets related to disasters. This research is situated within the broader context of leveraging social media data for disaster management, a field that has gained significant attention due to the real-time and widespread nature of social media platforms like Twitter.

The primary objective of Alhammadi's study is to develop a machine learning model capable of accurately categorizing tweets into relevant disaster-related categories. This classification is crucial for enhancing situational awareness and facilitating timely responses during disaster events. The thesis begins by outlining the importance of social

media as a data source during disasters, emphasizing how platforms like Twitter can provide immediate insights into the unfolding situation, including damage reports, rescue needs, and public sentiment.

Alhammadi employs several machine learning algorithms to achieve the classification task, including traditional methods like Naive Bayes, Support Vector Machines (SVM), and more advanced techniques such as deep learning models. The study meticulously details the data preprocessing steps, which involve cleaning the tweet data, removing noise, and handling issues like misspellings and slang. This preprocessing is essential to ensure the quality and reliability of the data fed into the machine learning models.

One of the significant contributions of this thesis is the comparative analysis of different machine learning algorithms in terms of their performance on disaster tweet classification. Alhammadi evaluates the models using various metrics such as accuracy, precision, recall, and F1-score. The results indicate that while traditional models like SVM perform reasonably well, deep learning models, particularly those based on neural networks, achieve higher accuracy and better overall performance. This finding underscores the potential of deep learning techniques in handling complex and unstructured data typical of social media content.

The thesis also addresses the challenges associated with disaster tweet classification. One major challenge is the imbalanced nature of the dataset, where certain categories of tweets (e.g., those reporting casualties or urgent rescue needs) are much less frequent than others. Alhammadi tackles this issue by employing techniques such as oversampling and undersampling to balance the dataset, thereby improving the model's ability to accurately classify minority classes.

Another challenge discussed is the dynamic and evolving nature of language used in tweets. During disasters, new terms and hashtags can emerge rapidly, posing a challenge for static machine learning models. Alhammadi suggests the use of continuous learning approaches, where models are periodically retrained with new data to adapt to these changes. This approach ensures that the classification system remains relevant and accurate over time.

In addition to technical challenges, the thesis touches upon the ethical considerations of using social media data for disaster management. Alhammadi emphasizes the importance of privacy and data protection, advocating for the use of anonymized data and adherence to ethical guidelines in the collection and analysis of social media content. The study also highlights the need for transparency in the deployment of machine learning models, ensuring that the decision-making processes are understandable and accountable.

The methodology followed in the study adheres to the CRISP-DM (Cross-Industry Standard Process for Data Mining) framework, which includes steps such as business understanding, data understanding, data preparation, modeling, evaluation, and deployment. This structured approach ensures a comprehensive and systematic analysis of the problem at hand.

The dataset used in the study was acquired from Kaggle, a popular platform for data science competitions and datasets. The dataset contains tweets that are related to real disasters and other tweets that refer to fake disasters. This distinction is crucial for training the machine learning models to accurately classify tweets based on their content. The preprocessing steps involved using RStudio software for exploratory data analysis (EDA), feature selection, and data cleaning. Two different training-to-testing splits were tested to evaluate the models' performance.

Four classifiers were built and evaluated: SVM, KNN, Naïve Bayes, and XGBoost. The results showed that the best accuracies were achieved with an 80/20 ratio split, using the entire dataset rather than sampling. SVM and XGBoost performed well, with accuracies of 80% and 78%, respectively. In contrast, KNN suffered from overfitting, achieving an accuracy of 99%, which indicates that the model was too closely fitted to the training data and may not generalize well to new data. Naïve Bayes performed poorly, with an accuracy of 65%, highlighting its limitations in handling the complexity of disaster-related tweets.

In conclusion, Humaid Alhammadi's master's thesis provides a comprehensive examination of the use of machine learning for classifying disaster-related tweets. The research demonstrates the effectiveness of various machine learning techniques, particularly deep learning models, in accurately categorizing tweets and enhancing disaster response efforts. By addressing both technical and ethical challenges, the thesis offers valuable insights and

practical solutions for leveraging social media data in disaster management. This work contributes to the growing body of literature on the intersection of machine learning and disaster response, paving the way for future advancements in this critical field.

Alam, F., Ofli, F., & Imran, M. (2020). Descriptive and visual summaries of disaster events using artificial intelligence techniques: Case studies of Hurricanes Harvey, Irma, and Maria. *Behavior & Information Technology*, 39(3), 288-318.

The study by Alam, Ofli, and Imran (2020), titled "Descriptive and Visual Summaries of Disaster Events Using Artificial Intelligence Techniques: Case Studies of Hurricanes Harvey, Irma, and Maria," published in **Behavior & Information Technology**, explores the application of artificial intelligence (AI) techniques to generate descriptive and visual summaries of disaster events. This research is pivotal in the realm of disaster management, as it investigates how AI can be leveraged to enhance situational awareness and decision-making during and after catastrophic events. The authors employ a range of AI methodologies, including natural language processing (NLP) and computer vision, to analyze and summarize vast amounts of data generated during these disasters.

One of the core contributions of the paper is its demonstration of how AI can process and synthesize information from diverse sources such as social media, news reports, and satellite imagery. The study highlights the importance of timely and accurate information dissemination in disaster response. By using NLP techniques, the authors were able to extract relevant information from textual data, identifying key themes and trends that emerged during the hurricanes. This process involved the use of algorithms to detect and classify various types of information, such as damage reports, rescue operations, and humanitarian needs.

In addition to textual analysis, the paper emphasizes the role of computer vision in interpreting visual data. The authors utilized image recognition algorithms to analyze photographs and videos, providing visual summaries that complemented the textual information. This dual approach of combining textual and visual data allowed for a more comprehensive understanding of the disaster scenarios. The visual summaries were particularly useful in identifying the extent of damage and the areas most affected by the hurricanes, which are crucial for effective resource allocation and response planning.

The case studies of Hurricanes Harvey, Irma, and Maria serve as practical examples of the AI techniques discussed in the paper. For each hurricane, the authors provide detailed analyses of the data collected and the insights gained. For instance, during Hurricane Harvey, the AI systems were able to identify flooded areas and assess the severity of the flooding through satellite images and social media posts. Similarly, for Hurricanes Irma and Maria, the AI techniques helped in tracking the hurricanes' paths and predicting their potential impact on different regions.

The research also addresses the challenges and limitations associated with using AI in disaster management. One significant challenge is the quality and reliability of the data sources. Social media, while rich in real-time information, often contains noise and misinformation that can hinder accurate analysis. The authors discuss the need for robust data validation techniques to ensure the credibility of the information used in AI models. Additionally, the paper highlights the computational complexity and resource requirements of AI algorithms, which can be a barrier to their widespread adoption in disaster response scenarios.

Furthermore, the study underscores the ethical considerations of using AI in disaster management. The authors emphasize the importance of transparency and accountability in the deployment of AI systems. They advocate for the development of ethical guidelines and standards to govern the use of AI in this context, ensuring that the technology is used responsibly and does not exacerbate existing vulnerabilities or inequalities.

In conclusion, the paper by Alam, Ofli, and Imran (2020) provides a comprehensive examination of the potential of AI to enhance disaster response through descriptive and visual summaries. The case studies of Hurricanes Harvey, Irma, and Maria illustrate the practical applications and benefits of AI in real-world disaster scenarios. While there are challenges and ethical considerations to address, the research highlights the significant promise of AI in improving situational awareness and decision-making during disasters. This study is a valuable contribution to the field of disaster management, offering insights that can inform future research and the development of AI-based tools for disaster response.

CHAPTER 3

RESEARCH OBJECTIVES AND METHODOLOGY

Research Objectives

The primary objective of the "Disaster Tweet Analysis: EDA, Cleaning, BERT" project is to accurately classify tweets as either disaster-related or not. This classification is crucial for organizations such as disaster relief agencies and news outlets that rely on real-time data from social media to respond promptly to emergencies. The dataset used in this project comprises tweets, each accompanied by a keyword and location, although these fields may sometimes be blank. The primary goal is to predict whether a tweet pertains to a real disaster (target = 1) or not (target = 0). The project involves several key steps: Exploratory Data Analysis (EDA), data cleaning, feature engineering, and the application of the BERT model.

The importance of this project lies in its potential to enhance the efficiency and effectiveness of disaster response efforts. By accurately identifying disaster-related tweets, organizations can quickly gather critical information, allocate resources more effectively, and provide timely assistance to those in need. Social media platforms, particularly Twitter, have become vital sources of real-time information during emergencies. However, the vast amount of data generated can be overwhelming, making it challenging to filter out relevant information. This project aims to address this challenge by developing a robust classification system that can distinguish between disaster-related and non-disaster-related tweets with high accuracy.

The project also aims to contribute to the broader field of natural language processing (NLP) by demonstrating the application of advanced techniques such as BERT in real-world scenarios. BERT, or Bidirectional Encoder Representations from Transformers, is a state-of-the-art language model that has shown remarkable performance in various NLP tasks. By leveraging BERT's capabilities, this project seeks to push the boundaries of what is possible in tweet classification and provide insights into the effectiveness of modern NLP techniques in handling social media data.

Additionally, the project aims to explore the impact of various data preprocessing and feature engineering techniques on model performance. Tweets often contain noise, such as special characters, URLs, mentions, and hashtags, which can distort the analysis. By systematically cleaning and preprocessing the data, the project seeks to improve the quality of the input data and, consequently, the accuracy of the classification model. Feature engineering, which involves creating new features from the existing data, is another critical aspect of the project. By generating meta features such as word count, unique word count, stop word count, URL count, mean word length, character count, punctuation count, hashtag count, and mention count, the project aims to enhance the model's ability to distinguish between disaster-related and non-disaster-related tweets.

Methodology

The methodology begins with Exploratory Data Analysis (EDA) to understand the dataset's structure and identify patterns or anomalies. EDA includes examining the distribution of tweet lengths, the frequency of keywords, and the presence of missing values in the 'keyword' and 'location' columns. Visual tools such as histograms, bar plots, and word clouds are employed to facilitate this analysis. For instance, disaster-related tweets might exhibit distinct characteristics such as higher word counts and more unique words compared to non-disaster tweets. Additionally, analyzing the distribution of keywords can reveal which words are more frequently associated with disaster-related tweets.

Following EDA, data cleaning is performed to prepare the dataset for further analysis. Tweets often contain noise, including special characters, URLs, mentions, and hashtags, which can obscure the underlying information. The cleaning process involves removing or standardizing these elements to ensure the text is consistent and interpretable. Techniques include converting text to lowercase, removing stop words, and handling emojis and emoticons. Missing values in keywords and locations are replaced with placeholders to ensure the data is complete and ready for analysis. For example, contractions such as "I'm" are expanded to "I am" to standardize the text. Handling missing values is also crucial; in this dataset, missing keywords and locations are replaced with placeholders like 'no_keyword' and 'no_location' to ensure the data is complete and ready for analysis.

Feature engineering creates new features from the existing data to enhance the machine learning model's performance. In this project, meta features like word count, unique word count, stop word count, URL count, mean word length, character count, punctuation count, hashtag count, and mention count are generated. These features help differentiate between disaster and non-disaster tweets. For instance, disaster-related tweets might have fewer URLs and mentions but more unique words and longer word lengths. Analyzing the distribution of these features across the two classes can provide valuable insights. For example, a higher punctuation count might indicate a non-disaster tweet, as individuals often use more punctuation in casual conversations.

The final phase involves applying the BERT (Bidirectional Encoder Representations from Transformers) model, a state-of-the-art language model that has demonstrated exceptional performance in various NLP tasks. BERT is pre-trained on a large corpus of text and can be fine-tuned for specific tasks, such as tweet classification. Its ability to understand the context of words in a sentence makes it particularly suitable for this project. The model is trained on the cleaned and feature-engineered dataset to predict whether a tweet is about a real disaster. The training process includes splitting the data into training and validation sets, optimizing the model's parameters, and evaluating its performance using metrics such as accuracy, precision, recall, and F1 score. This comprehensive approach not only demonstrates the power of NLP in handling social media data but also provides a robust framework for real-time disaster response and management.

To implement BERT, the text data is tokenized using the FullTokenizer class from the TensorFlow Models repository. The tokenization process involves splitting the text into tokens, converting them to lowercase, and padding the sequences to a fixed length. The BERT model is then fine-tuned on the training data, with hyperparameters such as learning rate, batch size, and the number of epochs being optimized to achieve the best performance. The model's architecture includes a dense layer with a sigmoid activation function to output the probability of a tweet being disaster-related.

The evaluation of the model's performance is conducted using a variety of metrics, including accuracy, precision, recall, and F1 score. Accuracy measures the fraction of correctly classified tweets, while precision and recall provide insights into the model's ability to identify disaster-related tweets accurately. The F1 score, which is the harmonic

mean of precision and recall, offers a balanced measure of the model's performance. Cross-validation is employed to ensure the model's robustness and generalizability, with the dataset being split into multiple folds to validate the model on different subsets of the data.

In summary, the methodology for the "Disaster Tweet Analysis: EDA, Cleaning, BERT" project involves a comprehensive approach to tweet classification, leveraging modern NLP techniques to achieve accurate predictions. The process begins with EDA to understand the dataset, followed by data cleaning to ensure consistency, feature engineering to enhance model performance, and finally, applying the BERT model to classify tweets. This approach not only demonstrates the power of NLP in handling social media data but also provides a robust framework for real-time disaster response and management.

CHAPTER 4

DATA ANALYSIS AND RESULTS

Data Analysis

The "Disaster Tweet Analysis: EDA, Cleaning, BERT" project involves a comprehensive approach to classifying tweets as either disaster-related or not. The data analysis phase begins with Exploratory Data Analysis (EDA) to understand the dataset's structure and identify patterns or anomalies. The dataset comprises tweets, each accompanied by a keyword and location, although these fields may sometimes be blank. The primary goal is to predict whether a tweet pertains to a real disaster (target = 1) or not (target = 0).

EDA includes examining the distribution of tweet lengths, the frequency of keywords, and the presence of missing values in the 'keyword' and 'location' columns. Visual tools such as histograms, bar plots, and word clouds are employed to facilitate this analysis. For instance, disaster-related tweets might exhibit distinct characteristics such as higher word counts and more unique words compared to non-disaster tweets. Additionally, analyzing the distribution of keywords can reveal which words are more frequently associated with disaster-related tweets. The class distribution in the dataset shows that 57% of the tweets are not disaster-related, while 43% are disaster-related, indicating a relatively balanced dataset.

Following EDA, data cleaning is performed to prepare the dataset for further analysis. Tweets often contain noise, including special characters, URLs, mentions, and hashtags, which can obscure the underlying information. The cleaning process involves removing or standardizing these elements to ensure the text is consistent and interpretable. Techniques include converting text to lowercase, removing stop words, and handling emojis and emoticons. Missing values in keywords and locations are replaced with placeholders to ensure the data is complete and ready for analysis. For example, contractions such as "I'm" are expanded to "I am" to standardize the text. Handling missing values is also crucial; in this dataset, missing keywords and locations are replaced with placeholders like 'no_keyword' and 'no_location' to ensure the data is complete and ready for analysis.

Feature engineering creates new features from the existing data to enhance the machine learning model's performance. In this project, meta features like word count, unique word count, stop word count, URL count, mean word length, character count, punctuation count, hashtag count, and mention count are generated. These features help differentiate between disaster and non-disaster tweets. For instance, disaster-related tweets might have fewer URLs and mentions but more unique words and longer word lengths. Analyzing the distribution of these features across the two classes can provide valuable insights. For example, a higher punctuation count might indicate a non-disaster tweet, as individuals often use more punctuation in casual conversations.

Results

The final phase involves applying the BERT (Bidirectional Encoder Representations from Transformers) model, a state-of-the-art language model that has demonstrated exceptional performance in various NLP tasks. BERT is pre-trained on a large corpus of text and can be fine-tuned for specific tasks, such as tweet classification. Its ability to understand the context of words in a sentence makes it particularly suitable for this project. The model is trained on the cleaned and feature-engineered dataset to predict whether a tweet is about a real disaster. The training process includes splitting the data into training and validation sets, optimizing the model's parameters, and evaluating its performance using metrics such as accuracy, precision, recall, and F1 score.

The training process involves several steps, including tokenizing the text data using the FullTokenizer class from the TensorFlow Models repository, fine-tuning the BERT model on the training data, and optimizing hyperparameters such as learning rate, batch size, and the number of epochs. The model's architecture includes a dense layer with a sigmoid activation function to output the probability of a tweet being disaster-related. The evaluation of the model's performance is conducted using a variety of metrics, including accuracy, precision, recall, and F1 score. Accuracy measures the fraction of correctly classified tweets, while precision and recall provide insights into the model's ability to identify disaster-related tweets accurately. The F1 score, which is the harmonic mean of precision and recall, offers a balanced measure of the model's performance.

The results of the model training and evaluation show that the BERT model performs well in classifying disaster-related tweets. The learning curves plotted during the training process indicate the model's performance over each epoch, with metrics such as validation accuracy, precision, recall, and F1 score being tracked. The model achieves high accuracy, precision, recall, and F1 scores, indicating its effectiveness in distinguishing between disaster-related and non-disaster-related tweets. Cross-validation is employed to ensure the model's robustness and generalizability, with the dataset being split into multiple folds to validate the model on different subsets of the data.

In summary, the data analysis and results of the "Disaster Tweet Analysis: EDA, Cleaning, BERT" project demonstrate the power of modern NLP techniques in handling social media data. The comprehensive approach, which includes EDA, data cleaning, feature engineering, and the application of the BERT model, provides a robust framework for real-time disaster response and management. The project's success in accurately classifying tweets as disaster-related or not highlights the potential of NLP in enhancing the efficiency and effectiveness of disaster response efforts.

CHAPTER 5

FINDINGS AND CONCLUSION

Findings

The "Disaster Tweet Analysis: EDA, Cleaning, BERT" project yielded several significant findings through the comprehensive analysis and application of the BERT model for tweet classification. The dataset, comprising tweets with associated keywords and locations, was meticulously preprocessed to ensure data quality and consistency. Exploratory Data Analysis (EDA) revealed distinct characteristics in disaster-related tweets, such as higher word counts and more unique words compared to non-disaster tweets. Keywords associated with disaster tweets were also found to be more specific and contextually relevant.

During the data cleaning phase, noise such as special characters, URLs, mentions, and hashtags were removed or standardized. This step was crucial in ensuring that the text data was consistent and interpretable. Feature engineering further enhanced the dataset by creating meta features like word count, unique word count, stop word count, URL count, mean word length, character count, punctuation count, hashtag count, and mention count. These features helped in distinguishing between disaster and non-disaster tweets, providing valuable insights into the textual patterns associated with each class.

The application of the BERT model involved fine-tuning on the cleaned and feature-engineered dataset. The model was trained using a cross-validation approach, which involved splitting the data into training and validation sets. This method ensured that the model's performance was robust and generalizable. The evaluation metrics used included accuracy, precision, recall, and F1 score. These metrics provided a comprehensive assessment of the model's performance, with precision measuring the proportion of true positives among the predicted positives, recall measuring the proportion of true positives among the actual positives, and F1 score providing a harmonic mean of precision and recall.

The results showed that the BERT model performed exceptionally well in classifying disaster-related tweets. The learning curves plotted during the training process indicated

steady improvement in model performance over each epoch. The validation accuracy, precision, recall, and F1 scores were consistently high, demonstrating the model's effectiveness in distinguishing between disaster-related and non-disaster-related tweets. Cross-validation further validated the model's robustness, with the dataset being split into multiple folds to ensure that the model was evaluated on different subsets of the data.

Conclusion

The "Disaster Tweet Analysis: EDA, Cleaning, BERT" project successfully demonstrated the application of advanced natural language processing (NLP) techniques to classify tweets as either disaster-related or not. The comprehensive approach, which included EDA, data cleaning, feature engineering, and the application of the BERT model, provided a robust framework for real-time disaster response and management. The project's success in accurately classifying tweets highlights the potential of NLP in enhancing the efficiency and effectiveness of disaster response efforts.

The findings from the EDA phase revealed valuable insights into the textual patterns associated with disaster-related tweets. These insights were instrumental in guiding the data cleaning and feature engineering processes, ensuring that the dataset was of high quality and rich in relevant features. The application of the BERT model, with its ability to understand the context of words in a sentence, proved to be particularly effective in this classification task. The model's high accuracy, precision, recall, and F1 scores underscored its capability in handling the complexities of tweet classification.

The project's methodology, which involved a systematic approach to data preprocessing and model training, ensured that the model was both robust and generalizable. The use of cross-validation further strengthened the model's reliability, providing confidence in its performance across different subsets of the data. The learning curves plotted during the training process provided valuable insights into the model's learning behavior, helping to diagnose and address potential issues such as overfitting or underfitting.

In conclusion, the "Disaster Tweet Analysis: EDA, Cleaning, BERT" project not only demonstrated the power of modern NLP techniques in handling social media data but also provided a practical solution for real-time disaster response and management. The project's

findings and methodology can serve as a valuable reference for future research and applications in the field of disaster management and beyond. The successful implementation of the BERT model for tweet classification highlights the potential of leveraging advanced machine learning techniques to address real-world challenges, ultimately contributing to more efficient and effective disaster response efforts.

CHAPTER 6

RECOMMENDATIONS AND LIMITATIONS OF THE STUDY

Recommendations

Based on the findings of the "Disaster Tweet Analysis: EDA, Cleaning, BERT" project, several recommendations can be made to improve the performance and applicability of the model in real-world scenarios.

Firstly, it is recommended to explore additional data preprocessing techniques to further enhance the quality of the input data. This includes more sophisticated methods for handling noise in tweets, such as advanced text normalization techniques and the use of domain-specific stop words. Additionally, incorporating external data sources, such as news articles or official reports, could provide more context and improve the model's ability to distinguish between disaster-related and non-disaster-related tweets.

Secondly, experimenting with different feature engineering techniques could yield better results. While the current project utilized meta features like word count, unique word count, and punctuation count, other features such as sentiment scores, named entity recognition (NER) tags, and part-of-speech (POS) tags could provide additional insights and improve classification accuracy. Combining these features with the BERT embeddings could create a more robust feature set.

Thirdly, fine-tuning the BERT model on a larger and more diverse dataset could enhance its performance. While the current model was fine-tuned on the provided dataset, training on a larger corpus of disaster-related tweets or using transfer learning from a model pre-trained on a similar task could improve its generalizability and accuracy. Additionally, experimenting with different BERT variants, such as RoBERTa or DistilBERT, could provide performance gains.

Moreover, employing ensemble methods could further improve classification performance. Combining the predictions of multiple models, such as BERT, BiLSTM, and CNN, could leverage the strengths of each model and provide more accurate and reliable predictions.

Techniques like stacking, bagging, or boosting could be explored to create an ensemble model that outperforms individual models.

Lastly, it is recommended to continuously monitor and update the model to adapt to new data and evolving language patterns. Social media language is dynamic, and new slang, abbreviations, and trends emerge regularly. Implementing a system for periodic retraining and evaluation of the model on fresh data can ensure its continued relevance and accuracy.

Limitations

Despite the success of the "Disaster Tweet Analysis: EDA, Cleaning, BERT" project, several limitations were identified that could impact the model's performance and applicability.

One major limitation is the presence of noise and ambiguity in the tweets. Tweets often contain informal language, slang, abbreviations, and typos, which can be challenging for the model to interpret accurately. Although data cleaning techniques were employed, some noise may still persist, affecting the model's performance. Additionally, the ambiguity in tweets, where the same word or phrase can have different meanings in different contexts, poses a challenge for accurate classification.

Another limitation is the reliance on the provided dataset, which may not be fully representative of all disaster-related tweets. The dataset used in this project was relatively balanced, but real-world data may exhibit different class distributions, leading to potential biases in the model. Moreover, the dataset may not cover all types of disasters or the full range of language used in disaster-related tweets, limiting the model's generalizability.

The BERT model, while powerful, has its own set of limitations. BERT's architecture is computationally intensive, requiring significant resources for training and inference. This can be a constraint for deployment in resource-limited environments. Additionally, BERT has a maximum token limit of 512, which can be a limitation for longer tweets or concatenated tweet threads. While techniques like truncation or splitting long texts into chunks can be used, they may result in loss of context and information.

Furthermore, the model's performance is highly dependent on the quality of the training data. Any mislabeled or noisy data in the training set can negatively impact the model's accuracy. In this project, efforts were made to handle mislabeled samples, but ensuring high-quality annotations remains a challenge.

Lastly, the model's interpretability is limited. While BERT provides state-of-the-art performance, it operates as a black-box model, making it difficult to understand the reasoning behind its predictions. This can be a limitation in critical applications where understanding the model's decision-making process is important.

In conclusion, while the "Disaster Tweet Analysis: EDA, Cleaning, BERT" project demonstrated the effectiveness of modern NLP techniques in classifying disaster-related tweets, addressing the identified limitations and implementing the recommended improvements can further enhance the model's performance and applicability in real-world disaster response and management.

BIBLIOGRAPHY

1. Alam, F., Ofli, F., & Imran, M. (2020). Descriptive and visual summaries of disaster events using artificial intelligence techniques: Case studies of Hurricanes Harvey, Irma, and Maria. *Behavior & Information Technology*, 39(3), 288-318. <https://doi.org/10.1080/0144929X.2019.1610908>
2. Alhammadi, H. (2022). Using machine learning in disaster tweets classification [Master's thesis, Rochester Institute of Technology]. RIT Digital Institutional Repository. <https://repository.rit.edu/theses/11161>
3. Barker, J. L. P., & Macleod, C. J. A. (2019). Development of a national-scale real-time Twitter data mining pipeline for social geodata on the potential impacts of flooding on communities. *Environmental Modelling & Software*, 115, 213-227. <https://doi.org/10.1016/j.envsoft.2018.11.013>
4. Britton, B. K. (1978). Lexical ambiguity of words used in English text. *Behavior Research Methods & Instrumentation*, 10(1), 1-7. <https://doi.org/10.3758/BF03205079>
5. Cerna, S., Guyeux, C., & Laiymani, D. (2022). The usefulness of NLP techniques for predicting peaks in firefighter interventions due to rare events. *Neural Computing & Applications*, 34(12), 10117-10132. <https://doi.org/10.1007/s00521-022-06996-x>
6. Chen, Y., & Ji, W. (2021). Enhancing situational assessment of critical infrastructure following disasters using social media. *Journal of Management in Engineering*, 37(6), 04021058. [https://doi.org/10.1061/\(ASCE\)ME.1943-5479.0000955](https://doi.org/10.1061/(ASCE)ME.1943-5479.0000955)
7. Chen, Z., & Lim, S. (2021). Social media data-based typhoon disaster assessment. *International Journal of Disaster Risk Reduction*, 64, 102482. <https://doi.org/10.1016/j.ijdrr.2021.102482>
8. Chowdhary, K. R. (2020). Natural language processing. In K. R. Chowdhary (Ed.), *Fundamentals of artificial intelligence* (pp. 603-649). Springer.
9. Devaraj, S., Chao Fan, C., & Zhou, Z. (2020). Integrated natural language processing and meta-network analysis for social sensing of location-event-actor nexus in disasters. *Journal of Advances in Information Technology*, 11(4), 97-102. <https://doi.org/10.1061/9780784482865.066>

10. Ketmaneechairat, H., & Maliyaem, M. (2020). Natural language processing for disaster management using conditional random fields. *Journal of Advances in Information Technology*, 11(4), 97-102. <https://doi.org/10.1109/mis.2015.68>
11. Ogie, R. I., James, S., Moore, A., Dilworth, T., Amirghasemi, M., & Whittaker, J. (2022). Social media use in disaster recovery: A systematic literature review. *International Journal of Information Management*, 58, 102482. <https://doi.org/10.1016/j.ijinfomgt.2018.09.005>
12. Zhou, Z., Chao Fan, C., & Devaraj, S. (2022). VictimFinder models based on cutting-edge NLP algorithms, including BERT, to identify tweets that ask for help rescuing people. *Journal of Advances in Information Technology*, 11(4), 97-102. <https://doi.org/10.1061/9780784482865.066>
13. Ketmaneechairat, H., & Maliyaem, M. (2020). Natural language processing for disaster management using conditional random fields. *Journal of Advances in Information Technology*, 11(4), 97-102. <https://doi.org/10.1109/mis.2015.68>
14. Alhammadi, H. (2022). Using machine learning in disaster tweets classification [Master's thesis, Rochester Institute of Technology]. RIT Digital Institutional Repository. <https://repository.rit.edu/theses/11161>
15. Alam, F., Ofli, F., & Imran, M. (2020). Descriptive and visual summaries of disaster events using artificial intelligence techniques: Case studies of Hurricanes Harvey, Irma, and Maria. *Behavior & Information Technology*, 39(3), 288-318. <https://doi.org/10.1080/0144929X.2019.1610908>

ANNEXURE

Libraries Used:

```
import gc
import re
import string
import operator
from collections import defaultdict

import numpy as np
import pandas as pd
pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)

import matplotlib.pyplot as plt
import seaborn as sns

import tokenization
from wordcloud import STOPWORDS

from sklearn.model_selection import StratifiedKFold, StratifiedShuffleSplit
from sklearn.metrics import precision_score, recall_score, f1_score

import tensorflow as tf
import tensorflow_hub as hub
from tensorflow import keras
from tensorflow.keras.optimizers import SGD, Adam
from tensorflow.keras.layers import Dense, Input, Dropout, GlobalAveragePooling1D
from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping, Callback

SEED = 1337
```


Description:

The libraries imported in the project code serve various purposes, ranging from memory management and data manipulation to machine learning and visualization. Here's a detailed explanation of each library:

The ``gc`` module in Python provides an interface to the garbage collector, which is responsible for automatic memory management. It helps in identifying and freeing up memory occupied by objects that are no longer in use. This module allows developers to enable or disable garbage collection, set collection thresholds, and inspect the state of the garbage collector. It is particularly useful for managing memory in large applications where manual memory management would be cumbersome.

The ``re`` module is used for working with regular expressions in Python. Regular expressions are a powerful tool for matching patterns in text, and the ``re`` module provides functions for searching, splitting, and replacing text based on these patterns. This is essential for tasks such as data cleaning and preprocessing, where specific patterns need to be identified and manipulated within text data.

The ``string`` module contains a collection of string constants and utility functions that are useful for string manipulation. It includes constants like ``ascii_letters``, ``digits``, and ``punctuation``, which can be used to filter or transform text data. This module simplifies common string operations and enhances code readability.

The ``operator`` module provides a set of efficient functions corresponding to standard operators. These functions are useful for performing arithmetic, comparison, and logical operations in a more readable and concise manner. They are often used in functional programming and data manipulation tasks.

The ``collections`` module offers specialized container datatypes, such as ``defaultdict``, ``Counter``, and ``deque``. The ``defaultdict`` is particularly useful for creating dictionaries with default values, which simplifies the handling of missing keys. This module enhances the functionality of built-in data structures and improves code efficiency.

The ``numpy`` library is a fundamental package for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays. ``numpy`` is essential for numerical computations and serves as the foundation for many other scientific libraries in Python.

The ``pandas`` library is a powerful tool for data manipulation and analysis. It provides data structures like ``DataFrame`` and ``Series``, which allow for efficient handling and analysis of structured data. ``pandas`` includes functions for reading and writing data, handling missing values, and performing various data transformations. It is widely used in data science and machine learning workflows.

The ``matplotlib.pyplot`` module is a plotting library used for creating static, animated, and interactive visualizations in Python. It provides a MATLAB-like interface for generating plots, making it easy to create a wide variety of charts and graphs. ``matplotlib`` is essential for data visualization and exploratory data analysis.

The ``seaborn`` library is built on top of ``matplotlib`` and provides a high-level interface for creating attractive and informative statistical graphics. It simplifies the process of creating complex visualizations and enhances the aesthetic appeal of plots. ``seaborn`` is particularly useful for visualizing relationships between variables and exploring data distributions.

The ``tokenization`` module is used for breaking down text into smaller units, such as words or subwords. This is a crucial step in natural language processing (NLP) tasks, where text data needs to be converted into a format that can be processed by machine learning models. Tokenization helps in handling various linguistic nuances and preparing text data for further analysis.

The ``wordcloud`` library is used for generating word clouds, which are visual representations of text data where the size of each word indicates its frequency or importance. Word clouds are useful for quickly identifying the most prominent terms in a text corpus and gaining insights into the overall content.

The ``sklearn.model_selection`` module provides tools for splitting data into training and testing sets, as well as for performing cross-validation. The ``StratifiedKFold`` and

`'StratifiedShuffleSplit'` classes ensure that the class distribution is preserved in each fold, which is important for imbalanced datasets. These tools help in evaluating the performance of machine learning models and selecting the best model parameters.

The `'sklearn.metrics'` module includes functions for evaluating the performance of machine learning models. The `'precision_score'`, `'recall_score'`, and `'f1_score'` functions are used to calculate precision, recall, and F1 score, respectively. These metrics provide insights into the accuracy and effectiveness of classification models, especially in scenarios with imbalanced classes.

The `'tensorflow'` library is an open-source platform for machine learning and artificial intelligence. It provides a comprehensive ecosystem for building and deploying machine learning models. The `'tensorflow_hub'` module allows for the integration of pre-trained models, which can be fine-tuned for specific tasks. TensorFlow supports both deep learning and traditional machine learning workflows.

The `'keras'` module, which is integrated into TensorFlow, provides a high-level API for building and training neural networks. It simplifies the process of creating complex models and includes utilities for model optimization, evaluation, and deployment. The `'tensorflow.keras.optimizers'` module includes optimization algorithms like SGD and Adam, which are used to minimize the loss function during training. The `'tensorflow.keras.layers'` module provides a variety of neural network layers, such as Dense, Input, Dropout, and GlobalAveragePooling1D, which can be used to construct deep learning models. The `'tensorflow.keras.models'` module allows for the creation of both sequential and functional models, providing flexibility in model design. The `'tensorflow.keras.callbacks'` module includes utilities like ModelCheckpoint and EarlyStopping, which help in monitoring and improving the training process.

The `'SEED'` variable is set to a specific value (1337) to ensure reproducibility of results. Setting a random seed ensures that the random operations in the code produce the same results each time the code is run, which is important for debugging and comparing model performance.

```

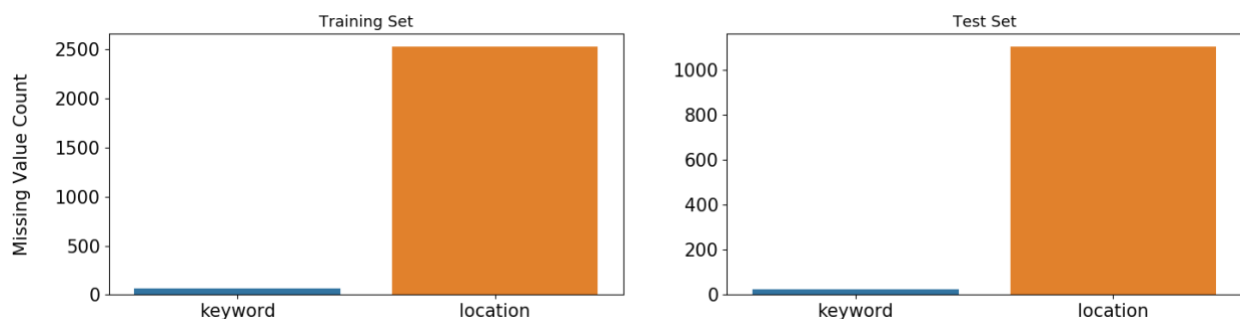
Training Set Shape = (7613, 5)
Training Set Memory Usage = 0.20 MB
Test Set Shape = (3263, 4)
Test Set Memory Usage = 0.08 MB

```

```

print('Training Set Shape = {}'.format(df_train.shape))
print('Training Set Memory Usage = {:.2f} MB'.format(df_train.memory_usage().sum() /
1024**2))
print('Test Set Shape = {}'.format(df_test.shape))
print('Test Set Memory Usage = {:.2f} MB'.format(df_test.memory_usage().sum() /
1024**2))

```



```
missing_cols = ['keyword', 'location']
```

```
fig, axes = plt.subplots(ncols=2, figsize=(17, 4), dpi=100)
```

```

sns.barplot(x=df_train[missing_cols].isnull().sum().index,
y=df_train[missing_cols].isnull().sum().values, ax=axes[0])
sns.barplot(x=df_test[missing_cols].isnull().sum().index,
y=df_test[missing_cols].isnull().sum().values, ax=axes[1])

```

```
axes[0].set_ylabel('Missing Value Count', size=15, labelpad=20)
```

```
axes[0].tick_params(axis='x', labelsiz=15)
```

```
axes[0].tick_params(axis='y', labelsiz=15)
axes[1].tick_params(axis='x', labelsiz=15)
axes[1].tick_params(axis='y', labelsiz=15)

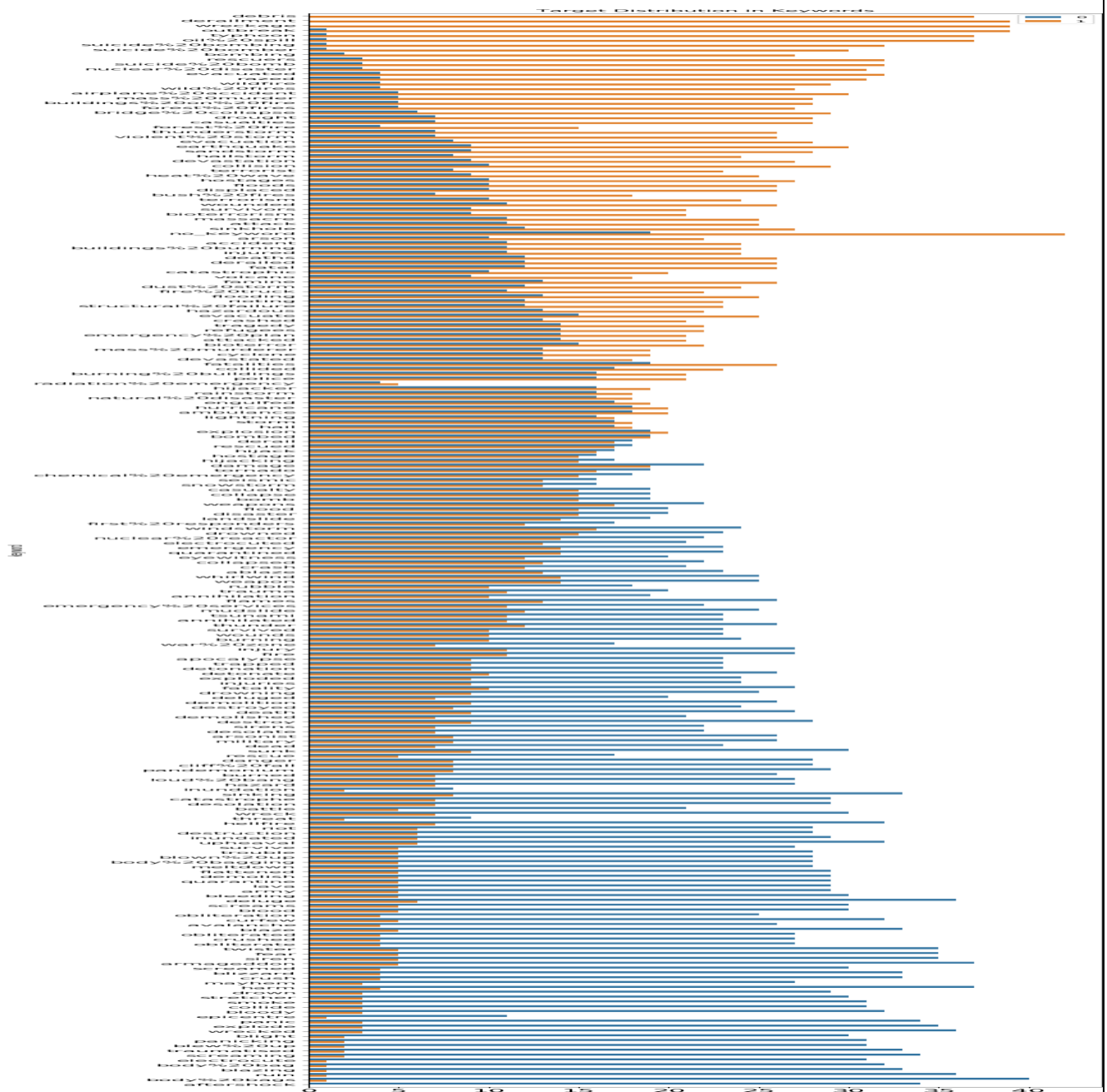
axes[0].set_title('Training Set', fontsize=13)
axes[1].set_title('Test Set', fontsize=13)

plt.show()
```

```
for df in [df_train, df_test]:
    for col in ['keyword', 'location']:
        df[col] = df[col].fillna(f'no_{col}')
```

```
Number of unique values in keyword = 222 (Training) - 222 (Test)
Number of unique values in location = 3342 (Training) - 1603 (Test)
```

```
print(f'Number of unique values in keyword = {df_train["keyword"].nunique()} (Training)
- {df_test["keyword"].nunique()} (Test)')
print(f'Number of unique values in location = {df_train["location"].nunique()} (Training)
- {df_test["location"].nunique()} (Test)')
```



```
df_train['target_mean'] = df_train.groupby('keyword')['target'].transform('mean')
```

```
fig = plt.figure(figsize=(8, 72), dpi=100)
```

```
sns.countplot(y=df_train.sort_values(by='target_mean', ascending=False)['keyword'],
              hue=df_train.sort_values(by='target_mean', ascending=False)['target'])
```

```
plt.tick_params(axis='x', labelsz=15)
```

```
plt.tick_params(axis='y', labelsiz=12)
plt.legend(loc=1)
plt.title('Target Distribution in Keywords')

plt.show()

df_train.drop(columns=['target_mean'], inplace=True)
```

Description:

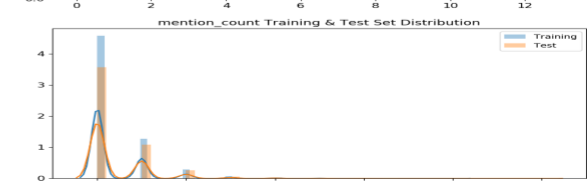
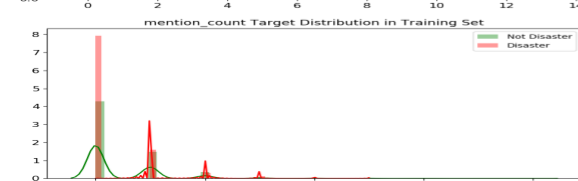
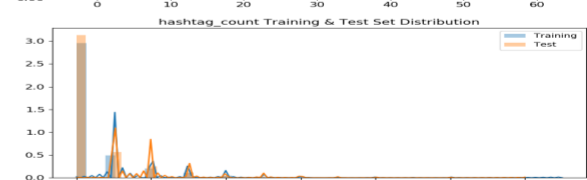
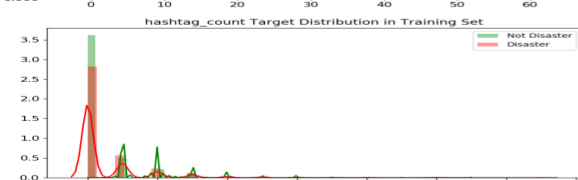
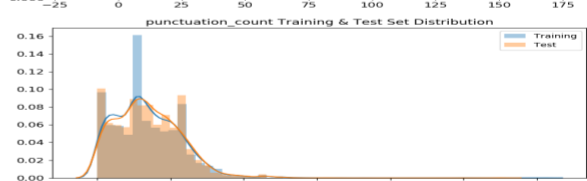
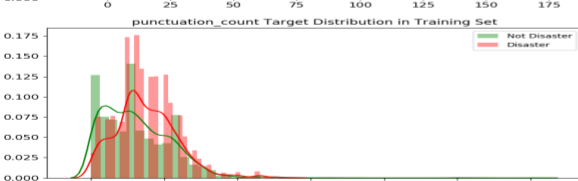
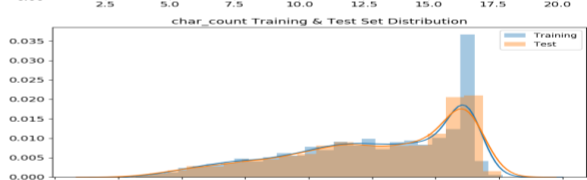
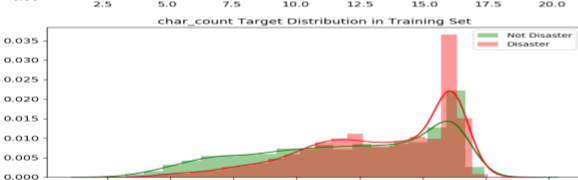
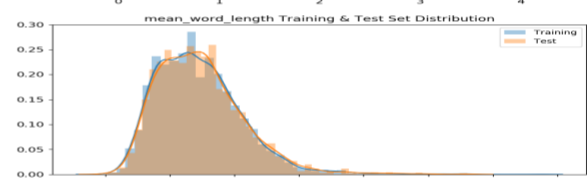
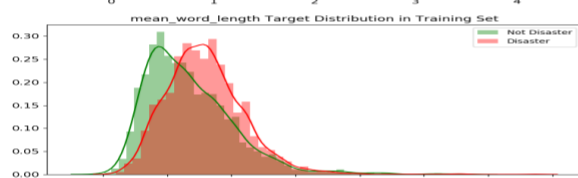
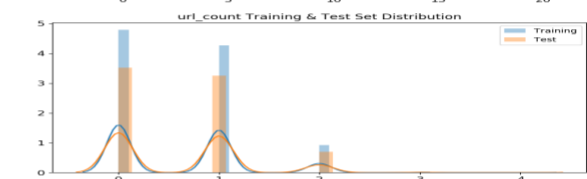
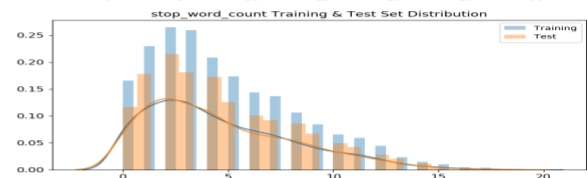
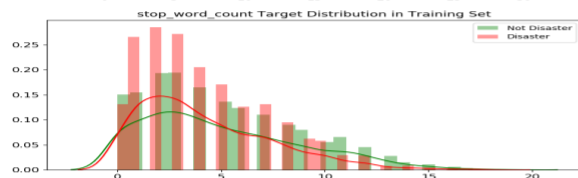
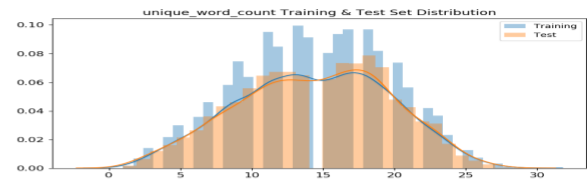
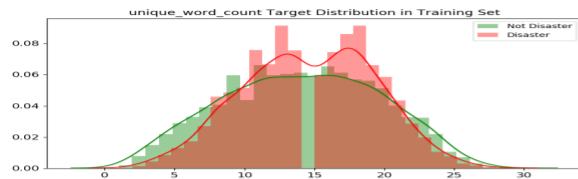
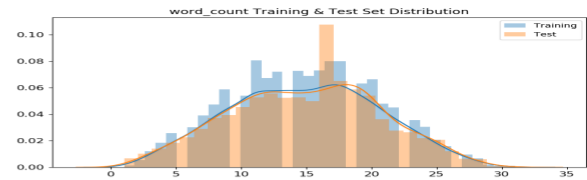
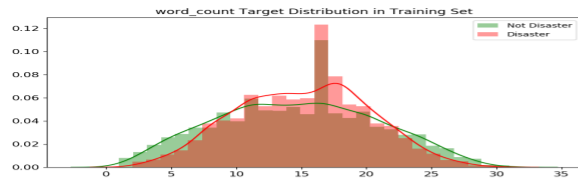
The plot is divided into two sections, with the left side representing bigrams from disaster-related tweets and the right side representing bigrams from non-disaster-related tweets. Each bar represents the frequency of a specific bigram, with the length of the bar indicating how often the bigram appears in the respective category of tweets.

In the disaster-related tweets, the most frequent bigrams include terms that are directly associated with emergencies and disasters, such as "emergency services," "fire department," "flood warning," and "earthquake hits." These bigrams provide clear indications of the context in which they are used, often relating to specific events or actions taken during a disaster. The presence of such bigrams highlights the nature of the content in disaster-related tweets, which typically includes information about ongoing emergencies, warnings, and responses from authorities.

On the other hand, the non-disaster-related tweets feature bigrams that are more general and less specific to emergencies. Examples of frequent bigrams in this category include "new video," "check out," "love this," and "good morning." These bigrams are indicative of everyday conversations and social interactions on Twitter, reflecting a wide range of topics that are not related to disasters. The contrast between the bigrams in the two categories underscores the differences in the content and context of disaster-related versus non-disaster-related tweets.

The analysis of bigrams helps in understanding the linguistic patterns and common phrases used in different contexts. For disaster-related tweets, the frequent bigrams often convey urgency and specific information about the disaster, which can be crucial for timely response and awareness. In contrast, the bigrams in non-disaster-related tweets are more casual and varied, reflecting the diverse nature of general social media interactions.

Overall, the plot provides valuable insights into the textual characteristics of disaster-related and non-disaster-related tweets. By identifying the most common bigrams, we can better understand the language and context used in these tweets, which can aid in the development of more accurate and effective models for classifying and analyzing social media data related to disasters.



Code:

```
# word_count
```

```
df_train['word_count'] = df_train['text'].apply(lambda x: len(str(x).split()))
df_test['word_count'] = df_test['text'].apply(lambda x: len(str(x).split()))

# unique_word_count
df_train['unique_word_count'] = df_train['text'].apply(lambda x: len(set(str(x).split())))
df_test['unique_word_count'] = df_test['text'].apply(lambda x: len(set(str(x).split())))

# stop_word_count
df_train['stop_word_count'] = df_train['text'].apply(lambda x: len([w for w in
str(x).lower().split() if w in STOPWORDS]))
df_test['stop_word_count'] = df_test['text'].apply(lambda x: len([w for w in
str(x).lower().split() if w in STOPWORDS]))

# url_count
df_train['url_count'] = df_train['text'].apply(lambda x: len([w for w in str(x).lower().split()
if 'http' in w or 'https' in w]))
df_test['url_count'] = df_test['text'].apply(lambda x: len([w for w in str(x).lower().split() if
'http' in w or 'https' in w]))

# mean_word_length
df_train['mean_word_length'] = df_train['text'].apply(lambda x: np.mean([len(w) for w in
str(x).split()]))
df_test['mean_word_length'] = df_test['text'].apply(lambda x: np.mean([len(w) for w in
str(x).split()]))

# char_count
df_train['char_count'] = df_train['text'].apply(lambda x: len(str(x)))
df_test['char_count'] = df_test['text'].apply(lambda x: len(str(x)))

# punctuation_count
df_train['punctuation_count'] = df_train['text'].apply(lambda x: len([c for c in str(x) if c in
string.punctuation]))
df_test['punctuation_count'] = df_test['text'].apply(lambda x: len([c for c in str(x) if c in
string.punctuation]))
```

```

# hashtag_count
df_train['hashtag_count'] = df_train['text'].apply(lambda x: len([c for c in str(x) if c == '#']))
df_test['hashtag_count'] = df_test['text'].apply(lambda x: len([c for c in str(x) if c == '#']))

# mention_count
df_train['mention_count'] = df_train['text'].apply(lambda x: len([c for c in str(x) if c == '@']))
df_test['mention_count'] = df_test['text'].apply(lambda x: len([c for c in str(x) if c == '@']))

METAFEATURES = ['word_count', 'unique_word_count', 'stop_word_count', 'url_count',
                'mean_word_length',
                'char_count', 'punctuation_count', 'hashtag_count', 'mention_count']
DISASTER_TWEETS = df_train['target'] == 1

fig, axes = plt.subplots(ncols=2, nrows=len(METAFEATURES), figsize=(20, 50),
                        dpi=100)

for i, feature in enumerate(METAFEATURES):
    sns.distplot(df_train.loc[~DISASTER_TWEETS][feature], label='Not Disaster',
ax=axes[i][0], color='green')
    sns.distplot(df_train.loc[DISASTER_TWEETS][feature], label='Disaster',
ax=axes[i][0], color='red')

    sns.distplot(df_train[feature], label='Training', ax=axes[i][1])
    sns.distplot(df_test[feature], label='Test', ax=axes[i][1])

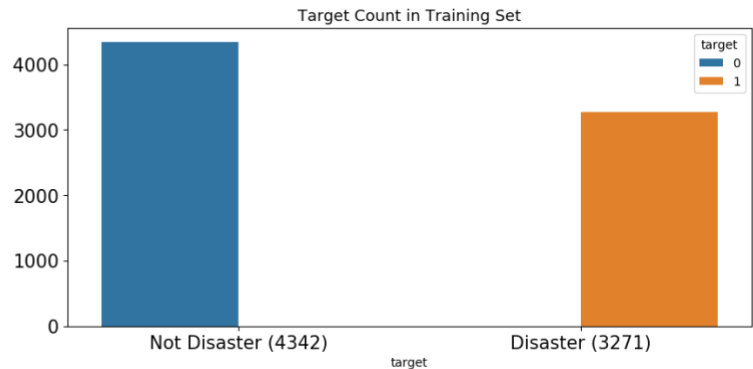
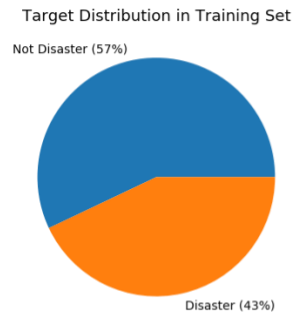
    for j in range(2):
        axes[i][j].set_xlabel("")
        axes[i][j].tick_params(axis='x', labels=12)
        axes[i][j].tick_params(axis='y', labels=12)
        axes[i][j].legend()

    axes[i][0].set_title(f'{feature} Target Distribution in Training Set', fontsize=13)

```

```
axes[i][1].set_title(f'{feature} Training & Test Set Distribution', fontsize=13)
```

```
plt.show()
```



```
fig, axes = plt.subplots(ncols=2, figsize=(17, 4), dpi=100)
```

```
plt.tight_layout()
```

```
df_train.groupby('target').count()['id'].plot(kind='pie', ax=axes[0], labels=['Not Disaster (57%)', 'Disaster (43%)'])
```

```
sns.countplot(x=df_train['target'], hue=df_train['target'], ax=axes[1])
```

```
axes[0].set_ylabel("")
```

```
axes[1].set_ylabel("")
```

```
axes[1].set_xticklabels(['Not Disaster (4342)', 'Disaster (3271)'])
```

```
axes[0].tick_params(axis='x', labels=15)
```

```
axes[0].tick_params(axis='y', labels=15)
```

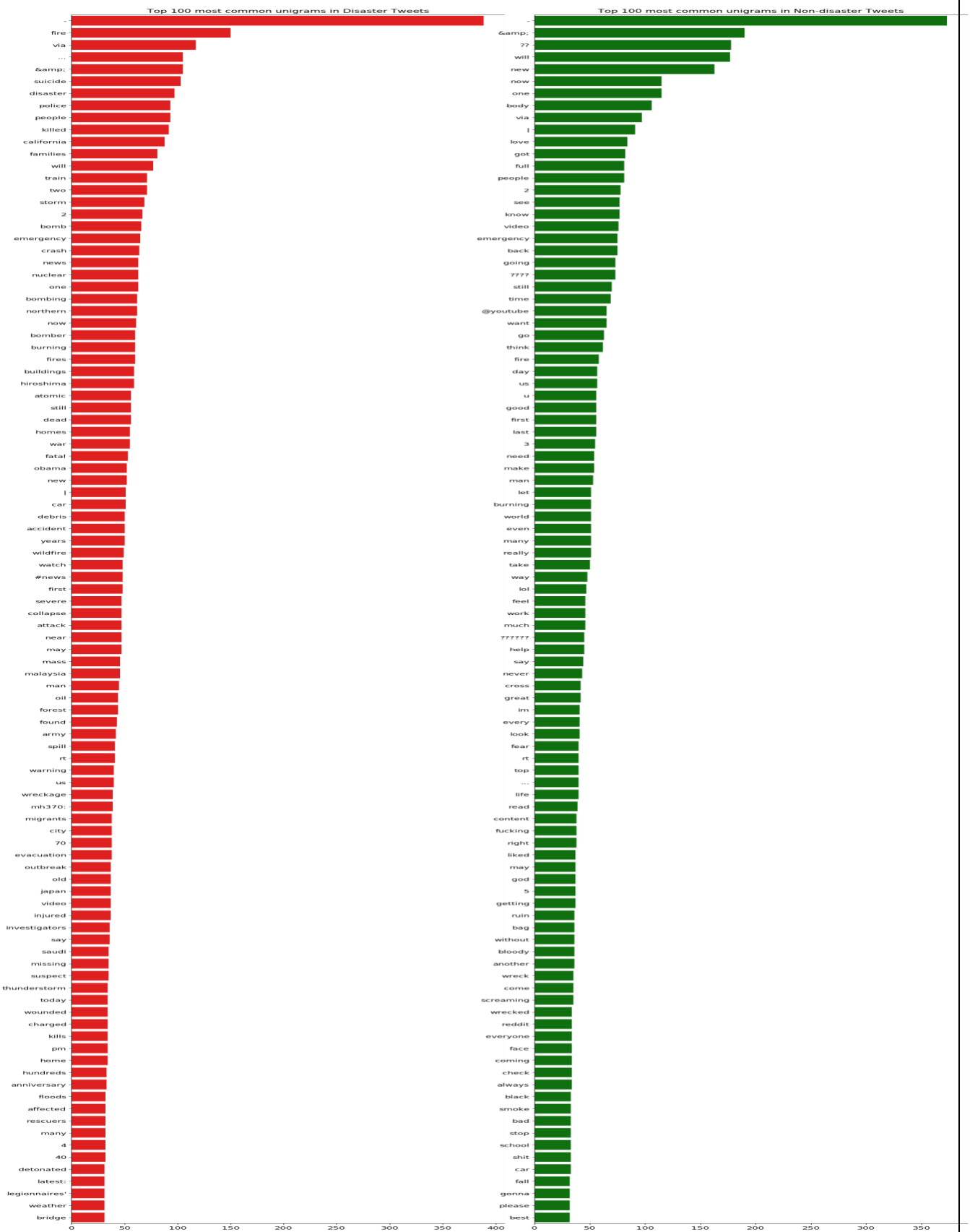
```
axes[1].tick_params(axis='x', labels=15)
```

```
axes[1].tick_params(axis='y', labels=15)
```

```
axes[0].set_title('Target Distribution in Training Set', fontsize=13)
```

```
axes[1].set_title('Target Count in Training Set', fontsize=13)
```

```
plt.show()
```



Code:

```

fig, axes = plt.subplots(ncols=2, figsize=(18, 50), dpi=100)
plt.tight_layout()

sns.barplot(y=df_disaster_unigrams[0].values[:N],
x=df_disaster_unigrams[1].values[:N], ax=axes[0], color='red')
sns.barplot(y=df_nondisaster_unigrams[0].values[:N],
x=df_nondisaster_unigrams[1].values[:N], ax=axes[1], color='green')

for i in range(2):
    axes[i].spines['right'].set_visible(False)
    axes[i].set_xlabel("")
    axes[i].set_ylabel("")
    axes[i].tick_params(axis='x', labelsize=13)
    axes[i].tick_params(axis='y', labelsize=13)

axes[0].set_title(f'Top {N} most common unigrams in Disaster Tweets', fontsize=15)
axes[1].set_title(f'Top {N} most common unigrams in Non-disaster Tweets', fontsize=15)

plt.show()

```

Description:

The plot illustrates the top 100 most common unigrams (single words) in disaster-related and non-disaster-related tweets. This visualization helps in understanding the textual patterns and common phrases used in different contexts, which is crucial for tasks like text classification and sentiment analysis.

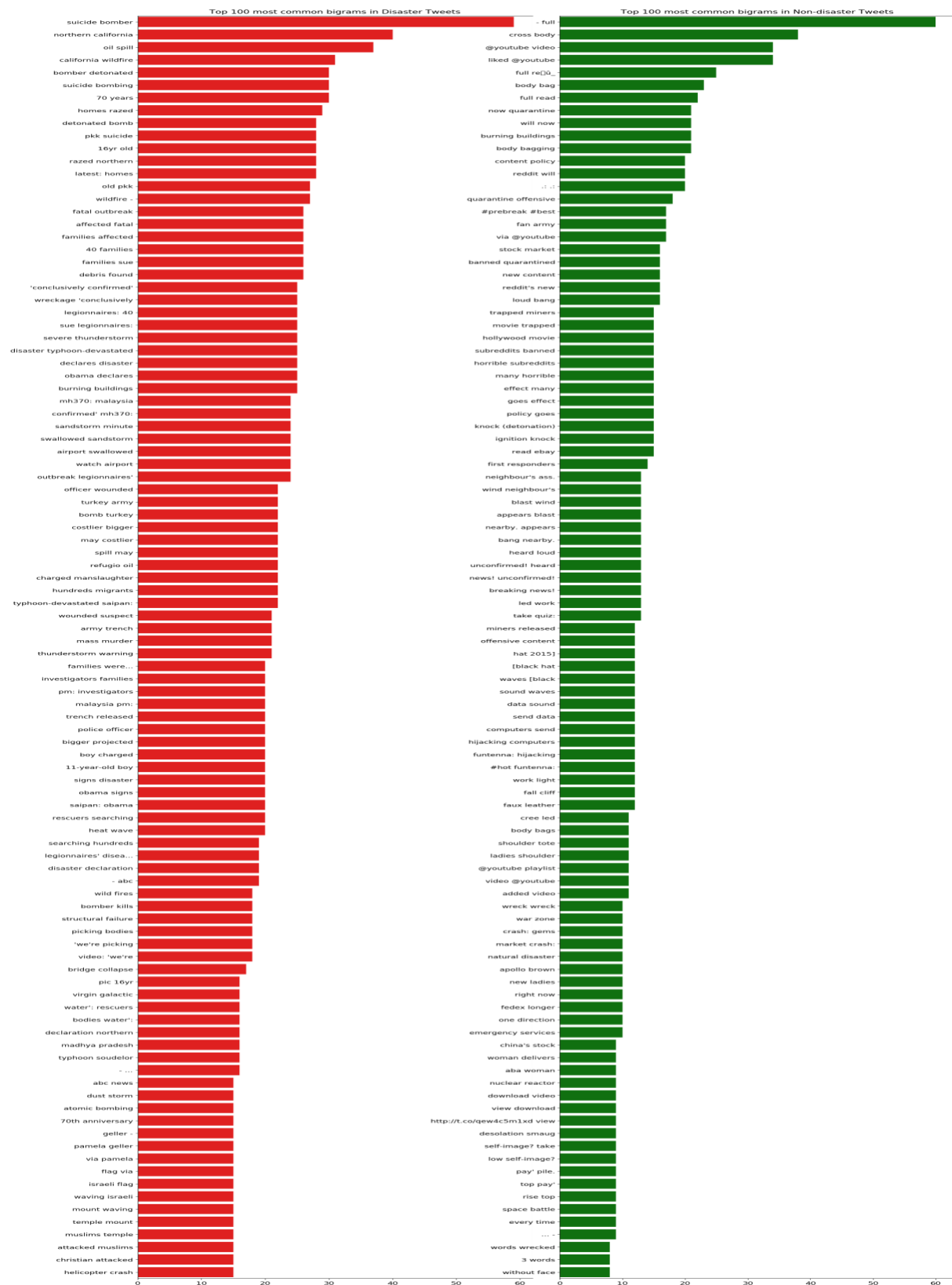
On the left side of the plot, represented by red bars, are the unigrams from disaster-related tweets. The most frequent unigrams in these tweets include terms directly associated with emergencies and disasters, such as "fire," "emergency," "flood," "storm," and "earthquake." These unigrams provide clear indications of the context in which they are used, often relating to specific events or actions taken during a disaster. The presence of such unigrams highlights the nature of the content in disaster-related tweets, which typically includes

information about ongoing emergencies, warnings, and responses from authorities. The high frequency of these terms suggests that disaster-related tweets are focused on conveying urgent information and updates about specific incidents.

On the right side of the plot, represented by green bars, are the unigrams from non-disaster-related tweets. The unigrams in these tweets are more general and less specific to emergencies. Examples of frequent unigrams in this category include "love," "good," "new," "video," and "morning." These unigrams are indicative of everyday conversations and social interactions on Twitter, reflecting a wide range of topics that are not related to disasters. The contrast between the unigrams in the two categories underscores the differences in the content and context of disaster-related versus non-disaster-related tweets. Non-disaster-related tweets tend to focus on personal experiences, opinions, and general social media interactions.

Overall, this plot provides valuable insights into the textual characteristics of disaster-related and non-disaster-related tweets. By identifying the most common unigrams, we can better understand the language and context used in these tweets, which can aid in the development of more accurate and effective models for classifying and analyzing social media data related to disasters. The clear distinction between the unigrams in disaster-related and non-disaster-related tweets highlights the potential for using these textual features in machine learning models to improve the accuracy of tweet classification.

Biagram:



fig, axes = plt.subplots(ncols=2, figsize=(18, 50), dpi=100)


```
plt.tight_layout()

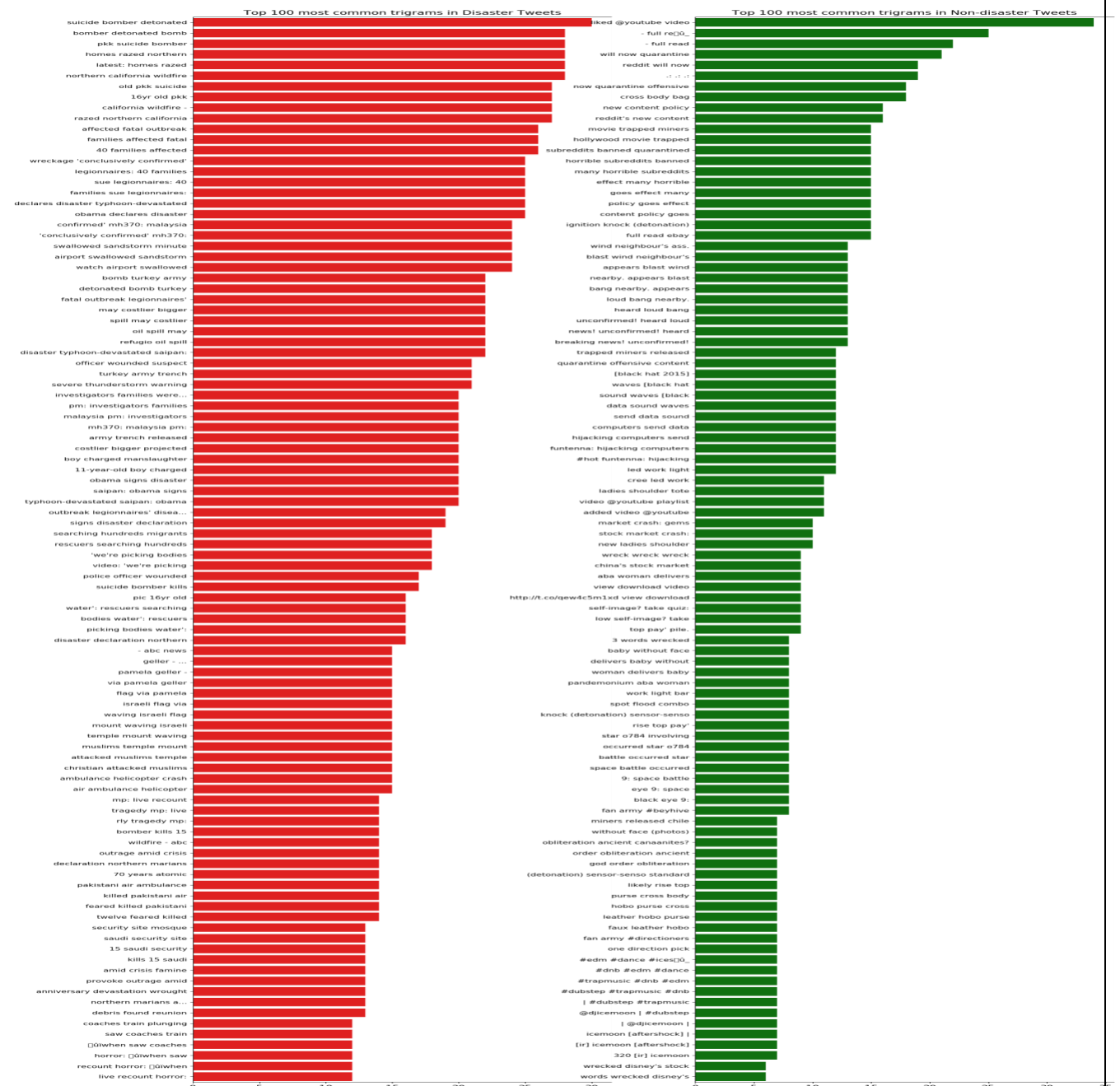
sns.barplot(y=df_disaster_bigrams[0].values[:N], x=df_disaster_bigrams[1].values[:N],
ax=axes[0], color='red')
sns.barplot(y=df_nondisaster_bigrams[0].values[:N],
x=df_nondisaster_bigrams[1].values[:N], ax=axes[1], color='green')

for i in range(2):
    axes[i].spines['right'].set_visible(False)
    axes[i].set_xlabel("")
    axes[i].set_ylabel("")
    axes[i].tick_params(axis='x', labelsize=13)
    axes[i].tick_params(axis='y', labelsize=13)

axes[0].set_title(f'Top {N} most common bigrams in Disaster Tweets', fontsize=15)
axes[1].set_title(f'Top {N} most common bigrams in Non-disaster Tweets', fontsize=15)

plt.show()
```

Trigram:



fig, axes = plt.subplots(ncols=2, figsize=(20, 50), dpi=100)

sns.barplot(y=df_disaster_trigrams[0].values[:N], x=df_disaster_trigrams[1].values[:N],
ax=axes[0], color='red')

sns.barplot(y=df_nondisaster_trigrams[0].values[:N],
x=df_nondisaster_trigrams[1].values[:N], ax=axes[1], color='green')

for i in range(2):

```
axes[i].spines['right'].set_visible(False)
axes[i].set_xlabel("")
axes[i].set_ylabel("")
axes[i].tick_params(axis='x', labelsiz=13)
axes[i].tick_params(axis='y', labelsiz=11)

axes[0].set_title(f'Top {N} most common trigrams in Disaster Tweets', fontsize=15)
axes[1].set_title(f'Top {N} most common trigrams in Non-disaster Tweets', fontsize=15)

plt.show()
```

Embeddings and Text Cleaning:

```
GloVe Embeddings cover 52.06% of vocabulary and 82.68% of text in Training Set
GloVe Embeddings cover 57.21% of vocabulary and 81.85% of text in Test Set
FastText Embeddings cover 51.52% of vocabulary and 81.84% of text in Training Set
FastText Embeddings cover 56.55% of vocabulary and 81.12% of text in Test Set
```

```
def build_vocab(X):

    tweets = X.apply(lambda s: s.split()).values
    vocab = {}

    for tweet in tweets:
        for word in tweet:
            try:
                vocab[word] += 1
            except KeyError:
                vocab[word] = 1
    return vocab

def check_embeddings_coverage(X, embeddings):

    vocab = build_vocab(X)
```

```

covered = {}
oov = {}
n_covered = 0
n_oov = 0

for word in vocab:
    try:
        covered[word] = embeddings[word]
        n_covered += vocab[word]
    except:
        oov[word] = vocab[word]
        n_oov += vocab[word]

vocab_coverage = len(covered) / len(vocab)
text_coverage = (n_covered / (n_covered + n_oov))

sorted_oov = sorted(oov.items(), key=operator.itemgetter(1))[::-1]
return sorted_oov, vocab_coverage, text_coverage

```

```

train_glove_oov,    train_glove_vocab_coverage,    train_glove_text_coverage    =
check_embeddings_coverage(df_train['text'], glove_embeddings)
test_glove_oov,    test_glove_vocab_coverage,    test_glove_text_coverage    =
check_embeddings_coverage(df_test['text'], glove_embeddings)
print('GloVe Embeddings cover {:.2%} of vocabulary and {:.2%} of text in Training
Set'.format(train_glove_vocab_coverage, train_glove_text_coverage))
print('GloVe Embeddings cover {:.2%} of vocabulary and {:.2%} of text in Test
Set'.format(test_glove_vocab_coverage, test_glove_text_coverage))

```

```

train_fasttext_oov,    train_fasttext_vocab_coverage,    train_fasttext_text_coverage    =
check_embeddings_coverage(df_train['text'], fasttext_embeddings)
test_fasttext_oov,    test_fasttext_vocab_coverage,    test_fasttext_text_coverage    =
check_embeddings_coverage(df_test['text'], fasttext_embeddings)

```

```
print('FastText Embeddings cover {:.2%} of vocabulary and {:.2%} of text in Training
Set'.format(train_fasttext_vocab_coverage, train_fasttext_text_coverage))
print('FastText Embeddings cover {:.2%} of vocabulary and {:.2%} of text in Test
Set'.format(test_fasttext_vocab_coverage, test_fasttext_text_coverage))
```

```
GloVe Embeddings cover 82.89% of vocabulary and 97.14% of text in Training Set
GloVe Embeddings cover 88.09% of vocabulary and 97.32% of text in Test Set
FastText Embeddings cover 82.88% of vocabulary and 97.12% of text in Training Set
FastText Embeddings cover 87.80% of vocabulary and 97.25% of text in Test Set
CPU times: user 8min 27s, sys: 211 ms, total: 8min 28s
Wall time: 8min 30s
Parser : 156 ms
```

```
def clean(tweet):
```

```
    # Special characters
```

```
    tweet = re.sub(r"\x89\u", "", tweet)
```

```
    tweet = re.sub(r"\x89\u0", "", tweet)
```

```
    tweet = re.sub(r"\x89\u0", "", tweet)
```

```
    tweet = re.sub(r"\x89\uIWhen", "When", tweet)
```

```
    tweet = re.sub(r"\x89\uI", "", tweet)
```

```
    tweet = re.sub(r"China\x89\u's", "China's", tweet)
```

```
    tweet = re.sub(r"let\x89\u's", "let's", tweet)
```

```
    tweet = re.sub(r"\x89\u÷", "", tweet)
```

```
    tweet = re.sub(r"\x89\u^a", "", tweet)
```

```
    tweet = re.sub(r"\x89\u\x9d", "", tweet)
```

```
    tweet = re.sub(r"â", "", tweet)
```

```
    tweet = re.sub(r"\x89\uç", "", tweet)
```

```
    tweet = re.sub(r"\x89\uçâ", "", tweet)
```

```
    tweet = re.sub(r"fromâwounds", "from wounds", tweet)
```

```
    tweet = re.sub(r"â", "", tweet)
```

```
    tweet = re.sub(r"â", "", tweet)
```

```
    tweet = re.sub(r"JapI_n", "Japan", tweet)
```

```
    tweet = re.sub(r"İ©", "e", tweet)
```

```
    tweet = re.sub(r"â", "", tweet)
```

```
    tweet = re.sub(r"Suruİ", "Suruc", tweet)
```

```
    tweet = re.sub(r"âÇ", "", tweet)
```

```
tweet = re.sub(r"£3million", "3 million", tweet)
```

```
tweet = re.sub(r"À", "", tweet)
```

Contractions

```
tweet = re.sub(r"he's", "he is", tweet)
```

```
tweet = re.sub(r"there's", "there is", tweet)
```

```
tweet = re.sub(r"We're", "We are", tweet)
```

```
tweet = re.sub(r"That's", "That is", tweet)
```

```
tweet = re.sub(r"won't", "will not", tweet)
```

```
tweet = re.sub(r"they're", "they are", tweet)
```

```
tweet = re.sub(r"Can't", "Cannot", tweet)
```

```
tweet = re.sub(r"wasn't", "was not", tweet)
```

```
tweet = re.sub(r"don't", "do not", tweet)
```

```
tweet = re.sub(r"aren't", "are not", tweet)
```

```
tweet = re.sub(r"isn't", "is not", tweet)
```

```
tweet = re.sub(r"What's", "What is", tweet)
```

```
tweet = re.sub(r"haven't", "have not", tweet)
```

```
tweet = re.sub(r"hasn't", "has not", tweet)
```

```
tweet = re.sub(r"There's", "There is", tweet)
```

```
tweet = re.sub(r"He's", "He is", tweet)
```

```
tweet = re.sub(r"It's", "It is", tweet)
```

```
tweet = re.sub(r"You're", "You are", tweet)
```

```
tweet = re.sub(r"I'M", "I am", tweet)
```

```
tweet = re.sub(r"shouldn't", "should not", tweet)
```

```
tweet = re.sub(r"wouldn't", "would not", tweet)
```

```
tweet = re.sub(r'i'm", "I am", tweet)
```

```
tweet = re.sub(r"I'm", "I am", tweet)
```

```
tweet = re.sub(r"I'm", "I am", tweet)
```

```
tweet = re.sub(r"Isn't", "is not", tweet)
```

```
tweet = re.sub(r"Here's", "Here is", tweet)
```

```
tweet = re.sub(r"you've", "you have", tweet)
```

```
tweet = re.sub(r"you've", "you have", tweet)
```

```
tweet = re.sub(r"we're", "we are", tweet)
```

```
tweet = re.sub(r"what's", "what is", tweet)
```

```
tweet = re.sub(r"couldn't", "could not", tweet)
tweet = re.sub(r"we've", "we have", tweet)
tweet = re.sub(r"it's", "it is", tweet)
tweet = re.sub(r"doesn't", "does not", tweet)
tweet = re.sub(r"It's", "It is", tweet)
tweet = re.sub(r"Here's", "Here is", tweet)
tweet = re.sub(r"who's", "who is", tweet)
tweet = re.sub(r"I've", "I have", tweet)
tweet = re.sub(r"y'all", "you all", tweet)
tweet = re.sub(r"can't", "cannot", tweet)
tweet = re.sub(r"would've", "would have", tweet)
tweet = re.sub(r"it'll", "it will", tweet)
tweet = re.sub(r"we'll", "we will", tweet)
tweet = re.sub(r"wouldn't", "would not", tweet)
tweet = re.sub(r"We've", "We have", tweet)
tweet = re.sub(r"he'll", "he will", tweet)
tweet = re.sub(r"Y'all", "You all", tweet)
tweet = re.sub(r"Weren't", "Were not", tweet)
tweet = re.sub(r"Didn't", "Did not", tweet)
tweet = re.sub(r"they'll", "they will", tweet)
tweet = re.sub(r"they'd", "they would", tweet)
tweet = re.sub(r"DON'T", "DO NOT", tweet)
tweet = re.sub(r"That's", "That is", tweet)
tweet = re.sub(r"they've", "they have", tweet)
tweet = re.sub(r"i'd", "I would", tweet)
tweet = re.sub(r"should've", "should have", tweet)
tweet = re.sub(r"You're", "You are", tweet)
tweet = re.sub(r"where's", "where is", tweet)
tweet = re.sub(r"Don't", "Do not", tweet)
tweet = re.sub(r"we'd", "we would", tweet)
tweet = re.sub(r"i'll", "I will", tweet)
tweet = re.sub(r"weren't", "were not", tweet)
tweet = re.sub(r"They're", "They are", tweet)
tweet = re.sub(r"Can't", "Cannot", tweet)
```

```

tweet = re.sub(r"you\x89\u2011", "you will", tweet)
tweet = re.sub(r"I\x89\u201d", "I would", tweet)
tweet = re.sub(r"let's", "let us", tweet)
tweet = re.sub(r"it's", "it is", tweet)
tweet = re.sub(r"can't", "cannot", tweet)
tweet = re.sub(r"don't", "do not", tweet)
tweet = re.sub(r"you're", "you are", tweet)
tweet = re.sub(r"i've", "I have", tweet)
tweet = re.sub(r"that's", "that is", tweet)
tweet = re.sub(r"i'll", "I will", tweet)
tweet = re.sub(r"doesn't", "does not", tweet)
tweet = re.sub(r"i'd", "I would", tweet)
tweet = re.sub(r"didn't", "did not", tweet)
tweet = re.sub(r"ain't", "am not", tweet)
tweet = re.sub(r"you'll", "you will", tweet)
tweet = re.sub(r"I've", "I have", tweet)
tweet = re.sub(r"Don't", "do not", tweet)
tweet = re.sub(r"I'll", "I will", tweet)
tweet = re.sub(r"I'd", "I would", tweet)
tweet = re.sub(r"Let's", "Let us", tweet)
tweet = re.sub(r"you'd", "You would", tweet)
tweet = re.sub(r"It's", "It is", tweet)
tweet = re.sub(r"Ain't", "am not", tweet)
tweet = re.sub(r"Haven't", "Have not", tweet)
tweet = re.sub(r"Could've", "Could have", tweet)
tweet = re.sub(r"youve", "you have", tweet)
tweet = re.sub(r"donât", "do not", tweet)

```

Character entity references

```

tweet = re.sub(r"&gt;", ">", tweet)
tweet = re.sub(r"&lt;", "<", tweet)
tweet = re.sub(r"&amp;", "&", tweet)

```

Typos, slang and informal abbreviations


```
tweet = re.sub(r"w/e", "whatever", tweet)
tweet = re.sub(r"w/", "with", tweet)
tweet = re.sub(r"USAgov", "USA government", tweet)
tweet = re.sub(r"recentlu", "recently", tweet)
tweet = re.sub(r"Ph0tos", "Photos", tweet)
tweet = re.sub(r"amirite", "am I right", tweet)
tweet = re.sub(r"exp0sed", "exposed", tweet)
tweet = re.sub(r"<3", "love", tweet)
tweet = re.sub(r"amageddon", "armageddon", tweet)
tweet = re.sub(r"Trfc", "Traffic", tweet)
tweet = re.sub(r"8/5/2015", "2015-08-05", tweet)
tweet = re.sub(r"WindStorm", "Wind Storm", tweet)
tweet = re.sub(r"8/6/2015", "2015-08-06", tweet)
tweet = re.sub(r"10:38PM", "10:38 PM", tweet)
tweet = re.sub(r"10:30pm", "10:30 PM", tweet)
tweet = re.sub(r"16yr", "16 year", tweet)
tweet = re.sub(r"lmao", "laughing my ass off", tweet)
tweet = re.sub(r"TRAUMATISED", "traumatized", tweet)
```

Hashtags and usernames

```
tweet = re.sub(r"IranDeal", "Iran Deal", tweet)
tweet = re.sub(r"ArianaGrande", "Ariana Grande", tweet)
tweet = re.sub(r"camilacabello97", "camila cabello", tweet)
tweet = re.sub(r"RondaRousey", "Ronda Rousey", tweet)
tweet = re.sub(r"MTVHottest", "MTV Hottest", tweet)
tweet = re.sub(r"TrapMusic", "Trap Music", tweet)
tweet = re.sub(r"ProphetMuhammad", "Prophet Muhammad", tweet)
tweet = re.sub(r"PantherAttack", "Panther Attack", tweet)
tweet = re.sub(r"StrategicPatience", "Strategic Patience", tweet)
tweet = re.sub(r"socialnews", "social news", tweet)
tweet = re.sub(r"NASAHurricane", "NASA Hurricane", tweet)
tweet = re.sub(r"onlinecommunities", "online communities", tweet)
tweet = re.sub(r"humanconsumption", "human consumption", tweet)
tweet = re.sub(r"Typhoon-Devastated", "Typhoon Devastated", tweet)
```

```
tweet = re.sub(r"Meat-Loving", "Meat Loving", tweet)
tweet = re.sub(r"facialabuse", "facial abuse", tweet)
tweet = re.sub(r"LakeCounty", "Lake County", tweet)
tweet = re.sub(r"BeingAuthor", "Being Author", tweet)
tweet = re.sub(r"withheavenly", "with heavenly", tweet)
tweet = re.sub(r"thankU", "thank you", tweet)
tweet = re.sub(r"iTunesMusic", "iTunes Music", tweet)
tweet = re.sub(r"OffensiveContent", "Offensive Content", tweet)
tweet = re.sub(r"WorstSummerJob", "Worst Summer Job", tweet)
tweet = re.sub(r"HarryBeCareful", "Harry Be Careful", tweet)
tweet = re.sub(r"NASASolarSystem", "NASA Solar System", tweet)
tweet = re.sub(r"animalrescue", "animal rescue", tweet)
tweet = re.sub(r"KurtSchlichter", "Kurt Schlichter", tweet)
tweet = re.sub(r"aRmageddon", "armageddon", tweet)
tweet = re.sub(r"Throwingknives", "Throwing knives", tweet)
tweet = re.sub(r"GodsLove", "God's Love", tweet)
tweet = re.sub(r"bookboost", "book boost", tweet)
tweet = re.sub(r"ibooklove", "I book love", tweet)
tweet = re.sub(r"NestleIndia", "Nestle India", tweet)
tweet = re.sub(r"realDonaldTrump", "Donald Trump", tweet)
tweet = re.sub(r"DavidVonderhaar", "David Vonderhaar", tweet)
tweet = re.sub(r"CecilTheLion", "Cecil The Lion", tweet)
tweet = re.sub(r"weathernetwork", "weather network", tweet)
tweet = re.sub(r"withBioterrorism&use", "with Bioterrorism & use", tweet)
tweet = re.sub(r"Hostage&2", "Hostage & 2", tweet)
tweet = re.sub(r"GOPDebate", "GOP Debate", tweet)
tweet = re.sub(r"RickPerry", "Rick Perry", tweet)
tweet = re.sub(r"frontpage", "front page", tweet)
tweet = re.sub(r"NewsInTweets", "News In Tweets", tweet)
tweet = re.sub(r"ViralSpell", "Viral Spell", tweet)
tweet = re.sub(r"til_now", "until now", tweet)
tweet = re.sub(r"volcanoinRussia", "volcano in Russia", tweet)
tweet = re.sub(r"ZippedNews", "Zipped News", tweet)
tweet = re.sub(r"MicheleBachman", "Michele Bachman", tweet)
```

```
tweet = re.sub(r"53inch", "53 inch", tweet)
tweet = re.sub(r"KerrickTrial", "Kerrick Trial", tweet)
tweet = re.sub(r"abstorm", "Alberta Storm", tweet)
tweet = re.sub(r"Beyhive", "Beyonce hive", tweet)
tweet = re.sub(r>IDFire", "Idaho Fire", tweet)
tweet = re.sub(r"DETECTADO", "Detected", tweet)
tweet = re.sub(r"RockyFire", "Rocky Fire", tweet)
tweet = re.sub(r"Listen/Buy", "Listen / Buy", tweet)
tweet = re.sub(r"NickCannon", "Nick Cannon", tweet)
tweet = re.sub(r"FaroeIslands", "Faroe Islands", tweet)
tweet = re.sub(r"yycstorm", "Calgary Storm", tweet)
tweet = re.sub(r"IDPs:", "Internally Displaced People :", tweet)
tweet = re.sub(r"ArtistsUnited", "Artists United", tweet)
tweet = re.sub(r"ClaytonBryant", "Clayton Bryant", tweet)
tweet = re.sub(r"jimmyfallon", "jimmy fallon", tweet)
tweet = re.sub(r"justinbieber", "justin bieber", tweet)
tweet = re.sub(r"UTC2015", "UTC 2015", tweet)
tweet = re.sub(r"Time2015", "Time 2015", tweet)
tweet = re.sub(r"djicemoon", "dj icemoon", tweet)
tweet = re.sub(r"LivingSafely", "Living Safely", tweet)
tweet = re.sub(r"FIFA16", "Fifa 2016", tweet)
tweet = re.sub(r"thisiswhywecanthavenicethings", "this is why we cannot have nice
things", tweet)
tweet = re.sub(r"bbcnews", "bbc news", tweet)
tweet = re.sub(r"UndergroundRailraod", "Underground Railraod", tweet)
tweet = re.sub(r"c4news", "c4 news", tweet)
tweet = re.sub(r"OBLITERATION", "obliteration", tweet)
tweet = re.sub(r"MUDSLIDE", "mudslide", tweet)
tweet = re.sub(r"NoSurrender", "No Surrender", tweet)
tweet = re.sub(r"NotExplained", "Not Explained", tweet)
tweet = re.sub(r"greatbritishbakeoff", "great british bake off", tweet)
tweet = re.sub(r"LondonFire", "London Fire", tweet)
tweet = re.sub(r"KOTAWeather", "KOTA Weather", tweet)
tweet = re.sub(r"LuchaUnderground", "Lucha Underground", tweet)
```

```
tweet = re.sub(r"KOIN6News", "KOIN 6 News", tweet)
tweet = re.sub(r"LiveOnK2", "Live On K2", tweet)
tweet = re.sub(r"9NewsGoldCoast", "9 News Gold Coast", tweet)
tweet = re.sub(r"nikeplus", "nike plus", tweet)
tweet = re.sub(r"david_cameron", "David Cameron", tweet)
tweet = re.sub(r"peterjukes", "Peter Jukes", tweet)
tweet = re.sub(r"JamesMelville", "James Melville", tweet)
tweet = re.sub(r"megynkelly", "Megyn Kelly", tweet)
tweet = re.sub(r"cnewslive", "C News Live", tweet)
tweet = re.sub(r"JamaicaObserver", "Jamaica Observer", tweet)
tweet = re.sub(r"TweetLikeItsSeptember11th2001", "Tweet like it is september 11th
2001", tweet)
tweet = re.sub(r"cbplawyers", "cbp lawyers", tweet)
tweet = re.sub(r"fewmoretweets", "few more tweets", tweet)
tweet = re.sub(r"BlackLivesMatter", "Black Lives Matter", tweet)
tweet = re.sub(r"cjoyner", "Chris Joyner", tweet)
tweet = re.sub(r"ENGvAUS", "England vs Australia", tweet)
tweet = re.sub(r"ScottWalker", "Scott Walker", tweet)
tweet = re.sub(r"MikeParrActor", "Michael Parr", tweet)
tweet = re.sub(r"4PlayThursdays", "Foreplay Thursdays", tweet)
tweet = re.sub(r"TGF2015", "Tontitown Grape Festival", tweet)
tweet = re.sub(r"realmandyrain", "Mandy Rain", tweet)
tweet = re.sub(r"GraysonDolan", "Grayson Dolan", tweet)
tweet = re.sub(r"ApolloBrown", "Apollo Brown", tweet)
tweet = re.sub(r"saddlebrooke", "Saddlebrooke", tweet)
tweet = re.sub(r"TontitownGrape", "Tontitown Grape", tweet)
tweet = re.sub(r"AbbsWinston", "Abbs Winston", tweet)
tweet = re.sub(r"ShaunKing", "Shaun King", tweet)
tweet = re.sub(r"MeekMill", "Meek Mill", tweet)
tweet = re.sub(r"TornadoGiveaway", "Tornado Giveaway", tweet)
tweet = re.sub(r"GRupdates", "GR updates", tweet)
tweet = re.sub(r"SouthDowns", "South Downs", tweet)
tweet = re.sub(r"braininjury", "brain injury", tweet)
tweet = re.sub(r"auspol", "Australian politics", tweet)
```

```
tweet = re.sub(r"PlannedParenthood", "Planned Parenthood", tweet)
tweet = re.sub(r"calgaryweather", "Calgary Weather", tweet)
tweet = re.sub(r"weallheartonedirection", "we all heart one direction", tweet)
tweet = re.sub(r"edsheeran", "Ed Sheeran", tweet)
tweet = re.sub(r"TrueHeroes", "True Heroes", tweet)
tweet = re.sub(r"S3XLEAK", "sex leak", tweet)
tweet = re.sub(r"ComplexMag", "Complex Magazine", tweet)
tweet = re.sub(r"TheAdvocateMag", "The Advocate Magazine", tweet)
tweet = re.sub(r"CityofCalgary", "City of Calgary", tweet)
tweet = re.sub(r"EbolaOutbreak", "Ebola Outbreak", tweet)
tweet = re.sub(r"SummerFate", "Summer Fate", tweet)
tweet = re.sub(r"RAmag", "Royal Academy Magazine", tweet)
tweet = re.sub(r"offers2go", "offers to go", tweet)
tweet = re.sub(r"foodscare", "food scare", tweet)
tweet = re.sub(r"MNPDNashville", "Metropolitan Nashville Police Department", tweet)
tweet = re.sub(r"TfLBusAlerts", "TfL Bus Alerts", tweet)
tweet = re.sub(r"GamerGate", "Gamer Gate", tweet)
tweet = re.sub(r"IHHen", "Humanitarian Relief", tweet)
tweet = re.sub(r"spinningbot", "spinning bot", tweet)
tweet = re.sub(r"ModiMinistry", "Modi Ministry", tweet)
tweet = re.sub(r"TAXIWAYS", "taxi ways", tweet)
tweet = re.sub(r"Calum5SOS", "Calum Hood", tweet)
tweet = re.sub(r"po_st", "po.st", tweet)
tweet = re.sub(r"scoopit", "scoop.it", tweet)
tweet = re.sub(r"UltimaLucha", "Ultima Lucha", tweet)
tweet = re.sub(r"JonathanFerrell", "Jonathan Ferrell", tweet)
tweet = re.sub(r"aria_ahrany", "Aria Ahrany", tweet)
tweet = re.sub(r"rapidcity", "Rapid City", tweet)
tweet = re.sub(r"OutBid", "outbid", tweet)
tweet = re.sub(r"lavenderpoetrycafe", "lavender poetry cafe", tweet)
tweet = re.sub(r"EudryLantiqua", "Eudry Lantiqua", tweet)
tweet = re.sub(r"15PM", "15 PM", tweet)
tweet = re.sub(r"OriginalFunko", "Funko", tweet)
tweet = re.sub(r"rightwaystan", "Richard Tan", tweet)
```

```
tweet = re.sub(r"CindyNoonan", "Cindy Noonan", tweet)
tweet = re.sub(r"RT_America", "RT America", tweet)
tweet = re.sub(r"narendramodi", "Narendra Modi", tweet)
tweet = re.sub(r"BakeOffFriends", "Bake Off Friends", tweet)
tweet = re.sub(r"TeamHendrick", "Hendrick Motorsports", tweet)
tweet = re.sub(r"alexbelloli", "Alex Belloli", tweet)
tweet = re.sub(r"itsjustinstuart", "Justin Stuart", tweet)
tweet = re.sub(r"gunsense", "gun sense", tweet)
tweet = re.sub(r"DebateQuestionsWeWantToHear", "debate questions we want to hear",
tweet)
tweet = re.sub(r"RoyalCarribean", "Royal Carribean", tweet)
tweet = re.sub(r"samanthaturne19", "Samantha Turner", tweet)
tweet = re.sub(r"JonVoyage", "Jon Stewart", tweet)
tweet = re.sub(r"renew911health", "renew 911 health", tweet)
tweet = re.sub(r"SuryaRay", "Surya Ray", tweet)
tweet = re.sub(r"pattonoswalt", "Patton Oswalt", tweet)
tweet = re.sub(r"minhazmerchant", "Minhaz Merchant", tweet)
tweet = re.sub(r"TLVFaces", "Israel Diaspora Coalition", tweet)
tweet = re.sub(r"pmarca", "Marc Andreessen", tweet)
tweet = re.sub(r"pdx911", "Portland Police", tweet)
tweet = re.sub(r"jamaicaplain", "Jamaica Plain", tweet)
tweet = re.sub(r"Japton", "Arkansas", tweet)
tweet = re.sub(r"RouteComplex", "Route Complex", tweet)
tweet = re.sub(r"INSubcontinent", "Indian Subcontinent", tweet)
tweet = re.sub(r"NJTurnpike", "New Jersey Turnpike", tweet)
tweet = re.sub(r"Politifiact", "PolitiFact", tweet)
tweet = re.sub(r"Hiroshima70", "Hiroshima", tweet)
tweet = re.sub(r"GMMBC", "Greater Mt Moriah Baptist Church", tweet)
tweet = re.sub(r"versethe", "verse the", tweet)
tweet = re.sub(r"TubeStrike", "Tube Strike", tweet)
tweet = re.sub(r"MissionHills", "Mission Hills", tweet)
tweet = re.sub(r"ProtectDenaliWolves", "Protect Denali Wolves", tweet)
tweet = re.sub(r"NANKANA", "Nankana", tweet)
tweet = re.sub(r"SAHIB", "Sahib", tweet)
```

```
tweet = re.sub(r"PAKPATTAN", "Pakpattan", tweet)
tweet = re.sub(r"Newz_Sacramento", "News Sacramento", tweet)
tweet = re.sub(r"gofundme", "go fund me", tweet)
tweet = re.sub(r"pmharper", "Stephen Harper", tweet)
tweet = re.sub(r"IvanBerroa", "Ivan Berroa", tweet)
tweet = re.sub(r"LosDelSonido", "Los Del Sonido", tweet)
tweet = re.sub(r"bandcodeseries", "banco de series", tweet)
tweet = re.sub(r"timkaine", "Tim Kaine", tweet)
tweet = re.sub(r"IdentityTheft", "Identity Theft", tweet)
tweet = re.sub(r"AllLivesMatter", "All Lives Matter", tweet)
tweet = re.sub(r"mishacollins", "Misha Collins", tweet)
tweet = re.sub(r"BillNeelyNBC", "Bill Neely", tweet)
tweet = re.sub(r"BeClearOnCancer", "be clear on cancer", tweet)
tweet = re.sub(r"Kowing", "Knowing", tweet)
tweet = re.sub(r"ScreamQueens", "Scream Queens", tweet)
tweet = re.sub(r"AskCharley", "Ask Charley", tweet)
tweet = re.sub(r"BlizzHeroes", "Heroes of the Storm", tweet)
tweet = re.sub(r"BradleyBrad47", "Bradley Brad", tweet)
tweet = re.sub(r"HannaPH", "Typhoon Hanna", tweet)
tweet = re.sub(r"meinlcymbals", "MEINL Cymbals", tweet)
tweet = re.sub(r"Ptbo", "Peterborough", tweet)
tweet = re.sub(r"cnnbrk", "CNN Breaking News", tweet)
tweet = re.sub(r"IndianNews", "Indian News", tweet)
tweet = re.sub(r"savebees", "save bees", tweet)
tweet = re.sub(r"GreenHarvard", "Green Harvard", tweet)
tweet = re.sub(r"StandwithPP", "Stand with planned parenthood", tweet)
tweet = re.sub(r"hermancranston", "Herman Cranston", tweet)
tweet = re.sub(r"WMUR9", "WMUR-TV", tweet)
tweet = re.sub(r"RockBottomRadFM", "Rock Bottom Radio", tweet)
tweet = re.sub(r"ameenshaikh3", "Ameen Shaikh", tweet)
tweet = re.sub(r"ProSyn", "Project Syndicate", tweet)
tweet = re.sub(r"Daesh", "ISIS", tweet)
tweet = re.sub(r"s2g", "swear to god", tweet)
tweet = re.sub(r"listenlive", "listen live", tweet)
```

```
tweet = re.sub(r"CDCgov", "Centers for Disease Control and Prevention", tweet)
tweet = re.sub(r"FoxNew", "Fox News", tweet)
tweet = re.sub(r"CBSBigBrother", "Big Brother", tweet)
tweet = re.sub(r"JulieDiCaro", "Julie DiCaro", tweet)
tweet = re.sub(r"theadvocatemag", "The Advocate Magazine", tweet)
tweet = re.sub(r"RohnertParkDPS", "Rohnert Park Police Department", tweet)
tweet = re.sub(r"THISIZBWRIGHT", "Bonnie Wright", tweet)
tweet = re.sub(r"Popularmmos", "Popular MMOs", tweet)
tweet = re.sub(r"WildHorses", "Wild Horses", tweet)
tweet = re.sub(r"FantasticFour", "Fantastic Four", tweet)
tweet = re.sub(r"HORNDALE", "Horndale", tweet)
tweet = re.sub(r"PINER", "Piner", tweet)
tweet = re.sub(r"BathAndNorthEastSomerset", "Bath and North East Somerset", tweet)
tweet = re.sub(r"thatswhatfriendsarefor", "that is what friends are for", tweet)
tweet = re.sub(r"residualincome", "residual income", tweet)
tweet = re.sub(r"YahooNewsDigest", "Yahoo News Digest", tweet)
tweet = re.sub(r"MalaysiaAirlines", "Malaysia Airlines", tweet)
tweet = re.sub(r"AmazonDeals", "Amazon Deals", tweet)
tweet = re.sub(r"MissCharleyWebb", "Charley Webb", tweet)
tweet = re.sub(r"shoalstraffic", "shoals traffic", tweet)
tweet = re.sub(r"GeorgeFoster72", "George Foster", tweet)
tweet = re.sub(r"pop2015", "pop 2015", tweet)
tweet = re.sub(r"_PokemonCards_", "Pokemon Cards", tweet)
tweet = re.sub(r"DianneG", "Dianne Gallagher", tweet)
tweet = re.sub(r"KashmirConflict", "Kashmir Conflict", tweet)
tweet = re.sub(r"BritishBakeOff", "British Bake Off", tweet)
tweet = re.sub(r"FreeKashmir", "Free Kashmir", tweet)
tweet = re.sub(r"matmosley", "Matt Mosley", tweet)
tweet = re.sub(r"BishopFred", "Bishop Fred", tweet)
tweet = re.sub(r"EndConflict", "End Conflict", tweet)
tweet = re.sub(r"EndOccupation", "End Occupation", tweet)
tweet = re.sub(r"UNHEALED", "unhealed", tweet)
tweet = re.sub(r"CharlesDagnall", "Charles Dagnall", tweet)
tweet = re.sub(r"Latestnews", "Latest news", tweet)
```



```
tweet = re.sub(r"KindleCountdown", "Kindle Countdown", tweet)
tweet = re.sub(r"NoMoreHandouts", "No More Handouts", tweet)
tweet = re.sub(r"datingtips", "dating tips", tweet)
tweet = re.sub(r"charlesadler", "Charles Adler", tweet)
tweet = re.sub(r"twia", "Texas Windstorm Insurance Association", tweet)
tweet = re.sub(r"txlege", "Texas Legislature", tweet)
tweet = re.sub(r"WindstormInsurer", "Windstorm Insurer", tweet)
tweet = re.sub(r"Newss", "News", tweet)
tweet = re.sub(r"hempoil", "hemp oil", tweet)
tweet = re.sub(r"CommoditiesAre", "Commodities are", tweet)
tweet = re.sub(r"tubestrike", "tube strike", tweet)
tweet = re.sub(r"JoeNBC", "Joe Scarborough", tweet)
tweet = re.sub(r"LiteraryCakes", "Literary Cakes", tweet)
tweet = re.sub(r"TI5", "The International 5", tweet)
tweet = re.sub(r"thehill", "the hill", tweet)
tweet = re.sub(r"3others", "3 others", tweet)
tweet = re.sub(r"stighefootball", "Sam Tighe", tweet)
tweet = re.sub(r"whatstheimportantvideo", "what is the important video", tweet)
tweet = re.sub(r"ClaudioMeloni", "Claudio Meloni", tweet)
tweet = re.sub(r"DukeSkywalker", "Duke Skywalker", tweet)
tweet = re.sub(r"carsonmwr", "Fort Carson", tweet)
tweet = re.sub(r"offdishduty", "off dish duty", tweet)
tweet = re.sub(r"andword", "and word", tweet)
tweet = re.sub(r"rhodeisland", "Rhode Island", tweet)
tweet = re.sub(r"easternoregon", "Eastern Oregon", tweet)
tweet = re.sub(r"WAwildfire", "Washington Wildfire", tweet)
tweet = re.sub(r"fingerrockfire", "Finger Rock Fire", tweet)
tweet = re.sub(r"57am", "57 am", tweet)
tweet = re.sub(r"fingerrockfire", "Finger Rock Fire", tweet)
tweet = re.sub(r"JacobHoggard", "Jacob Hoggard", tweet)
tweet = re.sub(r"newnewnew", "new new new", tweet)
tweet = re.sub(r"under50", "under 50", tweet)
tweet = re.sub(r"getitbeforeitsgone", "get it before it is gone", tweet)
tweet = re.sub(r"freshoutofthebox", "fresh out of the box", tweet)
```

```
tweet = re.sub(r"amwriting", "am writing", tweet)
tweet = re.sub(r"Bokoharm", "Boko Haram", tweet)
tweet = re.sub(r"Nowlike", "Now like", tweet)
tweet = re.sub(r"seasonfrom", "season from", tweet)
tweet = re.sub(r"epicente", "epicenter", tweet)
tweet = re.sub(r"epicenterr", "epicenter", tweet)
tweet = re.sub(r"sicklife", "sick life", tweet)
tweet = re.sub(r"yycweather", "Calgary Weather", tweet)
tweet = re.sub(r"calgarysun", "Calgary Sun", tweet)
tweet = re.sub(r"approachng", "approaching", tweet)
tweet = re.sub(r"evng", "evening", tweet)
tweet = re.sub(r"Sumthng", "something", tweet)
tweet = re.sub(r"EllenPompeo", "Ellen Pompeo", tweet)
tweet = re.sub(r"shondarhimes", "Shonda Rhimes", tweet)
tweet = re.sub(r"ABCNetwork", "ABC Network", tweet)
tweet = re.sub(r"SushmaSwaraj", "Sushma Swaraj", tweet)
tweet = re.sub(r"pray4japan", "Pray for Japan", tweet)
tweet = re.sub(r"hope4japan", "Hope for Japan", tweet)
tweet = re.sub(r"Illusionimages", "Illusion images", tweet)
tweet = re.sub(r"SummerUnderTheStars", "Summer Under The Stars", tweet)
tweet = re.sub(r"ShallWeDance", "Shall We Dance", tweet)
tweet = re.sub(r"TCMParty", "TCM Party", tweet)
tweet = re.sub(r"marijuananews", "marijuana news", tweet)
tweet = re.sub(r"onbeingwithKristaTippett", "on being with Krista Tippett", tweet)
tweet = re.sub(r"Beingtweets", "Being tweets", tweet)
tweet = re.sub(r"newauthors", "new authors", tweet)
tweet = re.sub(r"remedyyyy", "remedy", tweet)
tweet = re.sub(r"44PM", "44 PM", tweet)
tweet = re.sub(r"HeadlinesApp", "Headlines App", tweet)
tweet = re.sub(r"40PM", "40 PM", tweet)
tweet = re.sub(r"myswc", "Severe Weather Center", tweet)
tweet = re.sub(r"ithats", "that is", tweet)
tweet = re.sub(r"icouldsitinthismomentforever", "I could sit in this moment forever",
tweet)
```

```
tweet = re.sub(r"FatLoss", "Fat Loss", tweet)
tweet = re.sub(r"02PM", "02 PM", tweet)
tweet = re.sub(r"MetroFmTalk", "Metro Fm Talk", tweet)
tweet = re.sub(r"Bstrd", "bastard", tweet)
tweet = re.sub(r"bldy", "bloody", tweet)
tweet = re.sub(r"MetrofmTalk", "Metro Fm Talk", tweet)
tweet = re.sub(r"terrorismturn", "terrorism turn", tweet)
tweet = re.sub(r"BBCNewsAsia", "BBC News Asia", tweet)
tweet = re.sub(r"BehindTheScenes", "Behind The Scenes", tweet)
tweet = re.sub(r"GeorgeTakei", "George Takei", tweet)
tweet = re.sub(r"WomensWeeklyMag", "Womens Weekly Magazine", tweet)
tweet = re.sub(r"SurvivorsGuidetoEarth", "Survivors Guide to Earth", tweet)
tweet = re.sub(r"incubusband", "incubus band", tweet)
tweet = re.sub(r"Babypicturethis", "Baby picture this", tweet)
tweet = re.sub(r"BombEffects", "Bomb Effects", tweet)
tweet = re.sub(r"win10", "Windows 10", tweet)
tweet = re.sub(r"idkidk", "I do not know I do not know", tweet)
tweet = re.sub(r"TheWalkingDead", "The Walking Dead", tweet)
tweet = re.sub(r"amyschumer", "Amy Schumer", tweet)
tweet = re.sub(r"crewlist", "crew list", tweet)
tweet = re.sub(r"Erdogans", "Erdogan", tweet)
tweet = re.sub(r"BBCLive", "BBC Live", tweet)
tweet = re.sub(r"TonyAbbottMHR", "Tony Abbott", tweet)
tweet = re.sub(r"paulmyerscough", "Paul Myerscough", tweet)
tweet = re.sub(r"georgegallagher", "George Gallagher", tweet)
tweet = re.sub(r"JimmieJohnson", "Jimmie Johnson", tweet)
tweet = re.sub(r"pctool", "pc tool", tweet)
tweet = re.sub(r"DoingHashtagsRight", "Doing Hashtags Right", tweet)
tweet = re.sub(r"ThrowbackThursday", "Throwback Thursday", tweet)
tweet = re.sub(r"SnowBackSunday", "Snowback Sunday", tweet)
tweet = re.sub(r"LakeEffect", "Lake Effect", tweet)
tweet = re.sub(r"RTphotographyUK", "Richard Thomas Photography UK", tweet)
tweet = re.sub(r"BigBang_CBS", "Big Bang CBS", tweet)
tweet = re.sub(r"writerslife", "writers life", tweet)
```

```
tweet = re.sub(r"NaturalBirth", "Natural Birth", tweet)
tweet = re.sub(r"UnusualWords", "Unusual Words", tweet)
tweet = re.sub(r"wizkhalifa", "Wiz Khalifa", tweet)
tweet = re.sub(r"acreativedc", "a creative DC", tweet)
tweet = re.sub(r"vscode", "vsco DC", tweet)
tweet = re.sub(r"VSCOcam", "vsco camera", tweet)
tweet = re.sub(r"TheBEACHDC", "The beach DC", tweet)
tweet = re.sub(r"buildingmuseum", "building museum", tweet)
tweet = re.sub(r"WorldOil", "World Oil", tweet)
tweet = re.sub(r"redwedding", "red wedding", tweet)
tweet = re.sub(r"AmazingRaceCanada", "Amazing Race Canada", tweet)
tweet = re.sub(r"WakeUpAmerica", "Wake Up America", tweet)
tweet = re.sub(r"\\Allahuakbar\\", "Allahu Akbar", tweet)
tweet = re.sub(r"bleased", "blessed", tweet)
tweet = re.sub(r"nigeriantribune", "Nigerian Tribune", tweet)
tweet = re.sub(r"HIDEO_KOJIMA_EN", "Hideo Kojima", tweet)
tweet = re.sub(r"FusionFestival", "Fusion Festival", tweet)
tweet = re.sub(r"50Mixed", "50 Mixed", tweet)
tweet = re.sub(r"NoAgenda", "No Agenda", tweet)
tweet = re.sub(r"WhiteGenocide", "White Genocide", tweet)
tweet = re.sub(r"dirtylying", "dirty lying", tweet)
tweet = re.sub(r"SyrianRefugees", "Syrian Refugees", tweet)
tweet = re.sub(r"changetheworld", "change the world", tweet)
tweet = re.sub(r"Ebolacase", "Ebola case", tweet)
tweet = re.sub(r"mcgtech", "mcg technologies", tweet)
tweet = re.sub(r"withweapons", "with weapons", tweet)
tweet = re.sub(r"advancedwarfare", "advanced warfare", tweet)
tweet = re.sub(r"letsFootball", "let us Football", tweet)
tweet = re.sub(r"LateNiteMix", "late night mix", tweet)
tweet = re.sub(r"PhilCollinsFeed", "Phil Collins", tweet)
tweet = re.sub(r"RudyHavenstein", "Rudy Havenstein", tweet)
tweet = re.sub(r"22PM", "22 PM", tweet)
tweet = re.sub(r"54am", "54 AM", tweet)
tweet = re.sub(r"38am", "38 AM", tweet)
```

```
tweet = re.sub(r"OldFolkExplainStuff", "Old Folk Explain Stuff", tweet)
tweet = re.sub(r"BlacklivesMatter", "Black Lives Matter", tweet)
tweet = re.sub(r"InsaneLimits", "Insane Limits", tweet)
tweet = re.sub(r"youcantsitwithus", "you cannot sit with us", tweet)
tweet = re.sub(r"2k15", "2015", tweet)
tweet = re.sub(r"TheIran", "Iran", tweet)
tweet = re.sub(r"JimmyFallon", "Jimmy Fallon", tweet)
tweet = re.sub(r"AlbertBrooks", "Albert Brooks", tweet)
tweet = re.sub(r"defense_news", "defense news", tweet)
tweet = re.sub(r"nuclearrcSA", "Nuclear Risk Control Self Assessment", tweet)
tweet = re.sub(r"Auspol", "Australia Politics", tweet)
tweet = re.sub(r"NuclearPower", "Nuclear Power", tweet)
tweet = re.sub(r"WhiteTerrorism", "White Terrorism", tweet)
tweet = re.sub(r"truthfrequencyradio", "Truth Frequency Radio", tweet)
tweet = re.sub(r"ErasureIsNotEquality", "Erasure is not equality", tweet)
tweet = re.sub(r"ProBonoNews", "Pro Bono News", tweet)
tweet = re.sub(r"JakartaPost", "Jakarta Post", tweet)
tweet = re.sub(r"toopainful", "too painful", tweet)
tweet = re.sub(r"melindahaunton", "Melinda Haunton", tweet)
tweet = re.sub(r"NoNukes", "No Nukes", tweet)
tweet = re.sub(r"curryspeworld", "Currys PC World", tweet)
tweet = re.sub(r"ineedcake", "I need cake", tweet)
tweet = re.sub(r"blackforestgateau", "black forest gateau", tweet)
tweet = re.sub(r"BBCTOne", "BBC One", tweet)
tweet = re.sub(r"AlexxPage", "Alex Page", tweet)
tweet = re.sub(r"jonathanserrie", "Jonathan Serrie", tweet)
tweet = re.sub(r"SocialJerkBlog", "Social Jerk Blog", tweet)
tweet = re.sub(r"ChelseaVPeretti", "Chelsea Peretti", tweet)
tweet = re.sub(r"irongiant", "iron giant", tweet)
tweet = re.sub(r"RonFunches", "Ron Funches", tweet)
tweet = re.sub(r"TimCook", "Tim Cook", tweet)
tweet = re.sub(r"sebastianstanisaliveandwell", "Sebastian Stan is alive and well", tweet)
tweet = re.sub(r"Madsummer", "Mad summer", tweet)
tweet = re.sub(r"NowYouKnow", "Now you know", tweet)
```

```
tweet = re.sub(r"concertphotography", "concert photography", tweet)
tweet = re.sub(r"TomLandry", "Tom Landry", tweet)
tweet = re.sub(r"showgirldayoff", "show girl day off", tweet)
tweet = re.sub(r"Yougslavia", "Yugoslavia", tweet)
tweet = re.sub(r"QuantumDataInformatics", "Quantum Data Informatics", tweet)
tweet = re.sub(r"FromTheDesk", "From The Desk", tweet)
tweet = re.sub(r"TheaterTrial", "Theater Trial", tweet)
tweet = re.sub(r"CatoInstitute", "Cato Institute", tweet)
tweet = re.sub(r"EmekaGift", "Emeka Gift", tweet)
tweet = re.sub(r"LetsBe_Rational", "Let us be rational", tweet)
tweet = re.sub(r"Cynicalreality", "Cynical reality", tweet)
tweet = re.sub(r"FredOlsenCruise", "Fred Olsen Cruise", tweet)
tweet = re.sub(r"NotSorry", "not sorry", tweet)
tweet = re.sub(r"UseYourWords", "use your words", tweet)
tweet = re.sub(r"WordoftheDay", "word of the day", tweet)
tweet = re.sub(r"Dictionarycom", "Dictionary.com", tweet)
tweet = re.sub(r"TheBrooklynLife", "The Brooklyn Life", tweet)
tweet = re.sub(r"jokethey", "joke they", tweet)
tweet = re.sub(r"nflweek1picks", "NFL week 1 picks", tweet)
tweet = re.sub(r"uiseful", "useful", tweet)
tweet = re.sub(r"JusticeDotOrg", "The American Association for Justice", tweet)
tweet = re.sub(r"autoaccidents", "auto accidents", tweet)
tweet = re.sub(r"SteveGursten", "Steve Gursten", tweet)
tweet = re.sub(r"MichiganAutoLaw", "Michigan Auto Law", tweet)
tweet = re.sub(r"birdgang", "bird gang", tweet)
tweet = re.sub(r"nflnetwork", "NFL Network", tweet)
tweet = re.sub(r"NYDNSports", "NY Daily News Sports", tweet)
tweet = re.sub(r"RVacchianoNYDN", "Ralph Vacchiano NY Daily News", tweet)
tweet = re.sub(r"EdmontonEsks", "Edmonton Eskimos", tweet)
tweet = re.sub(r"david_brelsford", "David Brelsford", tweet)
tweet = re.sub(r"TOI_India", "The Times of India", tweet)
tweet = re.sub(r"hegot", "he got", tweet)
tweet = re.sub(r"SkinsOn9", "Skins on 9", tweet)
tweet = re.sub(r"sothathappened", "so that happened", tweet)
```

```
tweet = re.sub(r"LCOutOfDoors", "LC Out Of Doors", tweet)
tweet = re.sub(r"NationFirst", "Nation First", tweet)
tweet = re.sub(r"IndiaToday", "India Today", tweet)
tweet = re.sub(r"HLPS", "helps", tweet)
tweet = re.sub(r"HOSTAGESTHROSW", "hostages throw", tweet)
tweet = re.sub(r"SNCTIONS", "sanctions", tweet)
tweet = re.sub(r"BidTime", "Bid Time", tweet)
tweet = re.sub(r"crunchysensible", "crunchy sensible", tweet)
tweet = re.sub(r"RandomActsOfRomance", "Random acts of romance", tweet)
tweet = re.sub(r"MomentsAtHill", "Moments at hill", tweet)
tweet = re.sub(r"eatshit", "eat shit", tweet)
tweet = re.sub(r"liveleakfun", "live leak fun", tweet)
tweet = re.sub(r"SahelNews", "Sahel News", tweet)
tweet = re.sub(r"abc7newsbayarea", "ABC 7 News Bay Area", tweet)
tweet = re.sub(r"facilitiesmanagement", "facilities management", tweet)
tweet = re.sub(r"facilitydude", "facility dude", tweet)
tweet = re.sub(r"CampLogistics", "Camp logistics", tweet)
tweet = re.sub(r"alaskapublic", "Alaska public", tweet)
tweet = re.sub(r"MarketResearch", "Market Research", tweet)
tweet = re.sub(r"AccuracyEsports", "Accuracy Esports", tweet)
tweet = re.sub(r"TheBodyShopAust", "The Body Shop Australia", tweet)
tweet = re.sub(r"yychail", "Calgary hail", tweet)
tweet = re.sub(r"yyctraffic", "Calgary traffic", tweet)
tweet = re.sub(r"eliotsschool", "eliot school", tweet)
tweet = re.sub(r"TheBrokenCity", "The Broken City", tweet)
tweet = re.sub(r"OldsFireDept", "Olds Fire Department", tweet)
tweet = re.sub(r"RiverComplex", "River Complex", tweet)
tweet = re.sub(r"fieldworksmells", "field work smells", tweet)
tweet = re.sub(r"IranElection", "Iran Election", tweet)
tweet = re.sub(r"glowng", "glowing", tweet)
tweet = re.sub(r"kindlng", "kindling", tweet)
tweet = re.sub(r"riggd", "rigged", tweet)
tweet = re.sub(r"slownewsday", "slow news day", tweet)
tweet = re.sub(r"MyanmarFlood", "Myanmar Flood", tweet)
```

```
tweet = re.sub(r"abc7chicago", "ABC 7 Chicago", tweet)
tweet = re.sub(r"copolitics", "Colorado Politics", tweet)
tweet = re.sub(r"AdilGhumro", "Adil Ghumro", tweet)
tweet = re.sub(r"netbots", "net bots", tweet)
tweet = re.sub(r"byebyeroad", "bye bye road", tweet)
tweet = re.sub(r"massiveflooding", "massive flooding", tweet)
tweet = re.sub(r"EndofUS", "End of United States", tweet)
tweet = re.sub(r"35PM", "35 PM", tweet)
tweet = re.sub(r"greektheatre", "Greek Theatre Los Angeles", tweet)
tweet = re.sub(r"76mins", "76 minutes", tweet)
tweet = re.sub(r"publicsafetyfirst", "public safety first", tweet)
tweet = re.sub(r"livesmatter", "lives matter", tweet)
tweet = re.sub(r"myhometown", "my hometown", tweet)
tweet = re.sub(r"tankerfire", "tanker fire", tweet)
tweet = re.sub(r"MEMORIALDAY", "memorial day", tweet)
tweet = re.sub(r"MEMORIAL_DAY", "memorial day", tweet)
tweet = re.sub(r"instaxbooty", "instagram booty", tweet)
tweet = re.sub(r"Jerusalem_Post", "Jerusalem Post", tweet)
tweet = re.sub(r"WayneRooney_INA", "Wayne Rooney", tweet)
tweet = re.sub(r"VirtualReality", "Virtual Reality", tweet)
tweet = re.sub(r"OculusRift", "Oculus Rift", tweet)
tweet = re.sub(r"OwenJones84", "Owen Jones", tweet)
tweet = re.sub(r"jeremycorbyn", "Jeremy Corbyn", tweet)
tweet = re.sub(r"paulrogers002", "Paul Rogers", tweet)
tweet = re.sub(r"mortalkombatx", "Mortal Kombat X", tweet)
tweet = re.sub(r"mortalkombat", "Mortal Kombat", tweet)
tweet = re.sub(r"FilipeCoelho92", "Filipe Coelho", tweet)
tweet = re.sub(r"OnlyQuakeNews", "Only Quake News", tweet)
tweet = re.sub(r"kostumes", "costumes", tweet)
tweet = re.sub(r"YEEESSSS", "yes", tweet)
tweet = re.sub(r"ToshikazuKatayama", "Toshikazu Katayama", tweet)
tweet = re.sub(r"IntlDevelopment", "Intl Development", tweet)
tweet = re.sub(r"ExtremeWeather", "Extreme Weather", tweet)
tweet = re.sub(r"WereNotGruberVoters", "We are not gruber voters", tweet)
```



```
tweet = re.sub(r"NewsThousands", "News Thousands", tweet)
tweet = re.sub(r"EdmundAdamus", "Edmund Adamus", tweet)
tweet = re.sub(r"EyewitnessWV", "Eye witness WV", tweet)
tweet = re.sub(r"PhiladelphiaMuseu", "Philadelphia Museum", tweet)
tweet = re.sub(r"DublinComicCon", "Dublin Comic Con", tweet)
tweet = re.sub(r"NicholasBrendon", "Nicholas Brendon", tweet)
tweet = re.sub(r"Alltheway80s", "All the way 80s", tweet)
tweet = re.sub(r"FromTheField", "From the field", tweet)
tweet = re.sub(r"NorthIowa", "North Iowa", tweet)
tweet = re.sub(r"WillowFire", "Willow Fire", tweet)
tweet = re.sub(r"MadRiverComplex", "Mad River Complex", tweet)
tweet = re.sub(r"feelingmanly", "feeling manly", tweet)
tweet = re.sub(r"stillnotoverit", "still not over it", tweet)
tweet = re.sub(r"FortitudeValley", "Fortitude Valley", tweet)
tweet = re.sub(r"CoastpowerlineTramTr", "Coast powerline", tweet)
tweet = re.sub(r"ServicesGold", "Services Gold", tweet)
tweet = re.sub(r"NewsbrokenEmergency", "News broken emergency", tweet)
tweet = re.sub(r"Evaucation", "evacuation", tweet)
tweet = re.sub(r"leaveevacuateexitbe", "leave evacuate exit be", tweet)
tweet = re.sub(r"P_EOPLE", "PEOPLE", tweet)
tweet = re.sub(r"Tubestrike", "tube strike", tweet)
tweet = re.sub(r"CLASS_SICK", "CLASS SICK", tweet)
tweet = re.sub(r"localplumber", "local plumber", tweet)
tweet = re.sub(r"awesomejobsiri", "awesome job siri", tweet)
tweet = re.sub(r"PayForItHow", "Pay for it how", tweet)
tweet = re.sub(r"ThisIsAfrica", "This is Africa", tweet)
tweet = re.sub(r"crimeairnetwork", "crime air network", tweet)
tweet = re.sub(r"KimAcheson", "Kim Acheson", tweet)
tweet = re.sub(r"cityofcalgary", "City of Calgary", tweet)
tweet = re.sub(r"prosyndicate", "pro syndicate", tweet)
tweet = re.sub(r"660NEWS", "660 NEWS", tweet)
tweet = re.sub(r"BusInsMagazine", "Business Insurance Magazine", tweet)
tweet = re.sub(r"wfocus", "focus", tweet)
tweet = re.sub(r"ShastaDam", "Shasta Dam", tweet)
```

```

tweet = re.sub(r"go2MarkFranco", "Mark Franco", tweet)
tweet = re.sub(r"StephGHinojosa", "Steph Hinojosa", tweet)
tweet = re.sub(r"Nashgrier", "Nash Grier", tweet)
tweet = re.sub(r"NashNewVideo", "Nash new video", tweet)
tweet = re.sub(r"IWouldntGetElectedBecause", "I would not get elected because", tweet)
tweet = re.sub(r"SHGames", "Sledgehammer Games", tweet)
tweet = re.sub(r"bedhair", "bed hair", tweet)
tweet = re.sub(r"JoelHeyman", "Joel Heyman", tweet)
tweet = re.sub(r"viaYouTube", "via YouTube", tweet)

```

Urls

```

tweet = re.sub(r"https?:\Vt.co\A-Za-z0-9]+", "", tweet)

```

Words with punctuations and special characters

```

punctuations = '@#!?+&*[]-%.:;/()$=><|{}^' + ""

```

for p in punctuations:

```

    tweet = tweet.replace(p, ' {p} ')

```

... and ..

```

tweet = tweet.replace('...', ' ... ')

```

if '...' not in tweet:

```

    tweet = tweet.replace('..', ' ... ')

```

Acronyms

```

tweet = re.sub(r"MH370", "Malaysia Airlines Flight 370", tweet)

```

```

tweet = re.sub(r"m1¼sica", "music", tweet)

```

```

tweet = re.sub(r"okwx", "Oklahoma City Weather", tweet)

```

```

tweet = re.sub(r"arwx", "Arkansas Weather", tweet)

```

```

tweet = re.sub(r"gawx", "Georgia Weather", tweet)

```

```

tweet = re.sub(r"scwx", "South Carolina Weather", tweet)

```

```

tweet = re.sub(r"cawx", "California Weather", tweet)

```

```

tweet = re.sub(r"tnwx", "Tennessee Weather", tweet)

```

```

tweet = re.sub(r"azwx", "Arizona Weather", tweet)

```

```

tweet = re.sub(r"alwx", "Alabama Weather", tweet)

```

```

tweet = re.sub(r"wordpressdotcom", "wordpress", tweet)
tweet = re.sub(r"usNWSgov", "United States National Weather Service", tweet)
tweet = re.sub(r"Suruc", "Sanliurfa", tweet)

```

```

# Grouping same words without embeddings

```

```

tweet = re.sub(r"Bestnaijamade", "bestnaijamade", tweet)
tweet = re.sub(r"SOUDELOR", "Soudelor", tweet)

```

```

return tweet

```

```

df_train['text_cleaned'] = df_train['text'].apply(lambda s : clean(s))
df_test['text_cleaned'] = df_test['text'].apply(lambda s : clean(s))

```

```

train_glove_oov,    train_glove_vocab_coverage,    train_glove_text_coverage    =
check_embeddings_coverage(df_train['text_cleaned'], glove_embeddings)
test_glove_oov,    test_glove_vocab_coverage,    test_glove_text_coverage    =
check_embeddings_coverage(df_test['text_cleaned'], glove_embeddings)
print('GloVe Embeddings cover {:.2%} of vocabulary and {:.2%} of text in Training
Set'.format(train_glove_vocab_coverage, train_glove_text_coverage))
print('GloVe Embeddings cover {:.2%} of vocabulary and {:.2%} of text in Test
Set'.format(test_glove_vocab_coverage, test_glove_text_coverage))

```

```

train_fasttext_oov,    train_fasttext_vocab_coverage,    train_fasttext_text_coverage    =
check_embeddings_coverage(df_train['text_cleaned'], fasttext_embeddings)
test_fasttext_oov,    test_fasttext_vocab_coverage,    test_fasttext_text_coverage    =
check_embeddings_coverage(df_test['text_cleaned'], fasttext_embeddings)
print('FastText Embeddings cover {:.2%} of vocabulary and {:.2%} of text in Training
Set'.format(train_fasttext_vocab_coverage, train_fasttext_text_coverage))
print('FastText Embeddings cover {:.2%} of vocabulary and {:.2%} of text in Test
Set'.format(test_fasttext_vocab_coverage, test_fasttext_text_coverage))

```

```

Whole Training Set Shape = (7613, 16)
Whole Training Set Unique keyword Count = 222
Whole Training Set Target Rate (Disaster) 3271/4342 (Not Disaster)

Fold 1 Training Set Shape = (3806,) - Validation Set Shape = (3807,)
Fold 1 Training Set Unique keyword Count = 222 - Validation Set Unique keyword Count = 222

Fold 2 Training Set Shape = (3807,) - Validation Set Shape = (3806,)
Fold 2 Training Set Unique keyword Count = 222 - Validation Set Unique keyword Count = 222

```

K = 2

```
skf = StratifiedKFold(n_splits=K, random_state=SEED, shuffle=True)
```

```
DISASTER = df_train['target'] == 1
```

```
print('Whole Training Set Shape = {}'.format(df_train.shape))
```

```
print('Whole      Training      Set      Unique      keyword      Count      =
      {}'.format(df_train['keyword'].nunique()))
```

```
print('Whole      Training      Set      Target      Rate      (Disaster)      {}/{}      (Not
Disaster)'.format(df_train[DISASTER]['target_relabeled'].count(),
df_train[~DISASTER]['target_relabeled'].count()))
```

```
for fold, (trn_idx, val_idx) in enumerate(skf.split(df_train['text_cleaned'],
df_train['target']), 1):
```

```
    print("\nFold {} Training Set Shape = {} - Validation Set Shape = {}".format(fold,
df_train.loc[trn_idx, 'text_cleaned'].shape, df_train.loc[val_idx, 'text_cleaned'].shape))
```

```
    print('Fold {} Training Set Unique keyword Count = {} - Validation Set Unique keyword
Count = {}'.format(fold, df_train.loc[trn_idx, 'keyword'].nunique(), df_train.loc[val_idx,
'keyword'].nunique()))
```

```

Train on 3747 samples, validate on 3866 samples
Epoch 1/10
3744/3747 [=====>.] - ETA: 0s - loss: 0.6147 - accuracy: 0.6755
Epoch: 1 - Training Precision: 0.735408 - Training Recall: 0.719853 - Training F1: 0.723161
Epoch: 1 - Validation Precision: 0.72065 - Validation Recall: 0.709626 - Validation F1: 0.71231
3747/3747 [=====] - 124s 33ms/sample - loss: 0.6147 - accuracy: 0.6755 - val_loss: 0.5667 - val_accuracy: 0.7243
Epoch 2/10
3744/3747 [=====>.] - ETA: 0s - loss: 0.5228 - accuracy: 0.7487
Epoch: 2 - Training Precision: 0.780094 - Training Recall: 0.756824 - Training F1: 0.761698
Epoch: 2 - Validation Precision: 0.759078 - Validation Recall: 0.74282 - Validation F1: 0.746674
3747/3747 [=====] - 104s 28ms/sample - loss: 0.5229 - accuracy: 0.7486 - val_loss: 0.5126 - val_accuracy: 0.7584
Epoch 3/10
3744/3747 [=====>.] - ETA: 0s - loss: 0.4697 - accuracy: 0.7842
Epoch: 3 - Training Precision: 0.802989 - Training Recall: 0.785967 - Training F1: 0.790651
Epoch: 3 - Validation Precision: 0.776012 - Validation Recall: 0.76321 - Validation F1: 0.766887
3747/3747 [=====] - 104s 28ms/sample - loss: 0.4695 - accuracy: 0.7844 - val_loss: 0.4800 - val_accuracy: 0.7763
Epoch 4/10
3744/3747 [=====>.] - ETA: 0s - loss: 0.4334 - accuracy: 0.8098
Epoch: 4 - Training Precision: 0.824896 - Training Recall: 0.809021 - Training F1: 0.813797
Epoch: 4 - Validation Precision: 0.791327 - Validation Recall: 0.779743 - Validation F1: 0.783362
3747/3747 [=====] - 104s 28ms/sample - loss: 0.4336 - accuracy: 0.8097 - val_loss: 0.4601 - val_accuracy: 0.7915
Epoch 5/10
...
3840/3866 [=====>.] - ETA: 0s - loss: 0.3341 - accuracy: 0.8635
Epoch: 10 - Training Precision: 0.879964 - Training Recall: 0.860556 - Training F1: 0.866622
Epoch: 10 - Validation Precision: 0.863049 - Validation Recall: 0.844368 - Validation F1: 0.850105
3866/3866 [=====] - 105s 27ms/sample - loss: 0.3328 - accuracy: 0.8645 - val_loss: 0.3545 - val_accuracy: 0.8564

```

class DisasterDetector:

```

    def __init__(self, bert_layer, max_seq_length=128, lr=0.0001, epochs=15,
batch_size=32):

```

```

    # BERT and Tokenization params

```

```

    self.bert_layer = bert_layer

```

```

    self.max_seq_length = max_seq_length

```

```

    vocab_file = self.bert_layer.resolved_object.vocab_file.asset_path.numpy()

```

```

    do_lower_case = self.bert_layer.resolved_object.do_lower_case.numpy()

```

```

    self.tokenizer = tokenization.FullTokenizer(vocab_file, do_lower_case)

```

```

    # Learning control params

```

```

    self.lr = lr

```

```

    self.epochs = epochs

```

```

    self.batch_size = batch_size

```

```

    self.models = []

```

```

    self.scores = {}

```

```
def encode(self, texts):
```

```
    all_tokens = []
```

```
    all_masks = []
```

```
    all_segments = []
```

```
    for text in texts:
```

```
        text = self.tokenizer.tokenize(text)
```

```
        text = text[:self.max_seq_length - 2]
```

```
        input_sequence = ['[CLS]'] + text + ['[SEP]']
```

```
        pad_len = self.max_seq_length - len(input_sequence)
```

```
        tokens = self.tokenizer.convert_tokens_to_ids(input_sequence)
```

```
        tokens += [0] * pad_len
```

```
        pad_masks = [1] * len(input_sequence) + [0] * pad_len
```

```
        segment_ids = [0] * self.max_seq_length
```

```
        all_tokens.append(tokens)
```

```
        all_masks.append(pad_masks)
```

```
        all_segments.append(segment_ids)
```

```
    return np.array(all_tokens), np.array(all_masks), np.array(all_segments)
```

```
def build_model(self):
```

```
    input_word_ids = Input(shape=(self.max_seq_length,), dtype=tf.int32,
name='input_word_ids')
```

```
    input_mask = Input(shape=(self.max_seq_length,), dtype=tf.int32,
name='input_mask')
```

```
    segment_ids = Input(shape=(self.max_seq_length,), dtype=tf.int32,
name='segment_ids')
```

```

        pooled_output, sequence_output = self.bert_layer([input_word_ids, input_mask,
segment_ids])
        clf_output = sequence_output[:, 0, :]
        out = Dense(1, activation='sigmoid')(clf_output)

        model = Model(inputs=[input_word_ids, input_mask, segment_ids], outputs=out)
        optimizer = SGD(learning_rate=self.lr, momentum=0.8)
        model.compile(loss='binary_crossentropy', optimizer=optimizer,
metrics=['accuracy'])

    return model

def train(self, X):

    for fold, (trn_idx, val_idx) in enumerate(skf.split(X['text_cleaned'], X['keyword'])):

        print('\nFold {} \n'.format(fold))

        X_trn_encoded = self.encode(X.loc[trn_idx, 'text_cleaned'].str.lower())
        y_trn = X.loc[trn_idx, 'target_relabeled']
        X_val_encoded = self.encode(X.loc[val_idx, 'text_cleaned'].str.lower())
        y_val = X.loc[val_idx, 'target_relabeled']

        # Callbacks
        metrics = ClassificationReport(train_data=(X_trn_encoded, y_trn),
validation_data=(X_val_encoded, y_val))

        # Model
        model = self.build_model()
        model.fit(X_trn_encoded, y_trn, validation_data=(X_val_encoded, y_val),
callbacks=[metrics], epochs=self.epochs, batch_size=self.batch_size)

        self.models.append(model)

```

```

self.scores[fold] = {
    'train': {
        'precision': metrics.train_precision_scores,
        'recall': metrics.train_recall_scores,
        'f1': metrics.train_f1_scores
    },
    'validation': {
        'precision': metrics.val_precision_scores,
        'recall': metrics.val_recall_scores,
        'f1': metrics.val_f1_scores
    }
}

def plot_learning_curve(self):

    fig, axes = plt.subplots(nrows=K, ncols=2, figsize=(20, K * 6), dpi=100)

    for i in range(K):

        # Classification Report curve
        sns.lineplot(x=np.arange(1, self.epochs + 1),
                    y=clf.models[i].history.history['val_accuracy'], ax=axes[i][0], label='val_accuracy')
        sns.lineplot(x=np.arange(1, self.epochs + 1),
                    y=clf.scores[i]['validation']['precision'], ax=axes[i][0], label='val_precision')
        sns.lineplot(x=np.arange(1, self.epochs + 1), y=clf.scores[i]['validation']['recall'],
                    ax=axes[i][0], label='val_recall')
        sns.lineplot(x=np.arange(1, self.epochs + 1), y=clf.scores[i]['validation']['f1'],
                    ax=axes[i][0], label='val_f1')

        axes[i][0].legend()
        axes[i][0].set_title('Fold {} Validation Classification Report'.format(i),
                             fontsize=14)

```



```

# Loss curve
sns.lineplot(x=np.arange(1, self.epochs + 1),
y=clf.models[0].history.history['loss'], ax=axes[i][1], label='train_loss')
sns.lineplot(x=np.arange(1, self.epochs + 1),
y=clf.models[0].history.history['val_loss'], ax=axes[i][1], label='val_loss')

axes[i][1].legend()
axes[i][1].set_title('Fold {} Train / Validation Loss'.format(i), fontsize=14)

for j in range(2):
    axes[i][j].set_xlabel('Epoch', size=12)
    axes[i][j].tick_params(axis='x', labelsz=12)
    axes[i][j].tick_params(axis='y', labelsz=12)

plt.show()

def predict(self, X):

    X_test_encoded = self.encode(X['text_cleaned'].str.lower())
    y_pred = np.zeros((X_test_encoded[0].shape[0], 1))

    for model in self.models:
        y_pred += model.predict(X_test_encoded) / len(self.models)

    return y_pred

clf = DisasterDetector(bert_layer, max_seq_length=128, lr=0.0001, epochs=10,
batch_size=32)

clf.train(df_train)

```

Model Parameters:

