BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CẦN THƠ

TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

BÁO CÁO
KHAI KHOÁNG DỮ LIỆU

Đề tài

XÂY DỰNG MÔ HÌNH NHẬN DẠNG TWEETS LIÊN QUAN ĐẾN THẢM HỌA

Giáo viên hướng dẫn:
TS. Lưu Tiến Đạo

Sinh viên thực hiện:
Lê Nhật Trọng B2106819
Nguyễn Phúc Hậu B2106790
Trần Nguyễn Nhật Huy B2113333

Cần Thơ, 11/2024

NHẬN XÉT CỦA GIẢNG VIÊN

LỜI CẢM ƠN

Nhóm chúng em xin gửi lời cảm ơn chân thành nhất đến toàn thể Quý thầy cô Trường Đại học Cần Thơ, Quý thầy cô Trường Công Nghệ Thông Tin và Truyền Thông đã dìu dắt và truyền đạt những kinh nghiệm quý báu trong suốt những năm học vừa qua của nhóm.

Nhóm chúng em cũng chân thành cảm ơn bạn bè cùng với gia đình đã luôn động viên, khích lệ và tạo điều kiện giúp đỡ trong suốt quá trình thực hiện để nhóm có thể hoàn thành bài báo cáo một cách tốt nhất.

Và để hoàn thành bài báo cáo này, nhóm chúng em xin bày tỏ lòng biết ơn sâu sắc đến thầy Lưu Tiến Đạo, người đã tận tình hướng dẫn nhóm trong việc lựa chọn đề tài phù hợp và trong suốt thời gian thực hiện báo cáo này. Nhóm chúng em xin cảm ơn thầy rất nhiều vì trong suốt khoảng thời gian này thầy đã dành rất nhiều thời gian và tâm huyết để hỗ trợ nhóm hoàn thành tốt bài báo cáo học phần Khai khoáng Dữ liệu của nhóm.

Với điều kiện thời gian, vốn kiến thức còn hạn hẹp cũng như kinh nghiệm còn hạn chế, bài báo cáo này không thể tránh được các thiếu sót. Nhóm chúng em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của thầy để nhóm có điều kiện bổ sung, nâng cao phần trình bày và thể hiện của bài báo cáo. Đó cũng là hành trang quý giá giúp nhóm chúng em hoàn thiện kiến thức của mình sau này.

Nhóm chúng em xin chân thành cảm ơn!

Cần Thơ, ngày 14 tháng 05 năm 2024

Người viết

Lê Nhật Trọng

Nguyễn Phúc Hậu

Trần Nguyễn Nhật Huy

CAM KẾT

Đề tài này được thực hiện bởi nhóm sinh viên bao gồm Nguyễn Phúc Hậu, Lê Nhật Trọng và Trần Nguyễn Nhật Huy. Nhóm xin cam đoan đề tài "***Xây dựng mô hình nhận dạng tweets liên quan đến thảm họa***" là quá trình tìm hiểu, học hỏi, nghiên cứu và thảo luận của chính nhóm trong học phần Khai khoáng Dữ liệu (Data Mining), dưới sự hướng dẫn bởi TS. Lưu Tiến Đạo.

Các thông tin có liên quan được sử dụng trong đề tài đều được tham khảo và trích dẫn từ các nguồn tài liệu đáng tin cậy, đã được kiểm chứng và công bố rộng rãi. Các kết quả đánh giá mà nhóm ghi lại trong bài báo cáo này hoàn toàn là có cơ sở và được thực hiện rõ ràng từng bước qua từng bước thực hiện, bao gồm việc chạy thực tế các mô hình sau khi đã xử lý lại dữ liệu ban đầu, từ đó rút trích ra kết quả đánh giá và sau đó được triển khai ứng dụng cơ bản trên nền tảng web.

PHÂN CHIA CÔNG VIỆC

Họ và tên	Công việc	Hoàn thành
Lê Nhật Trọng	- Tìm hiểu và xử lý tập dữ liệu ban đầu - Tham gia viết code làm sạch dữ liệu - Huấn luyện các mô hình cơ bản với Word2Vec - Xây dựng website dự đoán Tweet - Huấn luyện mô hình BERT - Deploy website lên Render - Góp ý, viết và chỉnh sửa quyển báo cáo và slide báo cáo	100%
Nguyễn Phúc Hậu	- Tìm hiểu và xử lý tập dữ liệu - Tham gia làm sạch dữ liệu - Huấn luyện các mô hình cơ bản với đặc trưng TF-IDF - Soạn slide chính để thuyết trình - Góp ý, chỉnh sửa quyển báo cáo	100%
Trần Nguyễn Nhật Huy	- Xây dựng các module làm sạch và chuẩn hóa dữ liệu - Thiết kế pipeline xử lý dữ liệu - Phân tích và trực quan hóa dữ liệu - Viết base code train các mô hình - Huấn luyện các mô hình cơ bản với đặc trưng Bag of Words - Huấn luyện các mô hình học sâu với cả 3 đặc trưng - Góp ý, viết, chỉnh sửa quyển báo cáo và slide	100%

Bảng 1. Bảng phân chia công việc

MỤC LỤC

DANH MỤC HÌNH

DANH MỤC BẢNG

ABSTRACT

This project focuses on developing a model to classify tweets related to disasters. Given the dataset's noisy nature and instances of data loss, the team built a processing pipeline to effectively clean and normalize the data. Once cleaned, the data is transformed into vector space before being used to train various models. The team experimented with a range of models, from basic approaches to deep learning and natural language processing models, aiming to identify the optimal method. Finally, a website will be developed, allowing users to input a tweet and receive a prediction on its relevance to disasters. This website will also offer users the option to select different classification models and feature extraction methods to suit their personal needs.

TÓM TẮT

Đề tài này tập trung vào việc xây dựng mô hình phân loại các tweets liên quan đến thảm họa. Với đặc điểm tập dữ liệu chứa nhiều nhiễu và tình trạng mất mát dữ liệu, nhóm đã xây dựng một pipeline xử lý để làm sạch và chuẩn hóa dữ liệu một cách hiệu quả. Dữ liệu sau khi làm sạch sẽ được chuyển đổi thành không gian vector trước khi đưa vào huấn luyện với các mô hình khác nhau. Nhóm đã thử nghiệm từ các mô hình cơ bản đến các mô hình học sâu và xử lý ngôn ngữ tự nhiên, với mục tiêu tìm ra phương pháp tối ưu nhất. Cuối cùng, một trang web sẽ được phát triển, cho phép người dùng nhập nội dung tweet để nhận dự báo về khả năng liên quan đến thảm họa của tweet đó. Trang web này còn cung cấp tùy chọn cho người dùng lựa chọn mô hình phân loại và phương pháp trích xuất đặc trưng phù hợp với nhu cầu cá nhân.

PHẦN GIỚI THIỆU

1. Đặt vấn đề

Trong bối cảnh thế giới đang phát triển một cách nhanh chóng như hiện nay, một phần lớn lượng khí thải sinh hoạt, việc khai thác quá mức thiên nhiên và các tác nhân khác đã gây tổn hại ít nhiều đến thiên nhiên gây ra nhiều hậu quả không thể nào lường trước được. Đứng trước thiên nhiên con người thật sự rất nhỏ bé, những sự kiện như lũ lụt, động đất, cháy rừng,... không chỉ gây thiệt hại về tánh mạng con người, tài sản mà còn ảnh hưởng sâu sắc đến môi trường, an sinh xã hội và đời sống của hàng triệu người. Do đó mà thảm họa do thiên nhiên và nhân tạo đang trở thành một trong những vấn đề cấp bách mà xã hội loài người cần phải đặt lên bàn cân để cân nhắc đưa ra giải pháp hiệu quả nhất nhằm giảm bớt tần suất xuất hiện và giảm tối thiểu thiệt hại do thiên tai bão lũ gây ra. Trong bối cảnh đó, việc theo dõi phân tích và nhận diện thông tin liên quan đến các thảm họa là cực kì quan trọng để nâng cao khả năng ứng phó.

Với sự phát triển mạnh mẽ của nền công nghệ và tốc độ lan truyền thông tin chóng mặt của mạng xã hội như hiện nay, điển hình như các Tweets trên Twitter. Mức độ lan tỏa thông tin các Tweets diễn ra nhanh chóng, đặc biệt là các Tweets có liên quan đến thông tin thảm họa. Thông tin này được đánh giá một cách khách quan và thực tế do nó được đăng tải bởi các người dùng trực tiếp trải nghiệm qua sự kiện, hoặc những người chứng kiến từ xa,... điều này giúp chúng ta có cái nhìn chân thực và tổng quát hơn về tình hình thực tế.

Trong bối cảnh đó, việc xây dựng mô hình nhận dạng các Tweets liên quan đến thảm họa trở thành một nhiệm vụ quan trọng. Mô hình này không chỉ giúp phát hiện nhanh chóng, chính xác các thông tin quan trọng nhằm giúp các cơ quan chính phủ nắm bắt thông tin để đưa ra thông tin sơ tác, cứu trợ và các biện pháp ứng phó kịp thời. Thông qua việc xử lí ngôn ngữ tự nhiên và máy học, mô hình này không chỉ dừng lại ở việc nhận dạng các thảm họa mà còn giúp giảm thiểu mức độ thiệt hại mà bão lụt, động đất,.. gây ra.

Ngoài ra, mô hình này còn góp phần nâng cao nhận thức của mọi người về những hậu quả mà thiên tai gây ra. Từ đó, khuyến khích mọi người có ý thức bảo vệ thiên nhiên, bảo vệ môi trường sống, giảm bớt lượng rác thải,... để nâng cao chất lượng đời sống giảm bớt hiện tượng biến đổi khí hậu.

2. Mục tiêu đề tài

Xây dựng các mô hình máy học cơ bản và nâng cao nhằm phân loại và dự báo các tweets thành các nhóm có liên quan đến thảm họa.

3. Đối tượng và phạm vi nghiên cứu

3.1. Đối tượng nghiên cứu

Đề tài tập trung nghiên cứu về các tweet trên nền tảng Twitter (hiện tại là X), đặc biệt là các tweet có nội dung liên quan đến thảm họa như thiên tai, tai nạn, cháy nổ, và các sự cố khẩn cấp khác. Ở các tweets, cần tập trung vào các thông tin như các từ khóa của tweet, vị trí địa lý có liên quan đến tweet, nội dung của tweet và tweet đó có liên quan đến thảm họa hay không.

3.2. Phạm vi nghiên cứu

Sau khi tìm hiểu về cơ sở lý thuyết và các đề tài có liên quan, các phạm vi nghiên cứu sẽ được liệt kê ra ở dưới đây:

- Đề tài chỉ nghiên cứu về các tweets trên nền tảng Twitter (hiện tại là X).
- Các mô hình được xây dựng dựa trên tập dữ liệu được cung cấp từ trước, không sử dụng thêm bất kỳ dữ liệu nào từ bên ngoài.
- Chỉ tập trung vào huấn luyện các mô hình máy học thông thường. Mặc dù các mô hình học sâu có huấn luyện nhưng chỉ được dùng để so sánh và phân tích, không dùng để triển khai.
- Triển khai trang web ứng dụng trên các nền tảng hosting miễn phí với các thư viện được hỗ trợ

4. Phương pháp nghiên cứu

4.1. Về mặt lý thuyết

Đề tài này tổng hợp các kiến thức đã học về máy học ứng dụng, khai khoáng dữ liệu và lập trình Web. Bên cạnh đó, các thư viện và công nghệ hiện đại như, pandas, nltk và Next.js được sử dụng để xử lý dữ liệu, xây dựng nên các mô hình và triển khai hệ thống. Vì vậy về mặt lý thuyết sẽ bao gồm những việc sau:

- Tìm hiểu cách phân tích thống kê và trực quan hóa mối quan hệ giữa các cột dữ liệu với matplotlib và seaborn
- Tham khảo về việc sử dụng regex và module lên quan đến NLP (Natureal Language Processing) để làm sạch dữ liệu khỏi các thành phần gây nhiễu
- Nghiên cứu về việc vector hóa văn bản bằng các kỹ thuật như Bag of Words và TF-IDF hay Word Embedding
- Tham khảo tài liệu của scikit-learn và tensorflow nhằm xây dựng và huấn luyện các mô hình cơ bản và nâng cao với các tham số tùy chỉnh và các kiến trúc khác nhau

- Nghiên cứu về CSR (Client-Side Rendering) trong mô hình client server, áp dụng chuẩn RESTful API để thiết kế các API endpoints cho việc trao đổi dữ liệu giữa Next.js và FastAPI

4.2. Về mặt thực hành

- Thiết kế và xây dựng các pipeline tiền xử lý dữ liệu ban đầu để làm sạch và chuẩn hóa dữ liệu
- Xây dựng mô hình dự báo để phân loại các tweets với nhiều giải thuật khác nhau, bao gồm các giải thuật cơ bản như Logistic Regression và Decision Tree hay mạng nơ ron truyền thẳng
- Thực hiện nghi thức hold out và huấn luyện nhiều lần để đảm bảo tính chính xác và độ tin cậy của các kết quả.
- Thiết kế ứng dụng web có giao diện người dùng đơn giản và hiện đại, logic xử lý của server nhanh chóng và hiệu quả.

5. Kết quả đạt được

Xử lý thành công tập dữ liệu thô ban đầu thành dữ liệu phù hợp để huấn luyện các mô hình. Huấn luyện thành công các mô hình cho phép dự báo xem một tweet đó có liên quan đến thảm họa hay không. Triển khai thành công một ứng dụng web với giao diện trực quan, thân thiện với người dùng, cho phép người dùng có thể nhập vào nội dung một tweet và hệ thống sẽ tiến hành dự đoán và trả về kết quả. Người dùng có thể lựa chọn các mô hình khác nhau để dự báo nhằm cung cấp thông tin một cách nhanh chóng và chính xác.

6. Bố cục bài báo cáo

Phần giới thiệu

Giới thiệu tổng quát về đề tài

Phần nội dung

Chương 1: Mô tả chi tiết bài toán, các vấn đề và giải pháp liên quan đến bài toán, các thư viện và công cụ hỗ trợ

Chương 2: Tổng quan về dữ liệu, phân tích tập dữ liệu và mối liên quan giữ các cột giá trị, các phương pháp xử lý dữ liệu

Chương 3: Các phương pháp trích xuất đặc trưng, các mô hình huấn luyện, nghi thức đánh giá mô hình, kết quả huấn luyện và đánh giá mô hình

Chương 4: Triển khai hệ thống trên nền tảng web, các công nghệ sử dụng, nền tảng dùng để deploy và quá trình deploy, giao diện thực tế

Phần kết luận

Trình bày các kết quả đạt được, hướng phát triển và kết luận chung

PHẦN NỘI DUNG
CHƯƠNG 1:
MÔ TẢ BÀI TOÁN

1.1. Mô tả chi tiết bài toán

Trong bối cảnh các phương tiện truyền thông xã hội trở thành một nguồn thông tin nhanh chóng và dễ tiếp cận, việc nhận diện các bài đăng (tweets) liên quan đến thảm họa trên Twitter đã trở nên quan trọng để hỗ trợ các tổ chức cứu trợ trong việc thu thập dữ liệu và ra quyết định. Đề tài này đặt ra mục tiêu xây dựng một mô hình máy học có khả năng phân loại các tweets thành hai nhóm: liên quan đến thảm họa và không liên quan đến thảm họa.

Dữ liệu đầu vào là tập hợp các tweets, trong đó mỗi bài đăng bao gồm nội dung, từ khóa và một số thông tin phụ khác như hashtag, và vị trí địa lí (nếu có). Mục tiêu của mô hình là phân loại từng tweet vào một trong hai lớp: các tweet có nội dung phản ánh tình huống khẩn cấp, thiên tai hoặc các sự kiện nguy cấp sẽ được gán nhãn "liên quan đến thảm họa", trong khi các tweet còn lại, có nội dung không liên quan đến các tình huống khẩn cấp, sẽ được gán nhãn "không liên quan đến thảm họa". Đầu ra mong muốn của mô hình là một giá trị nhị phân, trong đó 0 biểu thị tweet không liên quan đến thảm họa và 1 biểu thị tweet có liên quan đến thảm họa.

Để thực hiện được điều này, trước tiên cần tiến hành tiền xử lý dữ liệu với các bước như làm sạch, loại bỏ các ký tự đặc biệt, từ dừng và chuẩn hóa từ ngữ để đảm bảo dữ liệu được đưa về dạng thống nhất. Sau đó, các kỹ thuật xử lý ngôn ngữ tự nhiên sẽ được áp dụng để chuẩn hóa từ vựng và trích xuất đặc trưng từ văn bản, chẳng hạn như sử dụng Bag of Words, TF-IDF hoặc Word Embeddings để chuyển đổi các tweet thành dạng vector phục vụ cho việc huấn luyện mô hình. Bên cạnh đó, các trường dữ liệu quan trọng như từ khóa và hashtag sẽ được phân tích để bổ sung thông tin cho mô hình, giúp tăng cường độ chính xác trong quá trình phân loại.

Các mô hình máy học phù hợp với bài toán phân loại nhị phân, bao gồm Logistic Regression, Naive Bayes, SVM hoặc các mạng nơ-ron hồi quy như RNN và LSTM, sẽ được thử nghiệm và điều chỉnh để tìm ra mô hình tối ưu nhất. Các mô hình sẽ được đánh giá thông qua các chỉ số như Accuracy, Precision, Recall và F1-score.

Sau khi huấn luyện và tìm ra được mô hình có kết quả tốt nhất, chúng ta sẽ tiến hành deploy một website đơn giản cho phép người dùng nhập vào một đoạn văn bản để dự đoán. Đoạn văn bản sẽ được xử lý tương tự như khi chúng ta tiền xử lý văn bản nội dung và đưa qua mô hình để phân loại. Kết quả trả về cho người dùng sẽ là một con số cho biết tỉ lệ tweet trên có liên quan đến thảm họa.

1.2. Vấn đề và giải pháp liên quan đến bài toán

1.2.1. Tiền xử lý dữ liệu

Tiền xử lý dữ liệu là một bước quan trọng trong bài toán này, vì chất lượng của dữ liệu huấn luyện ảnh hưởng trực tiếp đến kết quả của mô hình. Trong trường hợp này, các tweets thường chứa thông tin không có cấu trúc, có thể bao gồm nhiều yếu tố gây nhiễu như các từ viết tắt, ký tự đặc biệt, từ dừng và ngôn ngữ không phải là tiếng Anh. Để giải quyết vấn đề này, cần thực hiện một loạt các bước tiền xử lý nhằm làm sạch và chuẩn hóa dữ liệu trước khi đưa vào mô hình.

Đầu tiên, việc làm sạch dữ liệu cần được thực hiện bằng cách loại bỏ các ký tự đặc biệt, biểu tượng cảm xúc, handle người dùng và các URL không cần thiết. Đối với các hashtags, đôi khi chúng có chứa các thông tin liên quan đến thảm họa, vì vẫn sẽ xử lí theo cách bỏ kí tự "#" và tiến hành tách các từ (nếu có thể). Ngoài ra, chúng ta cần phải loại bỏ các từ dừng (stop words) do chúng không mang lại nhiều thông tin hữu ích cho quá trình phân tích ngữ nghĩa.

Tiếp theo, việc chuẩn hóa văn bản là rất cần thiết để biến đổi các từ ngữ về dạng cơ bản của chúng. Quá trình này thường được thực hiện qua lemmatization, nơi các từ sẽ được đưa về dạng gốc của chúng, giúp giảm thiểu số lượng từ khác nhau mà mô hình cần xử lý và tăng tính nhất quán trong dữ liệu. Sau đó, các từ sẽ được đưa qua một bước kiểm tra để loại bỏ các từ không phải tiếng Anh, hoặc các từ có nghĩa khác xa hoàn toàn đối với các từ ban đầu. Điều này là cần thiết vì quá trình lemmatization có thể sản sinh ra các từ mặc dù vẫn có ngữ nghĩa, tuy nhiên đó là các từ nguyên không hề liên quan đến từ gốc.

1.2.2. Mô hình hóa văn bản lên không gian vector

Dữ liệu văn bản, như các tweet, thường chứa nhiều thông tin nhưng lại không có một cấu trúc cụ thể. Khi làm việc với dữ liệu này, các mô hình không thể trực tiếp xử lý các chuỗi văn bản, do đó cần phải chuyển đổi chúng thành dạng số để có thể áp dụng các giải thuật dùng để phân loại. Nếu không có bước chuyển đổi này, các mô hình sẽ không thể hiểu và phân tích nội dung của các tweets.

Có nhiều phương pháp vector hóa văn bản và mỗi phương pháp đều có ưu và nhược điểm riêng. Phương pháp Bag of Words (BoW) đơn giản biểu diễn mỗi tweet bằng một vector có kích thước tương ứng với số từ độc nhất trong tập dữ liệu, nhưng không nắm bắt được ngữ cảnh và mối quan hệ giữa các từ. Term Frequency-Inverse Document Frequency (TF-IDF) cải thiện BoW bằng cách tính trọng số cho từ, kết hợp tần suất từ (TF) và tần suất ngược tài liệu (IDF),

giúp nổi bật các từ đặc trưng. Một phương pháp đáng kể là sử dụng word embeddings như Word2Vec hoặc GloVe, chuyển đổi từ thành vector trong không gian nhiều chiều, cho phép nắm bắt mối quan hệ ngữ nghĩa giữa các từ. Sau khi chọn phương pháp, mỗi tweet sẽ được chuyển đổi thành một vector đặc trưng, giúp mô hình hiểu rõ hơn về nội dung của từng tweet.

1.2.3. Lựa chọn mô hình phân loại

Có nhiều thuật toán phân loại khác nhau có thể được áp dụng, và việc chọn mô hình phù hợp phụ thuộc vào đặc điểm dữ liệu cũng như mục tiêu cụ thể của bài toán. Một số mô hình phổ biến bao gồm Logistic Regression, Support Vector Machines (SVM), và Decision Trees. Logistic Regression là một phương pháp đơn giản nhưng hiệu quả, thường được sử dụng cho các bài toán phân loại nhị phân. Mô hình này hoạt động tốt khi có mối quan hệ tuyến tính giữa các đặc trưng và nhãn mục tiêu, đồng thời dễ dàng giải thích.

Ngoài ra, mạng nơ-ron (Neural Networks) ngày càng trở nên phổ biến trong các bài toán phân loại văn bản nhờ vào khả năng xử lý các mối quan hệ phức tạp trong dữ liệu. Long Short-Term Memory (LSTM) và Gated Recurrent Unit (GRU) là hai kiến trúc mạng nơ-ron hồi tiếp (RNN) đặc biệt hiệu quả trong việc xử lý chuỗi dữ liệu, như văn bản. LSTM được thiết kế để khắc phục vấn đề vanishing gradient, cho phép nó ghi nhớ thông tin trong thời gian dài hơn và nắm bắt các mối quan hệ ngữ nghĩa trong các tweet. GRU là một biến thể của LSTM, có cấu trúc đơn giản hơn, với số lượng tham số ít hơn, giúp giảm thời gian tính toán mà vẫn duy trì hiệu suất tương đối tốt.

1.3. Các thư viện và công cụ hỗ trợ

1.3.1. Scikit-learn

Scikit-learn là một thư viện mã nguồn mở mạnh mẽ dành cho Python, được thiết kế để phục vụ cho các tác vụ học máy và khai thác dữ liệu. Ra đời vào năm 2007, Scikit-learn cung cấp một giao diện đơn giản và nhất quán cho người dùng, cho phép thực hiện nhanh chóng nhiều thuật toán học máy như hồi quy, phân loại, phân cụm và giảm chiều.

Thư viện này không chỉ hỗ trợ một loạt các phương pháp học máy từ cơ bản đến nâng cao mà còn đi kèm với các công cụ hữu ích cho việc tiền xử lý dữ liệu, chọn lựa mô hình và đánh giá hiệu suất mô hình. Scikit-learn có thể tích hợp dễ dàng với các thư viện phổ biến khác như NumPy và Pandas, giúp người dùng thực hiện các phân tích và xử lý dữ liệu một cách hiệu quả. Nhờ vào cộng đồng đông đảo và tài liệu phong phú, Scikit-learn trở thành lựa chọn hàng đầu cho cả những người mới bắt đầu và các chuyên gia trong lĩnh vực học máy, giúp họ nhanh chóng triển khai các giải pháp học máy trong thực tế.

1.3.2. Tensorflow

TensorFlow là một framework mã nguồn mở do Google phát triển, chủ yếu dành cho việc xây dựng và triển khai các mô hình học máy và học sâu (deep learning). Ra mắt vào năm 2015, TensorFlow nhanh chóng trở thành công cụ phổ biến trong cộng đồng nghiên cứu và phát triển AI, hỗ trợ nhiều ngôn ngữ lập trình như Python, C++, và JavaScript. Điều này giúp các nhà phát triển dễ dàng tiếp cận và áp dụng vào các ứng dụng đa dạng.

Một trong những điểm mạnh của TensorFlow là khả năng xử lý tính toán ma trận phức tạp thông qua các đồ thị tính toán. Framework này cho phép người dùng định nghĩa và tối ưu hóa các mô hình học sâu với nhiều lớp (layers) và tham số (parameters). Bên cạnh đó, TensorFlow hỗ trợ đào tạo mô hình trên nhiều loại phần cứng như CPU, GPU, và TPU, tăng tốc độ tính toán và cải thiện hiệu suất.

Ngoài ra, TensorFlow cung cấp nhiều công cụ hữu ích như TensorBoard để trực quan hóa quá trình đào tạo và TensorFlow Serving để triển khai mô hình đã huấn luyện trong môi trường sản xuất. Với cộng đồng lớn và tài liệu phong phú, TensorFlow trở thành nền tảng lý tưởng cho cả người mới bắt đầu lẫn các chuyên gia. Nhờ tính linh hoạt và khả năng mở rộng, TensorFlow không chỉ được sử dụng trong học sâu mà còn trong nhiều lĩnh vực khác như xử lý ngôn ngữ tự nhiên và thị giác máy tính.

1.3.3. Pandas

Pandas là một thư viện mã nguồn mở trong Python, được phát triển để hỗ trợ xử lý và phân tích dữ liệu một cách dễ dàng và hiệu quả. Ra đời vào năm 2008, Pandas cung cấp hai cấu trúc dữ liệu chính là DataFrame và Series, cho phép người dùng làm việc với dữ liệu có cấu trúc dạng bảng và một chiều.

Với Pandas, người dùng có thể dễ dàng thực hiện các thao tác như đọc và ghi dữ liệu từ nhiều định dạng khác nhau (CSV, Excel, SQL, v.v.), xử lý dữ liệu thiếu, lọc và nhóm dữ liệu, cũng như thực hiện các phép toán thống kê. Thư viện này được tối ưu hóa để xử lý các tập dữ liệu lớn, giúp người dùng nhanh chóng trích xuất thông tin và thực hiện các phân tích phức tạp.

Ngoài ra, Pandas tích hợp tốt với các thư viện khác như NumPy, Matplotlib và Scikit-learn, tạo điều kiện thuận lợi cho việc trực quan hóa dữ liệu và áp dụng các thuật toán học máy. Nhờ vào cú pháp rõ ràng và khả năng linh hoạt, Pandas đã trở thành một công cụ quan trọng cho các nhà khoa học dữ liệu, nhà phân tích và lập trình viên trong việc xử lý và phân tích dữ liệu.

1.3.4. Natural Language Toolkit (NLTK)

Natural Language Toolkit (NLTK) là một thư viện mã nguồn mở dành cho Python, được thiết kế để xử lý và phân tích ngôn ngữ tự nhiên. Ra mắt vào năm 2001, NLTK cung cấp nhiều công cụ và tài nguyên hỗ trợ các tác vụ như phân tích cú pháp, xác thực ngữ nghĩa và trích xuất thông tin. Thư viện này rất hữu ích cho nhà nghiên cứu, giáo viên và lập trình viên trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP), nhờ vào sự đa dạng và phong phú của các module mà nó cung cấp. NLTK cho phép người dùng thực hiện các thao tác từ đơn giản như tách câu, tách từ đến các thao tác phức tạp hơn như phân tích ngữ nghĩa và xây dựng mô hình ngôn ngữ.

Ngoài ra, NLTK còn bao gồm các thuật toán mạnh mẽ như phân tích từ loại (part-of-speech tagging), nhận diện thực thể (named entity recognition) và lemmatization. Nó đi kèm với bộ dữ liệu phong phú và các tài nguyên học tập, giúp người dùng dễ dàng tiếp cận và thực hành. Với cộng đồng hỗ trợ rộng lớn và tài liệu chi tiết, NLTK đã trở thành công cụ phổ biến trong nghiên cứu và ứng dụng NLP.

1.3.5. Matplotlib và Seaborn

Matplotlib và Seaborn là hai thư viện phổ biến trong Python, chuyên dụng cho việc trực quan hóa dữ liệu, mỗi thư viện có những đặc điểm và tính năng riêng biệt. Matplotlib là một thư viện mạnh mẽ và linh hoạt, ra đời từ năm 2003, cho phép người dùng tạo ra nhiều loại đồ thị khác nhau như đồ thị đường, đồ thị cột, biểu đồ phân tán và biểu đồ ba chiều. Nó cung cấp nhiều tùy chọn để tùy chỉnh hình ảnh, từ màu sắc, nhãn, tiêu đề cho đến kiểu dáng, giúp người dùng có thể kiểm soát chi tiết về cách thức trình bày dữ liệu. Với khả năng tạo ra các đồ thị có chất lượng cao, Matplotlib đã trở thành một công cụ hữu ích cho các nhà khoa học dữ liệu, nhà phân tích và các chuyên gia trong nhiều lĩnh vực.

Trong khi đó, Seaborn, ra đời vào năm 2012 và được xây dựng trên nền tảng của Matplotlib, tập trung vào việc làm cho việc trực quan hóa dữ liệu trở nên dễ dàng và trực quan hơn. Seaborn cung cấp nhiều kiểu đồ thị nâng cao và các phương pháp thống kê, cùng khả năng tích hợp tốt với các cấu trúc dữ liệu của Pandas. Thư viện này đặc biệt mạnh mẽ trong việc tạo ra các đồ thị phức tạp như đồ thị phân phối, biểu đồ kết hợp và các biểu đồ tương quan, nhờ vào cú pháp đơn giản và trực quan. Seaborn cũng cung cấp các bảng màu đẹp mắt và tính năng tự động hóa việc tạo các đồ thị đẹp từ các tập dữ liệu lớn, giúp người dùng dễ dàng khám phá và phân tích dữ liệu.

1.3.6. Biểu thức chính quy

Biểu thức chính quy (regular expressions) là một công cụ mạnh mẽ cho phép người dùng mô tả các mẫu tìm kiếm trong chuỗi văn bản một cách chính xác và hiệu quả. Chúng thường được sử dụng trong các tác vụ như tìm kiếm, thay thế và xác thực chuỗi theo các quy tắc cụ thể. Trong Python, module "re" cung cấp nhiều hàm hữu ích để làm việc với biểu thức chính quy, cho phép người dùng dễ dàng thực hiện các thao tác trên chuỗi. Các ký tự đặc biệt như ^ (bắt đầu chuỗi), $ (kết thúc chuỗi), \d (số), và \w (ký tự chữ và số) là những thành phần quan trọng trong việc xây dựng biểu thức chính quy, giúp xác định các mẫu cần tìm.

Một trong những ưu điểm lớn nhất của biểu thức chính quy là khả năng tiết kiệm thời gian và công sức trong việc xử lý văn bản. Thay vì phải viết các đoạn mã phức tạp để phân tích chuỗi, người dùng chỉ cần định nghĩa một mẫu đơn giản để thực hiện các tác vụ như tìm kiếm, thay thế và phân tách. Điều này làm cho biểu thức chính quy trở thành một công cụ hữu ích trong nhiều lĩnh vực, từ phát triển phần mềm đến phân tích dữ liệu. Nhờ sự hỗ trợ từ module re trong Python, người dùng có thể dễ dàng áp dụng các kỹ thuật này để giải quyết các bài toán phức tạp liên quan đến văn bản, nâng cao hiệu quả và tính chính xác trong các tác vụ xử lý chuỗi.

1.3.7. Next.js

Next.js là một framework mã nguồn mở dựa trên React, được thiết kế để xây dựng các ứng dụng web hiện đại và hiệu quả. Với tính năng hỗ trợ server-side rendering (SSR) và static site generation (SSG), Next.js giúp cải thiện hiệu suất và khả năng tối ưu hóa SEO cho các trang web. Điều này cho phép người dùng tạo ra các trang web có tốc độ tải nhanh và trải nghiệm người dùng tốt hơn. Ngoài ra, Next.js cung cấp khả năng chia nhỏ mã (code splitting) tự động, giúp giảm kích thước tải ban đầu và tối ưu hóa thời gian tải trang.

Framework này cũng đi kèm với nhiều tính năng hữu ích khác như hỗ trợ API routes, dễ dàng cấu hình và tích hợp với các dịch vụ bên ngoài. Với hệ sinh thái phong phú và cộng đồng phát triển năng động, Next.js đã nhanh chóng trở thành một lựa chọn hàng đầu cho các nhà phát triển khi xây dựng ứng dụng web. Nhờ vào sự linh hoạt và khả năng mở rộng, Next.js cho phép phát triển các ứng dụng phức tạp, từ trang web tĩnh đến các ứng dụng web động, đáp ứng nhu cầu ngày càng cao của thị trường.

1.3.8. FastAPI

FastAPI là một framework mã nguồn mở dành cho phát triển API với Python, nổi bật nhờ hiệu suất cao và khả năng sử dụng dễ dàng. Được xây dựng trên nền tảng Starlette cho xử lý bất đồng bộ và Pydantic cho xác thực dữ liệu, FastAPI cho phép tạo ra các ứng dụng web nhanh chóng và hiệu quả với cấu trúc rõ ràng. Một trong những ưu điểm lớn của FastAPI là khả năng tự động tạo tài liệu API thông qua OpenAPI và JSON Schema, giúp lập trình viên dễ dàng xem và thử nghiệm các endpoint mà không cần viết tài liệu thủ công.

Ngoài ra, FastAPI hỗ trợ xác thực, phân quyền và xử lý bất đồng bộ, cho phép xây dựng các ứng dụng có khả năng mở rộng cao. Với tính năng gợi ý kiểu dữ liệu trong Python, FastAPI không chỉ giúp giảm thiểu lỗi mà còn cải thiện trải nghiệm lập trình, nhanh chóng trở thành lựa chọn hàng đầu cho việc xây dựng các ứng dụng API hiện đại.

1.3.9. Visual Studio Code

Visual Studio Code (VSCode) là một trình soạn thảo mã nguồn mở và miễn phí được phát triển bởi Microsoft. Được ra mắt vào năm 2015, VSCode nhanh chóng trở thành một trong những công cụ phổ biến nhất cho các lập trình viên. Điểm nổi bật nhất của VSCode là sự linh hoạt và mở rộng, cho phép người dùng tùy chỉnh và mở rộng chức năng thông qua các tiện ích mở rộng và khả năng cấu hình linh hoạt.

Một điểm mạnh khác của VSCode là giao diện người dùng thanh lịch và trực quan, kết hợp với khả năng tự động hoàn thiện code và gợi ý thông minh giúp tăng hiệu suất lập trình. VSCode hỗ trợ nhiều ngôn ngữ lập trình và công nghệ phổ biến, từ JavaScript và HTML/CSS đến React hay Vue.js, cung cấp cho người dùng một trải nghiệm đồng nhất khi làm việc với nhiều dự án khác nhau.

Ngoài ra, VSCode còn tích hợp mạnh mẽ với các công cụ kiểm tra mã nguồn và quản lý phiên bản như Git, giúp người dùng theo dõi lịch sử thay đổi mã và làm việc nhóm một cách hiệu quả. Tính linh hoạt, hiệu suất và cộng đồng hỗ trợ đông đảo là những lý do khiến VSCode trở thành một trong những lựa chọn hàng đầu cho lập trình viên trên toàn thế giới.

CHƯƠNG 2:
PHÂN TÍCH VÀ XỬ LÝ DỮ LIỆU

2.1. Tổng quan về tập dữ liệu

Tập dữ liệu bao gồm 7613 mẫu, với 5 thuộc tính bao gồm:

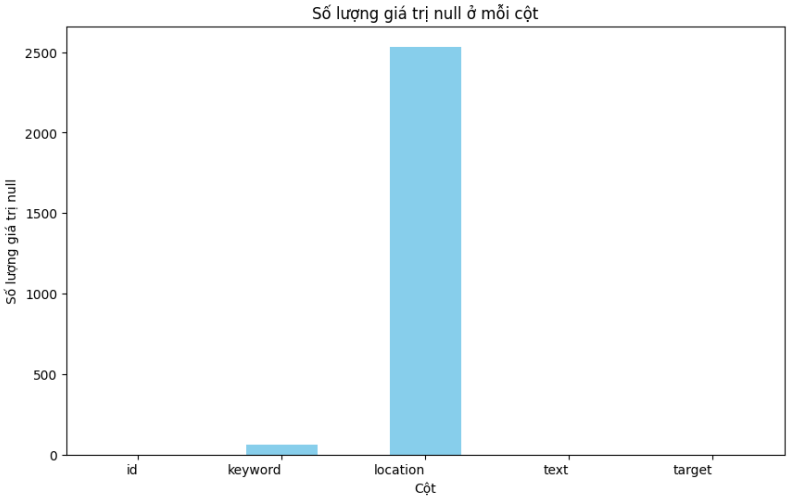
- ***id:*** Đây là khóa chính của tập dữ liệu, bao gồm một số nguyên lớn hơn 0 thể hiện cho một tweet duy nhất.
- ***keyword:*** Thể hiện cho một từ khóa có liên quan đến tweet, từ khóa này có thể không xuất hiện trong nội dung của tweet.
- ***location:*** Vị trí địa lý có liên quan đến tweet, phần lớn các tweet không có giá trị này.
- ***text:*** Là thuộc tính quan trọng nhất, thể hiện nội dung của tweet, có thể chứa thông tin liên quan đến thảm họa hoặc không.
- ***target:*** Giá trị nhị phân dùng để phân loại tweet đó có liên quan đến thảm họa hay không, với giá trị 1 là có liên quan và ngược lại với giá trị 0.

	id	keyword	location	text	target
0	1	NaN	NaN	Our Deeds are the Reason of this #earthquake M...	1
1	4	NaN	NaN	Forest fire near La Ronge Sask. Canada	1
2	5	NaN	NaN	All residents asked to 'shelter in place' are ...	1
3	6	NaN	NaN	13,000 people receive #wildfires evacuation or...	1
4	7	NaN	NaN	Just got sent this photo from Ruby #Alaska as ...	1
5	8	NaN	NaN	#RockyFire Update => California Hwy. 20 closed...	1
6	10	NaN	NaN	#flood #disaster Heavy rain causes flash flood...	1
7	13	NaN	NaN	I'm on top of the hill and I can see a fire in...	1
8	14	NaN	NaN	There's an emergency evacuation happening now ...	1
9	15	NaN	NaN	I'm afraid that the tornado is coming to our a...	1

Hình 1: Các dòng đầu tiên của tập dữ liệu thô

2.2. Phân tích tập dữ liệu

2.2.1. Các dữ liệu bị thiếu



	Null Count	Percentage
id	0	0.00
keyword	61	0.80
location	2534	33.29
text	0	0.00
target	0	0.00

Hình 2: Số lượng dữ liệu bị thiếu của mỗi cột dữ liệu

Ở các cột id, text và target không có giá trị null nào. Tuy nhiên ở cột location thì tỉ lệ giá trị null khá lớn (33.29%), với số lượng phần tử là 2534. Ngoài ra có 61 mẫu không có giá trị keyword, chiếm 0.8% trên tổng số các mẫu. Điều này có thể lí giải là do phần lớn người dùng không có thói quen thêm phần địa điểm vào tweet của mình, vì các thao tác như tìm kiếm và lựa chọn cũng mất thời gian. Tuy mạng xã hội có các công cụ AI thông minh nhằm tìm kiếm địa điểm dựa trên nội dung của tweet, chúng không thể hoạt động hiệu quả, bởi các địa điểm nó được gọi dưới dạng viết tắt hoặc theo tiếng lóng.

2.2.2. Các cột giá trị trong tập dữ liệu

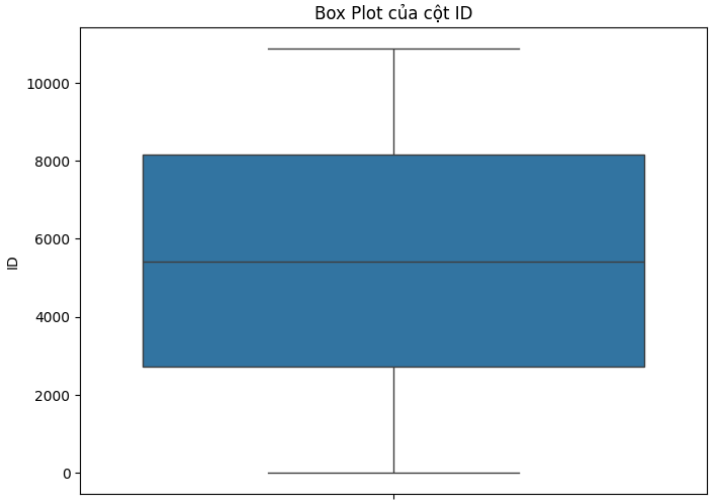
❖ **id**

Cột id chứa các giá trị số nguyên lớn hơn 0, dùng để định danh từng mẫu dữ liệu trong tập dữ liệu. Các giá trị id không liên tục mà ngẫu nhiên đứt quãng, khiến cho cột id không có giá trị để sử dụng khi huấn luyện mô hình

	id	keyword	location	text	target
0	1	NaN	NaN	Our Deeds are the Reason of this #earthquake M...	1
1	4	NaN	NaN	Forest fire near La Ronge Sask. Canada	1
2	5	NaN	NaN	All residents asked to 'shelter in place' are ...	1
3	6	NaN	NaN	13,000 people receive #wildfires evacuation or...	1
4	7	NaN	NaN	Just got sent this photo from Ruby #Alaska as ...	1
5	8	NaN	NaN	#RockyFire Update => California Hwy. 20 closed...	1

Hình 3: Các dòng đầu tiên trong tập dữ liệu thô

Khi hiển thị các giá trị của cột với box plot, có thể dễ dàng nhận thấy rằng: giá trị của cột id trải dài từ 1 đến 10813 và giá trị trung vị nằm ở khoảng 5550. Dữ liệu bị mất 3260 mẫu giá trị (so với thực tế chỉ có 7613 mẫu), tuy nhiên không bị mất quá nhiều mẫu ở một khoảng nào đó mà phân phối mất đều.



Hình 4: Boxplot của cột id

❖ **keyword**

Cột keyword chứa các chuỗi dữ liệu, thường là 1 hoặc đến 2 từ, biểu thị cho nội dung, chủ đề hoặc từ khóa thảm họa có liên quan đến tweet. Có tổng cộng 221 keyword khác nhau và trong đó có 61 tweets không có keyword nào:

```
unique_keywords = data['keyword'].nunique()
print(f"Tổng số lượng keywords khác nhau: {unique_keywords}")
```

✓ 0.0s

```
Tổng số lượng keywords khác nhau: 221
```

```
missing_keyword_count = data['keyword'].isnull().sum()
print(f"Số lượng tweet không có keyword: {missing_keyword_count}")
```

✓ 0.0s

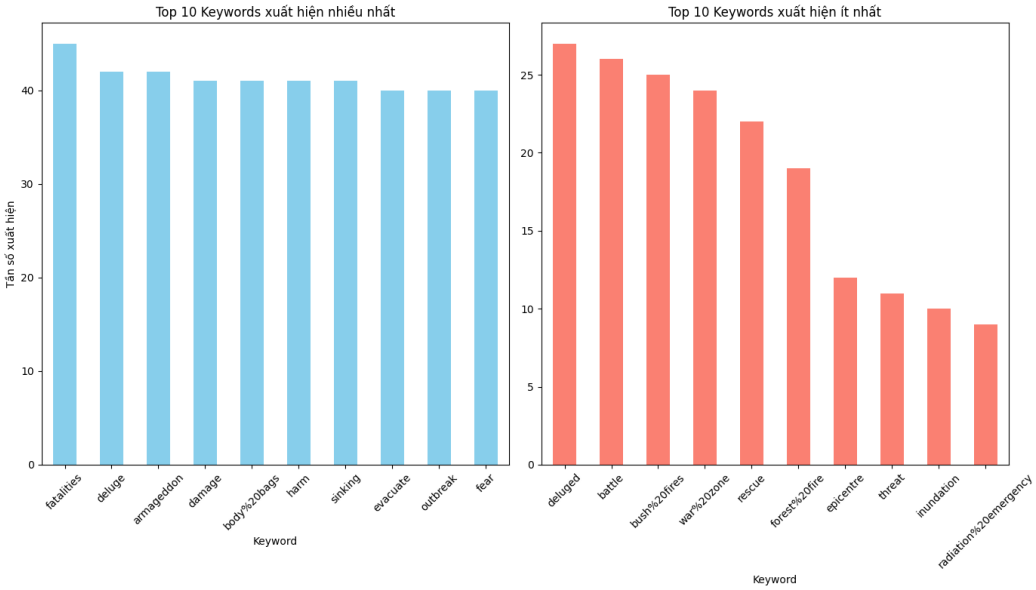
```
Số lượng tweet không có keyword: 61
```

Hình 5: Số lượng các keywword và các tweet không có keyword

Các keywỏd bao gồm các thuật ngữ như "derailment" (trật đường ray), "famine" (nạn đói), "earthquake" (động đất) và các sự kiện liên quan đến thảm họa tự nhiên và nhân tạo khác nhau.

Hình 6: Biểu đồ tag cloud đối với cá keywword

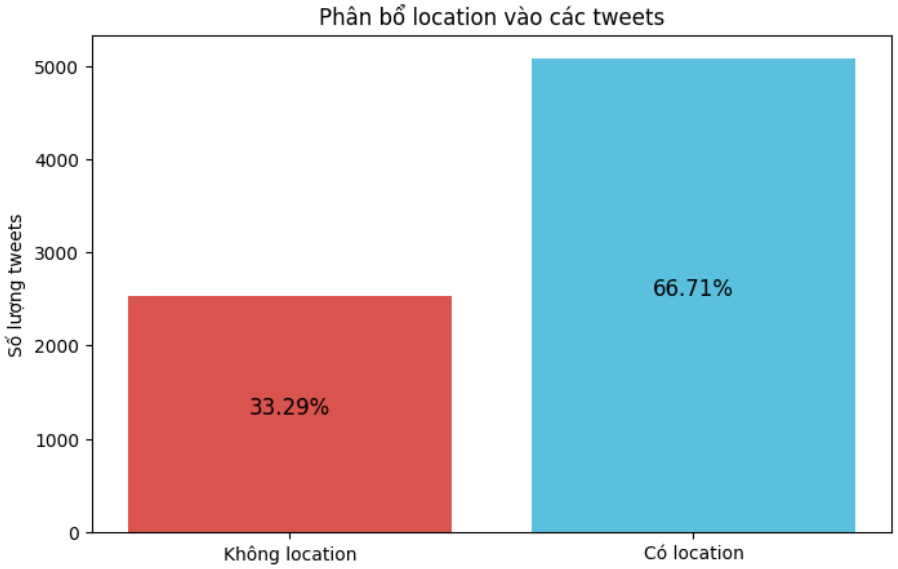
Trong số các keyword, cụm từ xuất hiện nhiều nhất là: "fatalities" (người thiệt mạng, "deluge" (trận lụt lớn) và "armageddon" (trận chiến đấu quyết liệt cuối cùng). Các cụm từ xuất hiện ít nhất là "radiation emergency" (các trường hợp khẩn cấp về phóng xạ), "inudation" (ngập lụt), "threat" (mối nguy hiểm).



Hình 7: Top 10 các keywword xuất hiện nhiều nhất và ít nhất

❖ **location**

Có 2534 mẫu không có location, chiếm gần 33.28% trong tổng số



Hình 8: Phân bổ giá trị của cột location vào các tweets

Các giá trị của keyword đôi khi xuất hiện các giá trị nhiễu như:

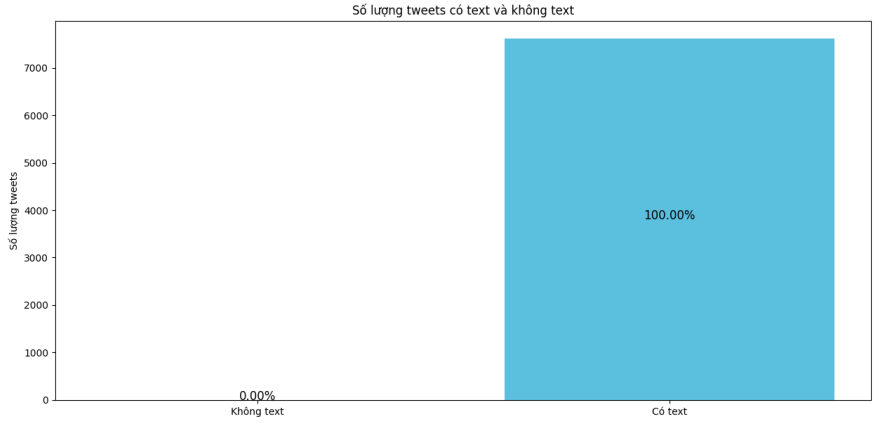
```
1324,http://www.amazon.com/dp/B00HR
1325,"New York, NY"
1326,[ kate + they/them + infp-t ]
```

Hình 9: Ví dụ về giá trị nhiễu trong cột location

Các giá trị của location không đóng góp nhiều trong việc dự báo tweet có có liên quan đến thảm họa hay không, vì vậy đối với cột giá trị này chúng ta nên bỏ qua chúng.

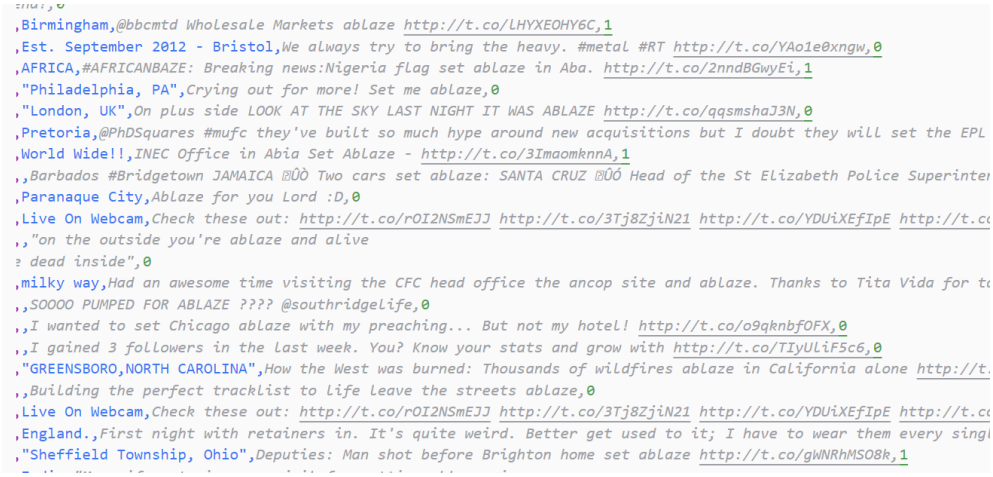
❖ **text**

Đây là phần quan trọng nhất của mỗi mẫu, cung cấp thông tin về nội dung của một tweet. Tất cả các tweet đều có cột text



Hình 10: Số lượng tweet có text và không có text

Trong text có thể xuất hiện các đối tượng gây nhiễu như: handler của người dùng, các đường link URL, các hashtags, các kí tự đặc biệt, các kí tự xuống dòng, các từ thuộc các ngôn ngữ khác, các kí tự unicode không nằm trong bảng mã ASCII.

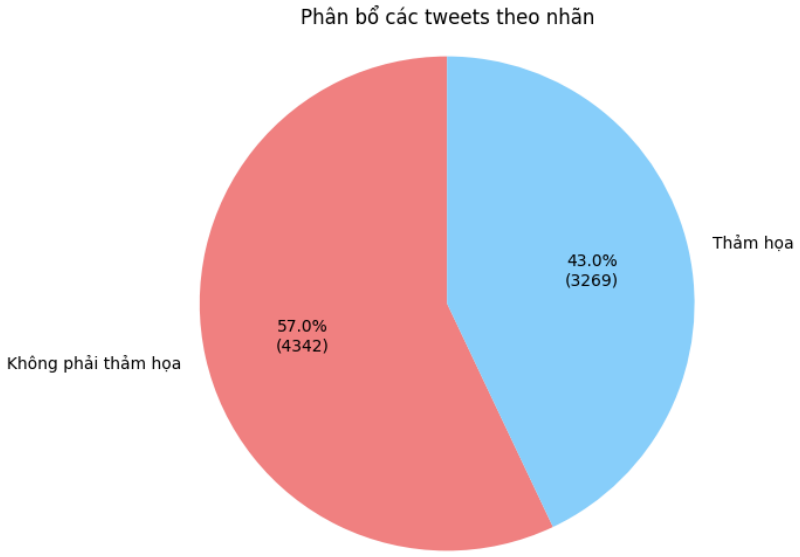


Hình 11: Các mẫu text có giá trị nhiễu

Ngoài ra các từ trong text đôi khi ở các dạng viết tắt như “you’re” (you are), “they’ve” (they have) hoặc các từ tiếng lóng như “bet”, “cringe”. Một số khác thì xuất hiện ở các dạng hình thái chia thì của động từ như “visitting” (visit) hoặc danh từ hóa (follower – follow).

❖ **target**

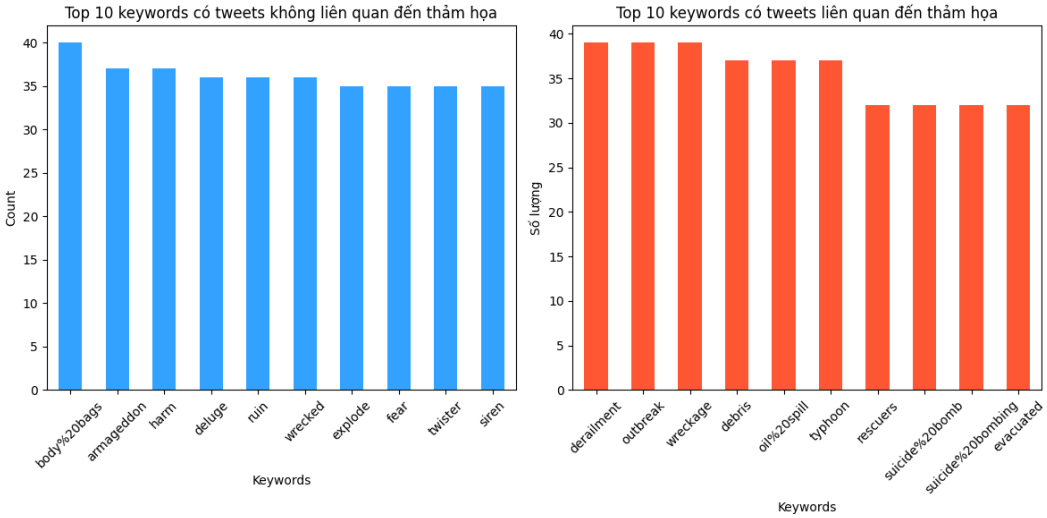
Dữ liệu khá cân bằng khi số lượng của các tweets thảm họa và các tweets không phải thảm họa không lệch nhau không quá lớn nhiều, với 3269 tweets thảm họa, chiếm 43% và không phải thảm họa với 4332 tweets (57%)



Hình 12: Phân bổ các tweets dựa theo nhãn

2.2.3. Mối quan hệ giữa các cột giá trị

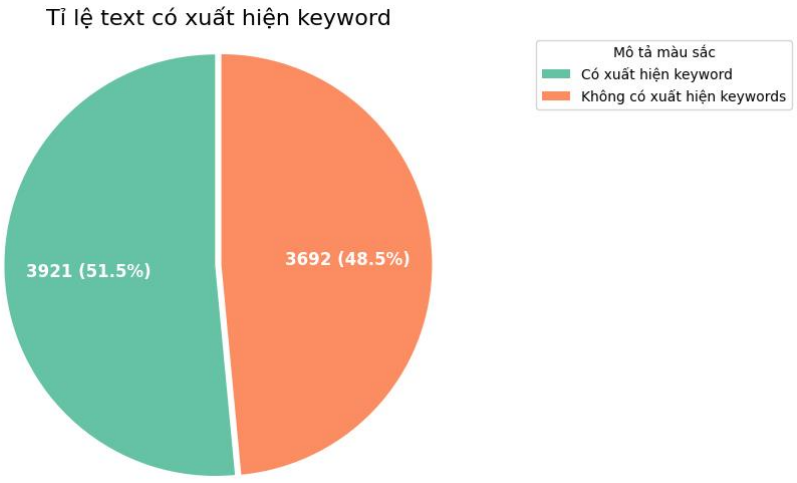
❖ **keyword và target**



Hình 13: Top 10 các keyword có liên quan đến thảm họa và ngược lại

Trong số các tweet có liên quan đến thảm họa, "derailment" (trật đường ray), "outbreak" (sự bùng nổ) và "wreckage" (mảnh vỡ) có cùng số lượng tweets nhiều nhất với 39 tweets. Ngược lại, đối với các tweet không liên quan đến thảm họa, cụm từ "body bags" (túi xác) xuất hiện sai nhiều nhất với 40 lần, tiếp theo đó là "armageddon" (trận chiến đấu quyết liệt cuối cùng) và "harm" (gây hại) với lần lượt là 36 lần. Lí do các từ khóa trên xuất hiện trên các tweets không có liên quan là do đây các chủ đề phổ biến trong game hoặc các truyện sci-fi, dẫn đến các tweets trên dễ bị nhầm lẫn là liên quan đến thảm họa trong thực tế.

❖ **keyword và text**



Hình 14: Tỉ lệ các keyword xuất hiện trong text

Gần 50% nội dung của tweet không xuất hiện keywords của chính nó. Đối với các tweet có xuất hiện keyword, chúng thường ở dạng bình thường hoặc ở dưới các hashtag, hoặc dưới username của người dùng. Ngoài ra, một số tweet có keyword xuất hiện ở các dạng từ khác nhau ("crashes" → "crash"). Tuy nhiên đối với một số từ có hình thái từ khác nhau thì sẽ không nhận ra được keyword ("volcano" → "volcanic").

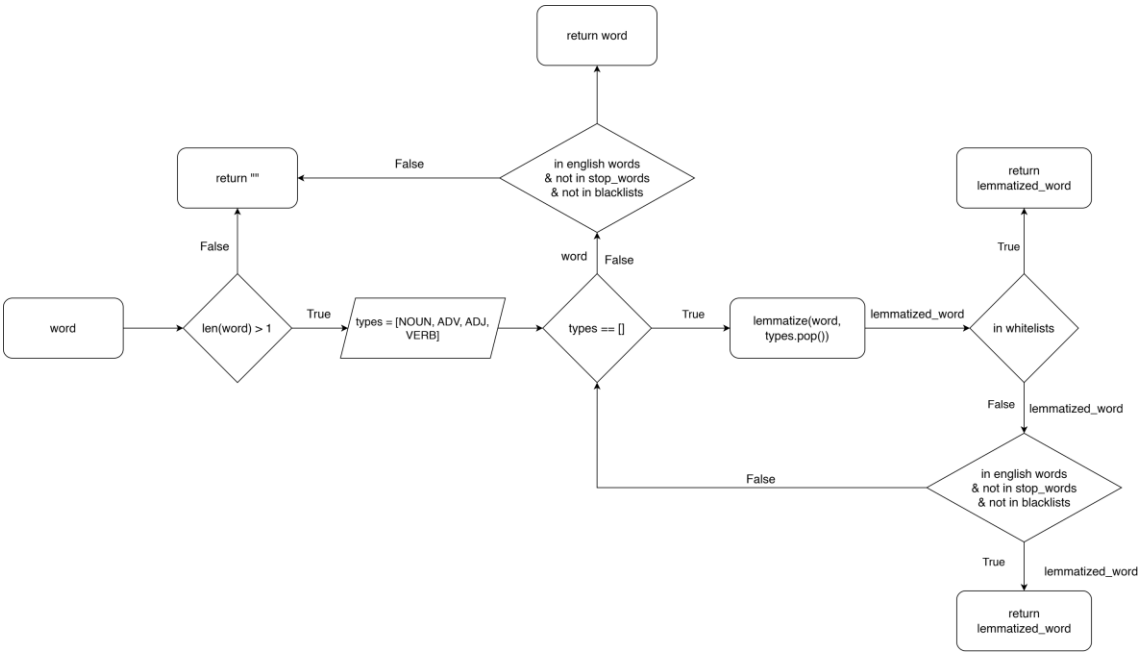
2.3. Xử lý dữ liệu

Dựa trên cách phân tích trên về dữ liệu, nhóm quyết định sẽ bỏ một số cột giá trị để tránh ảnh hưởng đến quá trình huấn luyện và kết quả, cũng như giảm thiểu độ phức tạp của dữ liệu.

- **id**: vì các giá trị trong cột id không liên tục (do mất mát dữ liệu), vì vậy cách tốt nhất là bỏ hoàn toàn cột này và tiến hành reset index của Dataframe
- **location**: số lượng tweets không có giá trị locations này rất cao, chiếm hơn 33%. Hơn nữa, do vị trí địa lý của tweets không ảnh hưởng lớn đến ý nghĩa của nội dung tweets, chúng ta sẽ tiến hành bỏ cột dữ liệu này.
- **keyword**: các keyword đều có ý nghĩa liên quan đến thảm họa. Tuy vậy, khi phân tích ở trên, một số keyword xuất hiện rất nhiều ở các tweets không liên quan đến thảm họa. Vì để tránh làm nhiễu loạn mô hình khi huấn luyện, cột keyword sẽ không được dùng để huấn luyện mô hình.

Sau quá trình tinh chỉnh ban đầu, dữ liệu được đưa đi huấn luyện sẽ gồm 2 cột giá trị, với 1 cột đặc trưng là text và 1 cột nhãn là target.

2.3.1. Thiết kế hàm lemmatize_word



Hình 15: Mô hình xử lý của hàm lemmatize_word

Khi xử lý các từ vựng tiếng Anh, chúng ta đôi khi bắt gặp các từ vựng điều biểu diễn ý nghĩa cho một từ, nhưng ở các dạng từ vựng khác nhau (ví dụ "create"/"creation"/"creative"). Khi vector hóa văn bản, nếu không xử lý các trường hợp này thì dễ dẫn đến trường hợp xuất hiện quá nhiều từ cùng biểu diễn cho một ý nghĩa. Vì vậy cần phải thực hiện các phép biến đổi và chuẩn hóa các từ về dạng nguyên bản của chúng.

Thư viện nltk có cung cấp các module lemmatize hỗ trợ việc này. Hàm này có nhiệm vụ biến đổi một từ về dạng nguyên của nó, ví dụ: "running" → "run", "mice" → "mouse", "interested" → "interest". Khi chuyển đổi từ vựng như thế, không ít trường hợp sẽ xuất hiện biến đổi sai, như "was" → "wa". Ngoài ra, một số từ vựng mới xuất hiện gần đây, hoặc các từ ghép thường sẽ không được biến đổi và chỉ trả về kết quả rỗng (ví dụ "meltdown" hay "hellfire"). Vì vậy cần xây dựng một hàm chuyển đổi hiệu quả và có thể xử lí các trường hợp nêu trên.

Hàm lemmatize_word được thiết kế để giải quyết vấn đề này. Đầu vào của hàm là một từ, một token trong câu. Đầu tiên, nếu đây chỉ là một kí tự, ngay lập tức trả về chuỗi rỗng để xử lí các dữ liệu nhiễu, hoặc stop word. Ngay sau đó, hàm này sẽ lần lượt thử lemmatize từ đầu vào với loại từ lần lượt theo thứ tự lần lượt là: Verb (động từ), Adverb (trạng từ), Adjective (tính từ) và cuối cùng là Noun (danh từ). Danh từ được xử lí cuối cùng vì một số từ kết thúc với kí tự "s" khi xử lí với loại từ này thì dẫn đến việc thành một từ không có nghĩa. Trong bất kì trường hợp lemmatize với loại từ nào, nếu đây là từ nằm trong danh sách cho phép (danh sách các từ ghép không thể lemmatize được) hoặc thỏa điều kiện: là từ thuộc tiếng Anh, không phải là stop word và không nằm trong danh sách từ bị loại thì hàm sẽ trả về kết quả của việc lemmatize ngay lập tức. Nếu sau bước này, từ gốc ban đầu vẫn thỏa 1 trong 2 điều kiện trên thì ta sẽ trả về từ gốc ban đầu. Cuối cùng, nếu không điều kiện nào thỏa mãn, hàm chỉ đơn giản trả về chuỗi rỗng.

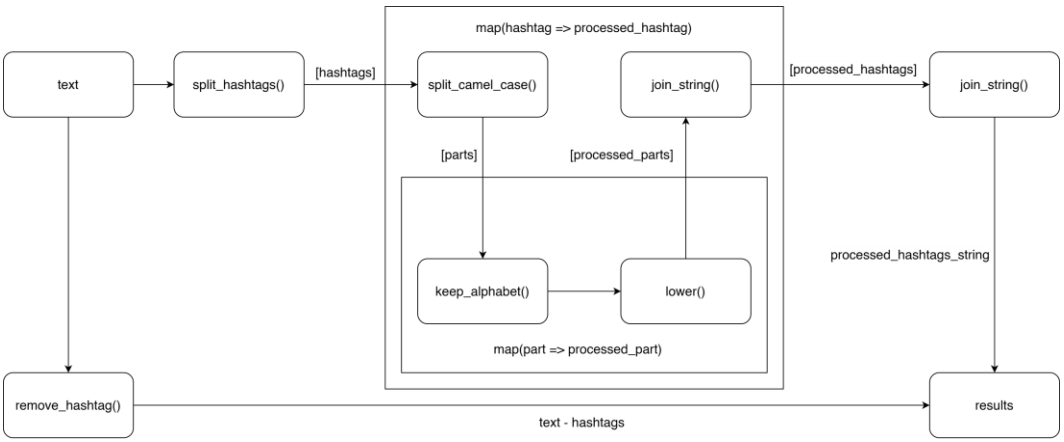
2.3.2. Thiết kế hàm process_hashtag

Các hashtag là danh sách các từ hoặc cụm từ đứng sau dấu "#" trong nội dung các tweets. Các hashtags đôi khi chứa các thông tin quan trọng, ảnh hưởng đến nội dung phân lớp của bài toán, ví dụ:

id	keyword	location	text	target
1			Our Deeds are the Reason of this #earthquake May ALLAH Forgive us all	1

Hình 16: Ví dụ về hashtag chứa thông tin trong text

Nếu khi xử lí chỉ đơn thuần là bỏ các hashtag, điều này có thể dẫn đến văn bản hoàn toàn mất ý nghĩa. Trong trường hợp trong tweets này, cụm từ "earthquakes" (động đất) trong hashtag giúp cho tweets này có ý nghĩa là cầu nguyện cho thảm họa, thay vì cầu nguyện như bình thường.



Hình 17: Mô hình xử lý của hàm process_hashtag

Hàm process_hashtag có nhiệm vụ xử lí các hashtags bên trong các text. Đầu vào là một text trong tweet và đầu ra là một text với nội dung không còn hashtag và có thêm các từ đã được xử lí từ nội dung của hashtag. Đầu tiên hàm sẽ thực hiện việc xóa bỏ các hashtag hiện tại trong text và dời sang một cột mới để xử lí. Sau đó các hashtag sẽ được tách lấy phần nội dung, và tách ra từng từ dựa trên CamelCase (ví dụ: "AMonadIsJustAMonoid") ra thành từng từ riêng biệt. Từ này sẽ được xử lí để bỏ các số, kí tự đặc biệt để đầu ra chỉ còn là các chữ nằm trong bảng chữ cái alphabet. Các từ này sẽ được gộp lại thành một câu, và từ các câu của hashtag gộp lại sẽ ra được câu kết quả chung của toàn bộ hashtag trong text. Sau đó cột giá trị tạm này sẽ được gộp chung lại với cột text gốc đã được bỏ các hashtag.

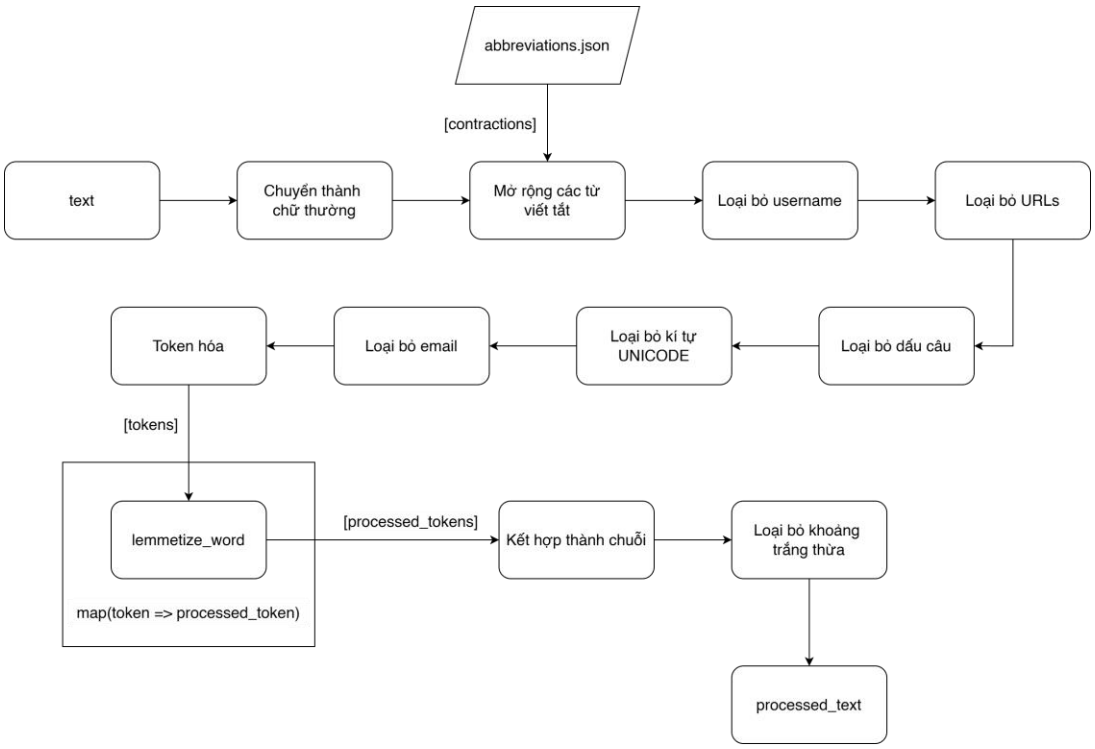
2.3.3. Thiết kế hàm process_text

Cột text là đặc trưng chính của tweets dùng để phân loại các tweets có liên quan đến thảm họa hay không. Như đã phân tích ở trên, văn bản trong cột này chứa rất nhiều thông tin nhiễu, hoặc không liên quan, từ các đường link URL đến các kí tự lỗi UNICODE.

SANTA CRUZ ÛÓ Head of the St Elizabeth Police Superintendent Lanford Salmon has r ... - http://t.co/vplR5Hka2u http://t.co/SxHW2TNNLf	0
Police: Arsonist Deliberately Set Black Church In North CarolinaåÊAblaze http://t.co/pcXarbH9An	1
Noches El-Bestia '@Alexis_Sanchez: happy to see my teammates and training hard ?? goodnight gunners.?????? http://t.co/uc4j4jHvGR'	0
#Kurds trampling on Turkmen flag later set it ablaze while others vandalized offices of Turkmen Front in #Diyala http://t.co/4IzFdYC3cg	1
TRUCK ABLAZE : R21. VOORTREKKER AVE. OUTSIDE OR TAMBO INTL. CARGO SECTION. http://t.co/8kscqKfKkF	1
Set our hearts ablaze and every city was a gift And every skyline was like a kiss upon the lips @ÛÒ_ https://t.co/cYoMPZ1A0Z	0

Hình 18: Ví dụ về các giá trị text bị nhiễu

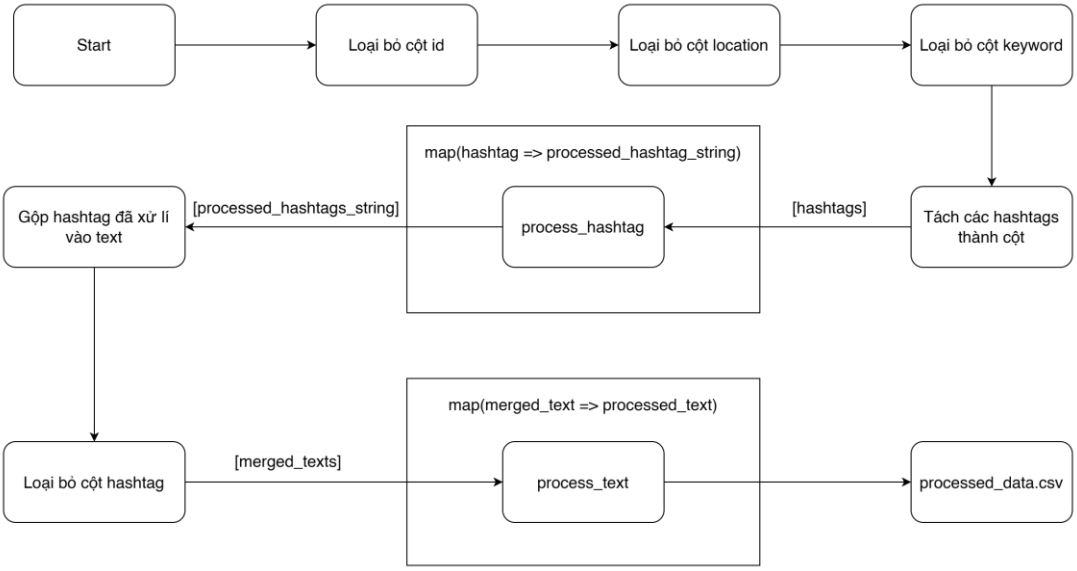
Do đó, cần phải xây dựng một pipeline hoàn chỉnh để xử lý các cụm từ gây nhiễu và các ký tự đặc biệt, đồng thời chuẩn hóa từ vựng một cách nhất quán và loại bỏ các stop word nhằm tối ưu hóa dữ liệu ngôn ngữ trước khi đưa vào các bước xử lý tiếp theo.



Hình 19: Mô hình xử lý của hàm process_text

Hàm process_text thực hiện nhiều bước để làm sạch, loại bỏ các yếu tố gây nhiễu và chuẩn hóa văn bản. Đầu tiên, toàn bộ văn bản được chuyển thành chữ thường để đảm bảo nhất quán về mặt chữ. Sau đó, các từ viết tắt trong văn bản được mở rộng thông qua việc duyệt qua danh sách các từ viết tắt đã định nghĩa trước. Tiếp theo, hàm này sẽ thực hiện xóa các handler của người dùng (các chuỗi bắt đầu bằng ký tự "@" với ý nghĩa biểu thị tên người dùng) và các link URL. Văn bản cũng được loại bỏ dấu câu, đồng thời các ký tự không thuộc mã ASCII sẽ bị xóa để chỉ giữ lại các ký tự tiếng Anh chuẩn. Để tăng tính bảo mật, các địa chỉ email cũng được loại bỏ bằng cách sử dụng biểu thức chính quy tương ứng để so khớp. Sau khi văn bản được tách thành các token, từng token được đưa về dạng gốc bằng cách áp dụng hàm lemmatize_word đã được định nghĩa ở mục 2.3.1. Các token đã được lemmatize sẽ kết hợp lại thành một chuỗi duy nhất. Cuối cùng, hàm tiến hành loại bỏ các khoảng trắng thừa và đảm bảo chuỗi không có khoảng trắng ở đầu hoặc cuối. Kết quả trả về là một chuỗi văn bản đã được làm sạch và chuẩn hóa hoàn chỉnh.

2.3.4. Mô hình tổng quan các bước tiền xử lý dữ liệu



Hình 20: Mô hình tổng quan của pipeline tiền xử lý dữ liệu

Đầu tiên, chúng ta sẽ loại bỏ các cột "id", "location" và "keyword" khỏi dữ liệu. Tiếp theo, các hashtag trong cột text sẽ được tách ra thành một cột riêng và xử lý theo cách đã nêu ở mục 2.3.2. Sau khi xử lý, các hashtag này sẽ được gộp lại vào dữ liệu text và loại bỏ cột hashtag. Cuối cùng, các giá trị trong cột text sẽ được làm sạch và chuẩn hóa như đã mô tả ở mục 2.3.3. Kết quả sẽ là file processed_data.csv với hai cột là text và target.

first night retainer quite weird better get use wear every single night next year least	0
deputy man shoot home set ablaze	1
man wife get six year jail set ablaze niece	1
head police superintendent salmon	0
police arsonist deliberately set black church north	1
happy see teammate train hard gunner	0
trample flag later set ablaze vandalize office front	1
truck ablaze ave outside tambo cargo section	1
set hearts ablaze every city gift every like kiss upon lip	0
sky ablaze tonight expect fill sunset shot know peep	0

Hình 21: Dữ liệu sau khi đã tiền xử lý

CHƯƠNG 3:
HUẤN LUYỆN MÔ HÌNH

3.1. Trích xuất đặc trưng

Trước khi có thể huấn luyện mô hình, tập dữ liệu văn phải phải được trích xuất các đặc trưng, vector hóa, biến từ dạng chữ thành những con số để có thể sử dụng những phép toán phức tạp để huấn luyện. Ở trong bài báo cáo nào, chúng ta sử dụng 3 phương pháp chính để vector hóa văn bản là: TF-IDF, Word2Vec, Bag of words.

3.1.1. TF – IDF

TF – IDF hay Term Frequency – Inverse Document Frequency là một thống kê số học nhằm phản ánh tầm quan trọng của một từ đối với toàn bộ văn bản, cụ thể ở đây là tầm quan trọng của một từ trong toàn bộ đoạn Tweet. Giá trị TF-IDF của một từ sẽ cao hơn khi từ đó xuất hiện nhiều trong một văn bản, nhưng giá trị này sẽ bị điều chỉnh nếu một từ thường xuyên xuất hiện trong nhiều tài liệu, thì tầm quan trọng của nó sẽ giảm xuống. TF-IDF là một trong những phương pháp tính trọng số phổ biến nhất hiện nay.

Giá trị TF-IDF được tính như sau:

$$TFIDF(t,d,D) = TF(t,d)\ x\ IDF(t,D)$$

TF - tần số xuất hiện của 1 từ trong 1 văn bản.

$$\text{TF(t,d)} = \frac{f(t,d)}{\max\{f(w,d) : w \in d\}},\quad 0 \le tf(t,d) \le 1$$

Trong đó:

- $\boldsymbol{f(t,d)}$ - số lần xuất hiện từ t trong văn bản **d**.
- $\boldsymbol{max\{f(w,d) : w \in d\}}$ - số lần xuất hiện nhiều nhất của một từ bất kỳ trong văn bản.

IDF - Tần số nghịch của 1 từ trong tập văn bản.

$$IDF(t,D) = log\frac{|D|}{|\{d \in D : t \in d\}|}$$

Trong đó:

- $\boldsymbol{|D|}$ - tổng số văn bản**.**
- $\boldsymbol{|\{d \in D : t \in d\}|}$ - số tài liệu chứa từ *t.*

3.1.2. Word2Vec

Word2Vec thường sử dụng một mạng nơ-ron để học cách ánh xạ từ sang vector. Khi mô hình được huấn luyện, một ma trận từ điển sẽ được tạo ra với mỗi từ được ánh xạ đến một vector trong không gian nhiều chiều. Các từ có ngữ

nghĩa tương tự sẽ có vector gần nhau trong không gian này hay nói cách khác khoảng cách Euclid của 2 vector là nhỏ.

3.1.3. Bag of words

Khác với 2 phương pháp trên, Bag of Words chỉ đơn giản là đếm số lần xuất hiện của từ trong văn bản, không quan tâm đến thứ tự hay ngữ cảnh. Bag of word thực hiện theo các bước sau đây:

- Xây dựng từ điển: tạo một danh sách tất cả các từ riêng biệt trong tập văn bản.
- Mỗi văn bản được chuyển hóa thành một vector nhiều chiều. Mỗi phần tử trong vector là số lượng từ xuất hiện trong văn bản đó.
- Sau khi mỗi văn bản đã được biểu diễn thành một vector, ta có thể so sánh các văn bản này để đánh giá mức độ tương tự nhau dựa vào vector của chúng.

3.2. Các mô hình huấn luyện

3.2.1. KNN

KNN là một mô hình tính toán dựa trên khoảng cách của các điểm dữ liệu. KNN xem xét k (thường là số lẻ 3, 5, 7, 9, …) điểm dữ liệu gần nhất trong không gian đặc trưng để dự đoán lớp cho một điểm dữ liệu mới đến. Đầu tiên, tính khoản cách (thường sử dụng Euclid) từ điểm cần phân loại đến tất cả các điểm khác trong dữ liệu huấn luyện, sau đó chọn k điểm gần nhất, lớp nào xuất hiện nhiều nhất trong k điểm này thì điểm dữ liệu mới đến sẽ được gán cho lớp đó. KNN chỉ thực hiện tính toán mà không có bước học tập rõ ràng trên dữ liệu. Ở đây, chúng ta sẽ sử dụng KNN với k = 7, tức là sẽ dự đoán một điểm dữ liệu dựa trên 7 điểm láng giềng gần nhất của điểm dữ liệu đó.

3.2.2. Naïve Bayes

Mô hình Naïve Bayes dưa trên lý thuyết Bayes, xác định xác suất một điểm thuộc về một lớp dựa trên đặc trưng của nó. Giả định rằng các đặc trưng độc lập với nhau (xác suất không ảnh hưởng bởi các đặc trưng khác). Naïve Bayes tính toán xác xuất của từng lớp dựa trên dữ liệu huấn luyện và gán điểm mới vào lớp có xác xuất cao nhất.

3.2.3. Logistic Regression

Logistic Regression áp dụng hồi quy để dự đoán một kết quả phân loại nhị phân. Đầu tiên, nó tính một tổ hợp tuyến tính của các đặc trưng đầu vào, sau đó sử dụng hàm sigmoid để biến đổi giá trị đó thành xác suất từ 0 đến 1. Khi xác suất này lớn hơn ngưỡng (thường là 0.5), mô hình sẽ gán điểm dữ liệu đó vào lớp 1, ngược lại thì vào lớp 0.

Mô hình có sử dụng các tham số sau:

- **C = 0.1:** tham số độ chính quy hóa. Giá trị C càng nhỏ sẽ tăng cường độ chính quy hóa, giúp mô hình tránh overfitting.
- **max_iter = 1000:** Đây là số lần lập tối đa của mô hình để đạt được độ hội tụ. Nếu số lần lặp đạt đến **max_iter** mà mô hình vẫn chưa hội tụ (không đạt được độ chính xác cần thiết), thì quá trình huấn luyện sẽ dừng lại.

3.2.4. SVM Linear kernel

Support Vector Machine (SVM) với kernel tuyến tính tìm siêu phẳng phân tách các điểm dữ liệu của hai lớp sao cho khoảng cách từ siêu phẳng đến các điểm gần nhất là lớn nhất. Quá trình này giúp tối đa hóa biên của siêu phẳng và giảm thiểu lỗi phân loại. Kernel được sử dụng ở đây chính là **"Linear"**

3.2.5. SVM Non-Linear kernel

Về bản chất thì văn bản là dữ liệu phi tuyến, thế nên ta có thể cân nhắc sử dụng SVM Non-Linear, cụ thể là sử dụng SVM với kernel Gaussian **(RBF)** để ánh xạ dữ liệu từ không gian ban đầu sang một không gian cao hơn, nơi các lớp có thể phân tách tuyến tính. Sau khi ánh xạ, SVM tìm một siêu phẳng tối ưu để phân chia dữ liệu trong không gian mới tương tự như SVM Linear.

3.2.6. Decision Tree

Decision Tree xây dựng một cây quyết định từ trên xuống. Tại mỗi nút, nó chọn đặc trưng tốt nhất để phân tách dữ liệu dựa trên các chỉ số như Entropy hoặc Gini, giúp giảm thiểu độ không tinh khiết của dữ liệu tại các nút con. Cây tiếp tục phân tách dữ liệu tại mỗi nhánh đến khi đạt được mức độ tinh khiết nhất định hoặc khi đạt đến chiều sâu tối đa đã định. Mỗi nhánh của cây dẫn đến một quyết định, và đầu ra cuối cùng được xác định bởi các điều kiện trong cây.

Mô hình sử dụng các tham số sau:

- **max_depth = 50:** Đây là độ sâu tối đa của cây quyết định.
- **min_samples_split = 4**: Đây là số lượng mẫu tối thiểu cần có để một node có thể tiếp tục phân chia. Nếu node có ít hơn 4 mẫu thì nó sẽ không chia nhỏ nữa.
- **criterion = 'gini':** Đây là hàm đánh giá để đo độ tinh khiết của node sử dụng chỉ số Gini.

3.2.7. Random Forest

Random Forest là một tập hợp của nhiều cây quyết định, mỗi cây được huấn luyện từ một tập dữ liệu khác nhau (lấy mẫu ngẫu nhiên với thay thế) và với một tập đặc trưng ngẫu nhiên tại mỗi bước phân tách. Khi có dự đoán mới, Random Forest sẽ lấy dự đoán của tất cả các cây và chọn kết quả theo nguyên tắc biểu quyết đa số (cho phân loại) hoặc trung bình (cho hồi quy). Cách tiếp cận này giúp giảm hiện tượng overfitting và tăng độ chính xác cho mô hình.

Mô hình sử dụng các tham số sau:

- **n_estimators = 200:** Đây là số lượng cây trong rừng ngẫu nhiên.
- **max_depth = 100:** Đây là độ sâu tối đa của mỗi cây trong rừng.
- **min_samples_split = 4:** Đây là số lượng mẫu tối thiểu cần có để một node có thể tiếp tục phân chia. Nếu node có ít hơn 4 mẫu thì nó sẽ không chia nhỏ nữa.
- **criterion = 'entropy':** Đây là hàm đánh giá được dùng để đo độ tinh khiết khi chia nhỏ node sử dụng hàm entropy.

3.2.8. Mô hình học sâu

Ngoài các mô hình truyền thống trên, chúng ta cũng cần sử dụng các mô hình học sâu như Feedforward neural network (FNN – mạng thần kinh truyền thẳng) hay RNN để dễ dàng so sánh cũng như đánh giá mô hình. Feedforward neural network sử dụng nhiều bộ lọc filter để lọc theo chiều dọc trên vector từ của câu. Giúp FNN tìm ra các đặc trưng quan trọng từ chuỗi từ ngắn gọn mà có thể hữu ích cho việc phân lớp. RNN có khả năng nắm bắt ngữ cảnh, với khả năng xử lý tuần tự thì RNN rất phù hợp cho việc xử lý văn bản, đặc biệt là với các câu có ngữ cảnh phức tạp và dài.

- Mô hình học sâu với TF-IDF có kiến trúc mô hình như sau:

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 64)	402,368
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 32)	2,080
dropout_1 (Dropout)	(None, 32)	0
dense_2 (Dense)	(None, 1)	33

Total params: 404,481 (1.54 MB)

Trainable params: 404,481 (1.54 MB)

Non-trainable params: 0 (0.00 B)

Hình 22. Kiến trúc mô hình học sâu với TF-IDF

- Mô hình học sâu với Bag of Words có kiến trúc mô hình như sau:

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 64)	402,432
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 32)	2,080
dropout_1 (Dropout)	(None, 32)	0
dense_2 (Dense)	(None, 1)	33

Total params: 404,545 (1.54 MB)

Trainable params: 404,545 (1.54 MB)

Non-trainable params: 0 (0.00 B)

Hình 23. Kiến trúc mô hình học sâu với Bag of Word

- Mô hình học sâu RNN với tầng Embedding có kiến trúc mô hình như sau:

Layer (type)	Output Shape	Param #
embedding (Embedding)	(None, 21, 64)	402,368
lstm (LSTM)	(None, 21, 64)	33,024
dropout (Dropout)	(None, 21, 64)	0
lstm_1 (LSTM)	(None, 64)	33,024
dropout_1 (Dropout)	(None, 64)	0
dense (Dense)	(None, 1)	65

Total params: 1,405,445 (5.36 MB)

Trainable params: 468,481 (1.79 MB)

Non-trainable params: 0 (0.00 B)

Optimizer params: 936,964 (3.57 MB)

Hình 24. Kiến trúc mô hình học sâu với RNN và Embedding

3.2.9. Mô hình pre-trained BERT

Bên cạnh các mô hình trên, mô hình BERT sử dụng pre-trained model để huấn luyện lại cũng được sử dụng để có thể so sánh với các mô hình khác. Sử dụng Tokenizer từ pre-trained model (bert-base-uncased). Do mô hình pre-trained BERT đã được huấn luyện trước đó, nên bước fine-tuning với tập dữ liệu mới tốn rất nhiều thời gian (3 tiếng cho 1 epochs). Nên ở đây chỉ huấn luyện mô hình với 1 epochs, batch_size = 8 và warmup_steps = 500. Trong quá trình huấn luyện, mô hình cũng sử dụng Wandb để monitoring quá trình huấn luyện cũng như các giá trị của máy như CPU, RAM,.. trong quá trình huấn luyện.

3.3. Nghi thức đánh giá

Để có thể đánh giá được mô hình tốt hay xấu, có thể sử dụng được hay không. Ta phải áp dụng kĩ thuật đánh giá mô hình. Cụ thể ở đây chúng ta sử dụng nghi thức Hold-out. Đối với nghi thức này, chúng ta chia tập dữ liệu ban đầu thành 2 phần là tập huấn luyện (80%) và tập kiểm tra (20%), dữ liệu sẽ được lấy một cách ngẫu nhiên và được xáo trộn để đảm bảo không bị lệch về một lớp nào đó.

Để có thể được đánh giá tổng quát hơn các mô hình, xác định chúng không phụ thuộc vào tập dữ liệu. Chúng ta sử dụng phương pháp cải tiến hơn của Hold-out chính là Repeated Hold-out. Với phương pháp này, tập dữ liệu sẽ được chia thành 2 phần sau đó huấn luyện, lưu lại kết quả, sau đó lại lấy dữ liệu ban đầu chia lại với tỉ lệ giống như trước đó nhưng đã bị xáo trộn, không giống với tập trước đó để tiếp tục huấn luyện mô hình. Quá trình trên có thể lập lại nhiều lần, cụ thể ở đây chúng ta lập lại 20 lần. Kết quả cuối cùng sẽ là trung bình kết quả của 20 lần chạy trên.

3.4. Kết quả huấn luyện và đánh giá mô hình

Sau khi áp dụng các quá trình trích xuất đặc trưng, chọn mô hình huấn luyện và áp dụng các nghi thức đánh giá để huấn luyện trên các mô hình: KNN, Naïve Bayes, Logistic Regression, SVM Linear, SVM Non-Linear, Decision Tree và Random Forest. Chúng ta thu được các kết quả sau:

❖ **Accuracy**

	KNN	Naives Bayes	Logistic Regression	SVM Linear	SVM Non-Linear	Decision Tree	Random Forest	Trung bình
TF-IDF	0.76	0.79	0.74	0.79	0.80	0.75	0.78	0.77
Word2Vec	0.65	0.54	0.56	0.57	0.57	0.63	0.71	0.61
Bag of Words	0.71	0.79	0.79	0.77	0.80	0.75	0.78	0.77

Bảng 2. Kết quả huấn luyện Accuracy mô hình truyền thống

❖ **Precision**

	KNN	Naives Bayes	Logistic Regression	SVM Linear	SVM Non-Linear	Decision Tree	Random Forest	Trung bình
TF-IDF	0.78	0.82	0.93	0.81	0.86	0.80	0.86	0.84
Word2Vec	0.63	0.48	0.45	0.86	0.00	0.57	0.74	0.53
Bag of Words	0.83	0.77	0.83	0.76	0.85	0.79	0.84	0.81

Bảng 3. Kết quả huấn luyện Precision mô hình truyền thống

❖ **Recall**

	KNN	Naives Bayes	Logistic Regression	SVM Linear	SVM Non-Linear	Decision Tree	Random Forest	Trung bình
TF-IDF	0.64	0.67	0.43	0.68	0.64	0.56	0.59	0.60
Word2Vec	0.44	0.68	0.05	0.01	0.00	0.56	0.48	0.32
Bag of Words	0.42	0.72	0.65	0.70	0.65	0.57	0.60	0.62

Bảng 4. Kết quả huấn luyện Recall mô hình truyền thống

❖ **F1-Score**

	KNN	Naives Bayes	Logistic Regression	SVM Linear	SVM Non-Linear	Decision Tree	Random Forest	Trung bình
TF-IDF	0.70	0.73	0.59	0.74	0.73	0.66	0.70	0.69
Word2Vec	0.52	0.56	0.09	0.01	0.00	0.56	0.59	0.33
Bag of Words	0.55	0.75	0.73	0.73	0.74	0.67	0.70	0.70

Bảng 5. Kết quả huấn luyện F1 mô hình truyền thống

Các mô hình học sâu cũng được tham khảo và sử dụng, tổng cộng có 3 mô hình đã được huấn luyện, trong đó: 1 mô hình sử dụng phương pháp trích xuất đặc trưng Bag of Words, 1 sử dụng phương pháp trích xuất đặc trưng TF-IDF và 1 mô hình còn lại sử dụng tầng Embedding với khả năng trích xuất đặc trưng ngay trong mô hình.

	Accuracy	Precision	Recall	F1
FNN & TF-IDF	0.81	0.81	0.79	0.79
FNN & Bag of Words	0.80	0.80	0.79	0.79
RNN & Embedding	0.78	0.78	0.78	0.78

Bảng 6. Tổng quan kết quả các mô hình học sâu

Mô hình BERT cũng được sử dụng để có thể so sánh và đánh giá so với cá mô hình truyền thống hay mô hình FNN, RNN

	Accuracy	Precision	Recall	F1
BERT	0.81	0.78	0.78	0.78

Bảng 7. Tổng quan kết quả mô hình BERT

Tổng quan thì kết quả của các mô hình truyền thống tương đối giống nhau, (độ chính xác vào khoảng 65-80%). Đặc biệt là mô hình SVM Non-Linear sử dụng phương pháp trích xuất đặc trưng Bag of Words có kết quả nổi bật hơn hết (Accuracy 80.01%). Phương pháp trích xuất đặc trưng Word2Vec cho kết quả trung bình thấp nhất so với 2 phương pháp còn lại.

Các mô hình học sâu cũng cho kết quả gần giống nhau (độ chính xác vào khoảng 78-81%), ở hầu hết 4 giá trị. Trong đó nổi trội nhất là mô hình FNN với phương pháp trích xuất đặc trưng là TF-IDF với tất cả chỉ số đều vượt trội hơn các mô hình còn lại (độ chính xác cao nhất với 81%).

CHƯƠNG 4:
TRIỂN KHAI HỆ THỐNG

4.1. Công nghệ sử dụng

Để có thể kết hợp được việc nhận và truyền tải dữ liệu từ trang web về máy chủ và sử dụng các mô hình đã huấn luyện để dự đoán kết quả và trả ngược lại về trang web người dùng, ta có thể sử dụng nhiều công nghệ khác nhau. Ở đây chúng ta sẽ sử dụng công nghệ FastAPI cho backend đi kèm với React Next.js làm frontend.

- FastAPI được chọn vì có thể dễ dàng tương tác được với các package, mô hình đã huấn luyện ở trên, FastAPI cũng cung cấp cho người dùng khả năng xây dựng hệ thống API một cách nhanh chóng.
- React Next.js cũng là một trong những công cụ xây dựng phần frontend cho ứng dụng web mượt mà, tối ưu hóa, đi kèm với nhiều thư viện components open source do cộng đồng đông đảo người dùng React phát triển trong nhiều năm.
- Docker cũng được tận dụng vì là một công cụ tiện lợi giúp chúng ta đóng gói toàn bộ ứng dụng và triển khai trên server một cách dễ dàng, tránh các trường hợp ứng dụng khi triển khai thực tế lại không hoạt động như môi trường development hoặc bị thiếu đi các packages cần thiết để hoạt động.

4.2. Nền tảng deploy và tiến hành deployment

Nền tảng deploy được chúng ta sử dụng ở đây là Render (https://render.com). Render cho phép ta sử dụng Python Runtime, NodeJS Runtime hoặc Docker runtime để chạy các ứng dụng web chỉ bằng cách kết nối tài khoản Render tới Gitub hoặc cung cấp trực tiếp liên kết tới Dockerfile đã tải lên Dockerhub.

Một khi dự án của chúng ta đã hoàn thành cả backend và frontend, thiết lập tất cả các thứ cần thiết, định nghĩa các packages cần thiết trong Dockerfile và định nghĩa các biến môi trường, ta có thể tiến hành deploy. Ở đây chúng ta sẽ sử dụng phương pháp kết nối tới Github và chọn Repository đã được xây dựng của chúng ta và sử dụng Docker runtime. Sau đó chúng ta chọn đường dẫn tới thư mục chính của dự án của chúng ta, chọn đường dẫn tới Dockerfile để có thể chạy các câu lệnh, cài đặt các packages cần thiết và chạy câu lệnh khởi động hệ thống. Sau đó chúng ta cũng cần phải thiết lập cả các biến môi trường, chẳng hạn như PORT và SECRET_KEY. Sau đó chúng ta tiến hành deploy, Render sẽ tự động tạo thực hiện và tạo domain cũng như chuyển tiếp cổng từ môi trường Docker tới website.

Tuy nhiên, do đây là một nền tảng miễn phí nên không thể triển khai quá nhiều tài nguyên, do đó không thể triển khai các mô hình học sâu vào trang web này.

4.3. Giao diện thực tế

Sau khi deploy, Render sẽ cung cấp cho chúng ta một domain tự tạo, liên kết này sẽ tự tạo dựa trên tên folder của Github reposity hoặc bạn có thể tự tùy chỉnh ngay lúc tạo dự án triển khai trên Render. Ở đây ta có được domain sau: tweetclassification.onrender.com

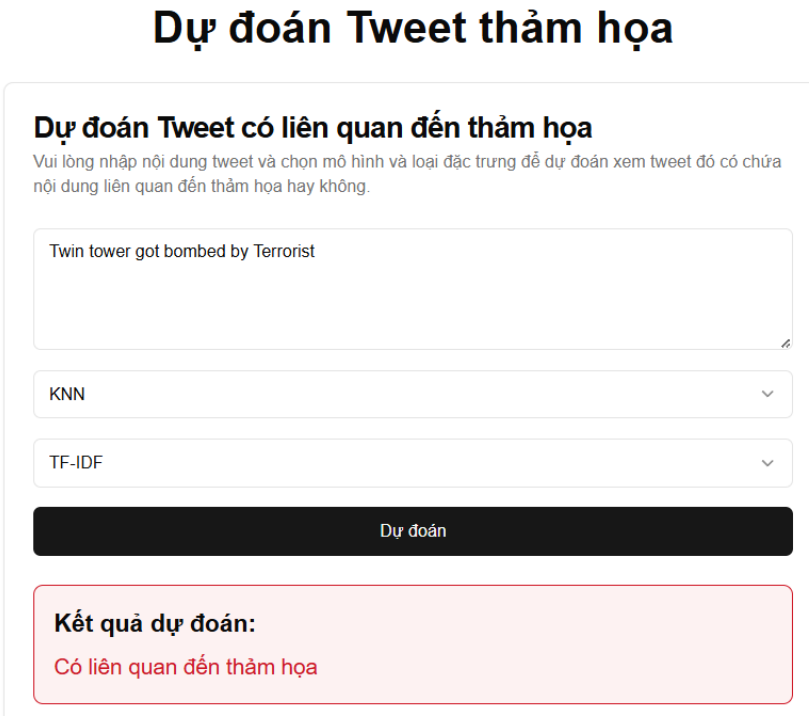
Giao diện đầu tiên người dùng truy cập vào trang web sẽ là như sau:

Hình 25. Giao diện người dùng truy cập Website

Hệ thống sẽ cho phép ta nhập vào nội dung Tweet cần dự đoán, cho phép chúng ta chọn mô hình dự đoán (KNN, Naives Bayes, Decision Tree, Random Forest, Logistic Regression, SVM Linear, SVM Non-Linear) và cuối cùng là chọn cách trích xuất đặc trưng (TF-IDF, Word2Vec, Bag of Words). Để có thể dự đoán, ta cần thực hiện đủ các bước: nhập nội dung tweet, chọn model và chọn cách trích xuất đặc trưng.

Khi nhấn nút dự đoán, hệ thống sẽ trả kết quả về ngay lập tức do sự tối ưu hóa về mặt xây dựng hệ thống cũng như thời gian phản hồi ngắn của mô hình đã được xây dựng. Hệ thống sẽ cho ra một trong 2 kết quả dựa trên nội dung người dùng đã nhập vào hệ thống.

- Khi nội dung có liên quan đến thảm họa, khung thông báo kết quả sẽ xuất hiện màu đỏ nhằm cảnh báo cho người dùng.



Hình 26. Giao diện thông báo kết quả có liên quan đến thảm họa

- Khi nội dung của Tweet không liên quan đến thảm họa, khung thông báo kết quả sẽ có màu xanh nhằm cho người dùng biết rằng đó là thông tin an toàn

Hình 27. Giao diện thông báo kết quả không liên quan đến thảm họa

PHẦN KẾT LUẬN

1. Kết quả đạt được

Đề tài đã được thực hiện thành công và đạt được các kết quả như sau:

- Xử lí được các phần dữ liệu dư thừa và các dữ liệu bị thiếu thông tin.
- Xây dựng và đánh giá tổng quan dưới nhiều mô hình huấn luyện để so sánh.
- Triển khai ứng dụng web cho phép kiểm tra nội dung đoạn Tweet có liên quan đến thảm họa hay không.

2. Hướng phát triển

- Mở rộng phạm vi nghiên cứu để phân loại chi tiết các loại thảm họa
    + Thay vì phân loại đơn giản thành các tweet có hoặc không có nội dung về thảm họa, mô hình có thể được mở rộng để phân loại chi tiết các loại thảm họa, như thiên tai (động đất, bão, lũ lụt), tai nạn công nghiệp, sự cố môi trường, và các tình huống khẩn cấp y tế.
    + Sự mở rộng này sẽ giúp cung cấp thông tin cụ thể hơn, từ đó các cơ quan chức năng có thể phản ứng một cách chính xác và nhanh chóng hơn đối với từng loại thảm họa. Bên cạnh đó, mô hình cũng có thể học cách nhận diện các từ khóa và ngữ cảnh đặc trưng cho từng loại thảm họa, từ đó giảm thiểu độ nhiễu thông tin khi phân tích dữ liệu.
- Phát triển mô hình máy học có khả năng dự đoán thảm họa và đưa ra các biện pháp phòng ngừa
    + Một bước tiến xa hơn so với nhận dạng thông tin thảm họa là xây dựng các mô hình dự đoán dựa trên các chỉ số xã hội và môi trường có thể được phản ánh qua các tweet hoặc thông tin mạng xã hội khác.
    + Mô hình có thể học cách nhận diện những dấu hiệu cảnh báo trước khi thảm họa xảy ra. Ví dụ, một số sự kiện như động đất có thể gây ra các thay đổi môi trường nhỏ trước khi có ảnh hưởng rõ rệt, và những dấu hiệu này có thể được nhận biết qua các mạng xã hội.
    + Các biện pháp phòng ngừa có thể bao gồm việc cảnh báo kịp thời đến các vùng có nguy cơ cao, gợi ý các biện pháp chuẩn bị cho người dân và hỗ trợ điều phối nguồn lực.