BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC ĐẠI NAM

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI TẬP LỚN

NHẬP MÔN HỌC MÁY

PHÂN LOẠI TIN NHẮN RÁC

Sinh viên thực hiện : **Hà Huy Khánh Dương**
Nguyễn Thế Hạnh

Ngành : **Công nghệ thông tin**

Giảng viên hướng dẫn : **ThS. Lê Thị Thùy Trang**

Lời cảm ơn

Chúng em xin gửi lời cảm ơn chân thành đến Ths. Lê Thị Thùy Trang đã tận tình giảng dạy và truyền đạt kiến thức trong suốt quá trình học tập, tạo nền tảng vững chắc để chúng em thực hiện bài tập lớn này

Mục lục

Danh sách bảng

Danh sách hình vẽ

Chương 1

Tổng quan về bài toán và mô hình ứng dụng

1.1 Tổng quan về phân loại văn bản

1.1.1 Khái niệm phân loại văn bản

Phân loại văn bản là một trong những bài toán cơ bản và quan trọng trong lĩnh vực xử lý ngôn ngữ tự nhiên. Nhiệm vụ chính của nó là gán cho mỗi văn bản một nhãn thuộc về một tập các nhãn đã được xác định trước. Nói cách khác, đây là quá trình tự động sắp xếp văn bản vào những nhóm có ý nghĩa dựa trên nội dung mà văn bản thể hiện.

Để thực hiện được điều này, hệ thống phân loại văn bản thường trải qua các bước chính. Trước hết là tiền xử lý dữ liệu, nhằm loại bỏ các yếu tố gây nhiễu và chuẩn hóa nội dung văn bản về dạng thống nhất. Tiếp theo, văn bản sẽ được chuyển đổi thành đặc trưng số để máy tính có thể hiểu và xử lý, thông qua các phương pháp biểu diễn phổ biến như Bag-of-Words, TF-IDF hoặc Embeddings. Trên cơ sở đặc trưng đã được trích xuất, các thuật toán học máy sẽ được áp dụng để xây dựng mô hình, từ đó học được cách nhận biết và phân biệt văn bản thuộc từng loại.

Phân loại văn bản ngày nay được ứng dụng rộng rãi nhờ khả năng xử lý hiệu quả khối lượng dữ liệu khổng lồ trong thời gian ngắn. Thay vì phải thực hiện thủ công, hệ thống phân loại tự động giúp tăng tốc độ, giảm chi phí nhân lực, đồng thời nâng cao độ chính xác và tính nhất quán trong xử lý dữ liệu. Đây cũng chính là lý do phân loại văn bản trở thành một trong những nền tảng quan trọng cho nhiều ứng dụng hiện đại trong lĩnh vực công nghệ thông tin và truyền thông.

1.1.2 Các ứng dụng thực tiễn của phân loại văn bản

Phân loại văn bản hiện nay được ứng dụng rộng rãi trong nhiều lĩnh vực, đặc biệt trong bối cảnh lượng dữ liệu ngôn ngữ trên Internet ngày càng lớn. Một trong những ứng dụng phổ biến nhất là việc phát hiện và loại bỏ tin nhắn rác hay thư điện tử rác, giúp nâng cao trải nghiệm người dùng và bảo vệ an toàn thông tin. Bên cạnh đó, các hệ thống phân loại còn được sử dụng để tổ chức và sắp xếp tài liệu theo chủ đề, từ đó hỗ trợ việc quản lý và tra cứu dữ liệu hiệu quả hơn.

Trong lĩnh vực kinh doanh, phân loại văn bản đóng vai trò quan trọng trong việc phân tích ý kiến khách hàng thông qua phản hồi hoặc bình luận trực tuyến. Kết quả phân tích này giúp doanh nghiệp hiểu rõ hơn nhu cầu và xu hướng của thị trường, từ đó đưa ra chiến lược phát triển phù hợp. Ngoài ra, trên các nền tảng truyền thông, công nghệ phân loại văn bản được khai thác để giám sát nội dung, phát hiện những thông tin sai lệch, tiêu cực hoặc độc hại, góp phần xây dựng môi trường trực tuyến lành mạnh.

Nhìn chung, phân loại văn bản không chỉ hỗ trợ xử lý dữ liệu ngôn ngữ một cách tự động mà còn mang lại giá trị thực tiễn trong nhiều hoạt động đời sống và sản xuất. Chính điều này cho thấy tầm quan trọng và tính ứng dụng cao của kỹ thuật này trong bối cảnh dữ liệu số ngày càng phát triển mạnh mẽ.

1.1.3 Thách thức trong phân loại tin nhắn SMS tiếng việt

Phân loại tin nhắn SMS tiếng Việt là một bài toán không đơn giản do nhiều đặc thù của ngôn ngữ và cách sử dụng trong thực tế. Trước hết, tiếng Việt có cấu trúc ngữ pháp phức tạp, từ ngữ thường được viết liền nhau mà không có dấu tách từ rõ ràng, điều này gây khó khăn cho quá trình tách từ và trích xuất đặc trưng. Bên cạnh đó, người dùng trong các tin nhắn thường sử dụng nhiều cách viết khác nhau như viết tắt, sử dụng teencode, ký tự đặc biệt hoặc xen lẫn tiếng Anh, khiến dữ liệu trở nên không thống nhất và khó chuẩn hóa.

Một thách thức khác là sự mất cân bằng dữ liệu giữa tin nhắn rác và tin nhắn hợp lệ. Trong thực tế, số lượng tin nhắn rác thường ít hơn hoặc nhiều hơn đáng kể so với tin nhắn thông thường, làm cho mô hình dễ bị thiên lệch khi học. Ngoài ra, nội dung của tin nhắn rác cũng thường xuyên thay đổi để né tránh các hệ thống phát hiện, khiến mô hình cần được cập nhật liên tục để duy trì hiệu quả.

Chính những đặc điểm trên khiến cho việc phân loại tin nhắn SMS tiếng Việt đòi hỏi phải có bước tiền xử lý kỹ lưỡng, lựa chọn phương pháp biểu diễn văn bản phù hợp, đồng thời áp dụng các thuật toán phân loại hiệu quả nhằm đạt được độ chính xác cao và khả năng thích ứng tốt.

1.2 Bài toán phân loại tin nhắn rác

1.2.1 Định nghĩa tin nhắn rác và ảnh hưởng trong thực tế

Tin nhắn rác (Spam SMS) là những tin nhắn được gửi đi hàng loạt đến nhiều người dùng mà không có sự cho phép hoặc nhu cầu tiếp nhận từ phía họ. Nội dung của các tin nhắn này thường mang tính quảng cáo, chào mời dịch vụ, hoặc thậm chí chứa liên kết độc hại nhằm lừa đảo và chiếm đoạt thông tin cá nhân. Điểm đặc trưng của tin nhắn rác là tần suất xuất hiện cao, lặp đi lặp lại và không mang lại giá trị thực sự cho người nhận.

Trong thực tế, tin nhắn rác gây ra nhiều tác động tiêu cực. Đối với người dùng cá nhân, chúng làm phiền, tiêu tốn thời gian và tiềm ẩn nguy cơ mất an toàn thông tin khi người nhận vô tình truy cập vào các liên kết nguy hiểm. Với các nhà cung cấp dịch vụ viễn thông, tin nhắn rác tạo gánh nặng cho hệ thống, ảnh hưởng đến chất lượng dịch vụ và uy tín đối với khách hàng. Ở mức độ xã hội, tình trạng phát tán tin nhắn rác tràn lan còn gây ra sự lãng phí tài nguyên mạng, đồng thời làm giảm niềm tin của người dùng đối với các kênh thông tin trực tuyến.

Vì những lý do đó, việc nghiên cứu và triển khai các mô hình phát hiện tin nhắn rác là cần thiết, nhằm đảm bảo an toàn, nâng cao trải nghiệm người dùng và góp phần xây dựng môi trường thông tin lành mạnh.

1.2.2 Đặc điểm dữ liệu SMS rác và SMS hợp lệ

Dữ liệu tin nhắn SMS có thể được chia thành hai nhóm chính: tin nhắn rác và tin nhắn hợp lệ. Mỗi nhóm có những đặc điểm riêng, tạo ra sự khác biệt rõ ràng nhưng cũng tồn tại nhiều điểm giao thoa khiến việc phân loại trở nên phức tạp.

Đối với tin nhắn rác, nội dung thường mang tính quảng cáo, mời chào dịch vụ hoặc chứa các liên kết dẫn tới website bên ngoài. Chúng thường ngắn gọn, lặp lại theo một khuôn mẫu nhất định và có xu hướng sử dụng ngôn ngữ gây chú ý mạnh như khuyến mãi, ưu đãi, hoặc thông báo khẩn cấp. Một đặc điểm khác là tần suất xuất hiện cao và thường được gửi đồng loạt đến nhiều người dùng trong cùng một thời điểm.

Ngược lại, tin nhắn hợp lệ thường mang tính cá nhân hóa cao, phục vụ cho giao tiếp hàng ngày hoặc trao đổi thông tin thực sự cần thiết. Nội dung của loại tin nhắn này đa dạng hơn, ít lặp lại và thường không chứa các từ khóa mang tính thương mại hoặc quảng cáo quá mức. Tin nhắn hợp lệ có thể đến từ người quen, tổ chức, hoặc dịch vụ chính thống mà người dùng đã đăng ký.

Sự khác biệt giữa hai loại tin nhắn là cơ sở để trích xuất đặc trưng và huấn luyện mô hình phân loại. Tuy nhiên, ranh giới giữa chúng không phải lúc nào cũng rõ ràng, bởi nhiều tin nhắn rác được thiết kế tinh vi, bắt chước hình thức của tin nhắn hợp lệ nhằm qua mặt hệ thống phát hiện. Đây chính là một trong những thách thức quan trọng trong việc xây dựng mô hình phát

hiện tin nhắn rác hiệu quả.

1.2.3 Vai trò của học máy trong phát hiện tin nhắn rác

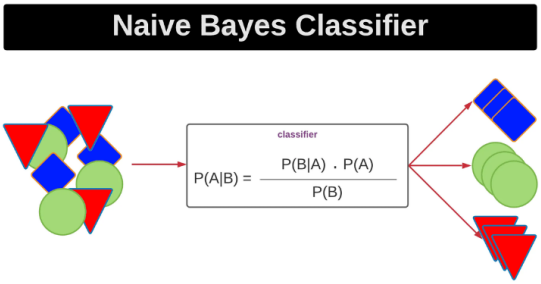
Học máy đóng vai trò then chốt trong việc phát hiện và phân loại tin nhắn rác. Khác với các phương pháp lọc thủ công dựa trên quy tắc tĩnh, học máy cho phép hệ thống tự động học từ dữ liệu và tìm ra những đặc trưng quan trọng giúp phân biệt tin nhắn rác với tin nhắn hợp lệ. Điều này đặc biệt hữu ích khi nội dung của tin nhắn rác thường xuyên thay đổi để né tránh các bộ lọc truyền thống.

Thông qua quá trình huấn luyện trên tập dữ liệu gồm cả tin nhắn rác và tin nhắn hợp lệ, mô hình học máy có khả năng nhận diện các mẫu ngôn ngữ đặc trưng, tần suất xuất hiện từ khóa hoặc cấu trúc câu thường gặp trong tin nhắn rác. Kết quả là hệ thống có thể đưa ra quyết định phân loại nhanh chóng và chính xác mà không cần can thiệp thủ công.

Bên cạnh đó, học máy còn cho phép so sánh nhiều phương pháp biểu diễn dữ liệu và thuật toán khác nhau, từ đó lựa chọn giải pháp tối ưu cho từng bối cảnh cụ thể. Quan trọng hơn, các mô hình học máy có khả năng thích ứng, nghĩa là chúng có thể được cập nhật và cải thiện liên tục khi có thêm dữ liệu mới. Nhờ những ưu điểm này, học máy không chỉ nâng cao hiệu quả trong việc phát hiện tin nhắn rác mà còn góp phần đảm bảo an toàn thông tin và tạo ra trải nghiệm tốt hơn cho người dùng.

1.3 Các thuật toán học máy áp dụng cho bài toán

1.3.1 Naive Bayes



Hình 1.1: Thuật toán Naive Bayes

Naive Bayes là một trong những thuật toán cơ bản và phổ biến nhất trong phân loại văn bản. Thuật toán này dựa trên định lý Bayes trong xác suất thống kê, với giả định rằng các đặc trưng trong dữ liệu là độc lập với nhau. Mặc dù giả định này đơn giản và không hoàn toàn đúng trong

thực tế, nhưng Naive Bayes vẫn thường cho kết quả hiệu quả và đáng tin cậy, đặc biệt trong các bài toán có dữ liệu văn bản lớn.

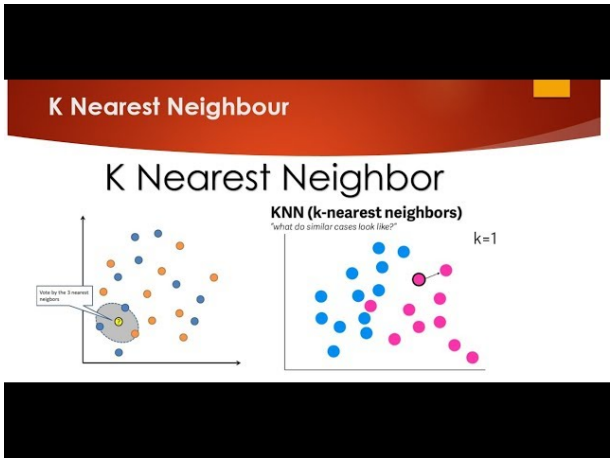
Cơ chế hoạt động của Naive Bayes là tính xác suất một văn bản thuộc về mỗi lớp dựa trên xác suất xuất hiện của các từ trong văn bản. Lớp có xác suất cao nhất sẽ được chọn làm nhãn dự đoán cho văn bản đó. Do cách tiếp cận này, Naive Bayes có ưu điểm là tốc độ huấn luyện nhanh, dễ triển khai và yêu cầu tài nguyên tính toán thấp.

Trong bài toán phát hiện tin nhắn rác, Naive Bayes thường được sử dụng vì khả năng xử lý tốt dữ liệu văn bản ngắn, đồng thời thích hợp với các phương pháp biểu diễn đặc trưng như Bag-of-Words hoặc TF-IDF. Tuy không phải lúc nào cũng đạt độ chính xác cao nhất so với các thuật toán phức tạp hơn, nhưng Naive Bayes vẫn được xem là một lựa chọn hiệu quả nhờ sự cân bằng giữa hiệu suất, độ chính xác và khả năng triển khai thực tế.

1.3.2 K-Nearest Neighbors

K-Nearest Neighbors (KNN) là một thuật toán phân loại dựa trên nguyên tắc "láng giềng gần nhất". Ý tưởng chính của KNN là một văn bản mới sẽ được gán nhãn dựa trên đa số nhãn của k mẫu dữ liệu gần nhất với nó trong không gian đặc trưng. Mức độ “gần” ở đây thường được đo bằng các khoảng cách như khoảng cách Euclid, khoảng cách Cosine hoặc các độ đo tương đồng khác.

Thuật toán KNN có ưu điểm là đơn giản, trực quan và không cần giai đoạn huấn luyện phức tạp. Toàn bộ dữ liệu huấn luyện được lưu giữ và quá trình phân loại chỉ diễn ra khi có dữ liệu mới cần dự đoán. Điều này giúp KNN có khả năng thích ứng với nhiều dạng dữ liệu khác nhau và dễ dàng triển khai trong các hệ thống phân loại văn bản.



Hình 1.2: Thuật toán K-Nearest Neighbors (KNN)

Trong bài toán phát hiện tin nhắn rác, KNN có thể khai thác hiệu quả các đặc trưng được biểu diễn bằng TF-IDF hoặc Embeddings để tính toán độ tương đồng giữa tin nhắn mới và tập

dữ liệu huấn luyện. Tuy nhiên, nhược điểm của KNN là chi phí tính toán cao khi kích thước dữ liệu lớn, vì mỗi lần phân loại đều phải so sánh với toàn bộ dữ liệu đã có. Ngoài ra, thuật toán này cũng dễ bị ảnh hưởng bởi dữ liệu nhiễu và sự mất cân bằng lớp.

Mặc dù vậy, nhờ sự đơn giản và tính trực quan, KNN vẫn được xem là một thuật toán cơ bản đáng được xem xét và so sánh trong quá trình xây dựng mô hình phát hiện tin nhắn rác.

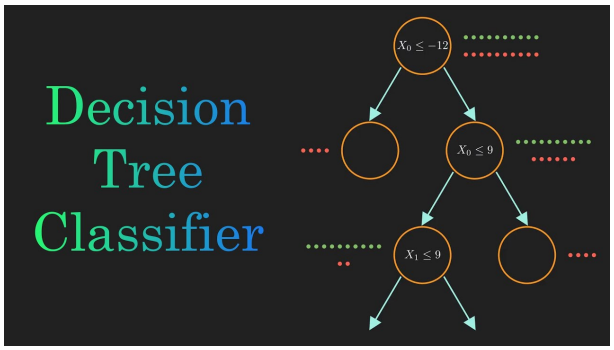
1.3.3 Decision Tree

Decision Tree (Cây quyết định) là một thuật toán phân loại phổ biến dựa trên cấu trúc cây nhị phân. Trong mô hình này, mỗi nút trong cây đại diện cho một điều kiện kiểm tra trên đặc trưng dữ liệu, các nhánh thể hiện kết quả của điều kiện, và các lá cuối cùng tương ứng với nhãn phân loại. Quá trình phân loại diễn ra bằng cách duyệt từ gốc đến lá theo các điều kiện được thiết lập sẵn.

Ưu điểm nổi bật của Decision Tree là tính trực quan và dễ giải thích. Mô hình có thể được biểu diễn bằng sơ đồ cây, giúp người dùng dễ dàng hiểu rõ lý do tại sao một văn bản lại được phân loại vào một nhãn cụ thể. Bên cạnh đó, thuật toán này có khả năng xử lý cả dữ liệu số và dữ liệu dạng phân loại, đồng thời hoạt động tốt với các bộ dữ liệu có số lượng đặc trưng vừa phải.

Trong bài toán phát hiện tin nhắn rác, Decision Tree có thể dựa vào sự xuất hiện hoặc tần suất của các từ khóa để đưa ra quyết định phân loại. Tuy nhiên, hạn chế của mô hình này là dễ bị quá khớp (overfitting) nếu cây được xây dựng quá sâu, dẫn đến khả năng tổng quát hóa kém khi gặp dữ liệu mới. Ngoài ra, khi số lượng đặc trưng trong dữ liệu văn bản quá lớn, hiệu quả của Decision Tree có thể bị suy giảm.

Dù còn tồn tại hạn chế, Decision Tree vẫn là một thuật toán quan trọng, thường được sử dụng như một mô hình cơ bản hoặc kết hợp với các phương pháp khác để cải thiện hiệu suất trong phân loại văn bản nói chung và phát hiện tin nhắn rác nói riêng.



Hình 1.3: Thuật toán Decision Tree

1.3.4 So sánh ưu và nhược điểm của các thuật toán

Ba thuật toán Naive Bayes, K-Nearest Neighbors (KNN) và Decision Tree đều được ứng dụng rộng rãi trong phân loại văn bản, tuy nhiên mỗi thuật toán lại có những đặc điểm riêng với ưu điểm và hạn chế khác nhau.

Naive Bayes có ưu điểm nổi bật là tốc độ huấn luyện nhanh, yêu cầu ít tài nguyên và hoạt động tốt với dữ liệu văn bản ngắn. Thuật toán này đặc biệt hiệu quả khi dữ liệu có quy luật xác suất rõ ràng. Tuy nhiên, giả định độc lập giữa các đặc trưng là chưa hoàn toàn chính xác trong thực tế, nên trong nhiều trường hợp độ chính xác không cao bằng các mô hình phức tạp hơn.

KNN lại có lợi thế ở tính trực quan và khả năng thích ứng với nhiều dạng dữ liệu. Mô hình không cần giai đoạn huấn luyện, mà chỉ dựa vào việc tính toán khoảng cách với dữ liệu đã có. Dù vậy, chi phí tính toán cao khi dữ liệu lớn và sự nhạy cảm với nhiễu là hạn chế đáng kể của KNN.

Trong khi đó, Decision Tree có ưu điểm là dễ hiểu, dễ giải thích và có thể trực quan hóa quá trình phân loại. Mô hình cũng xử lý tốt dữ liệu dạng phân loại. Tuy nhiên, Decision Tree dễ bị quá khớp nếu không có cơ chế cắt tỉa hợp lý, và hiệu quả có thể giảm khi số lượng đặc trưng quá lớn.

Tóm lại, không có thuật toán nào là hoàn hảo trong mọi trường hợp. Việc lựa chọn thuật toán phù hợp cần dựa trên đặc điểm dữ liệu, mục tiêu của bài toán cũng như sự cân bằng giữa độ chính xác, tốc độ xử lý và khả năng triển khai trong thực tế.

Chương 2

Triển khai ứng dụng và kết quả thực nghiệm

2.1 Thiết kế hệ thống

2.1.1 Kiến trúc tổng quát

Hệ thống phân loại tin nhắn rác trong nghiên cứu này được triển khai theo một kiến trúc tổng quát của quy trình học máy. Quy trình này bắt đầu từ giai đoạn chuẩn bị dữ liệu, sau đó lần lượt đi qua các bước tiền xử lý, trích xuất đặc trưng, huấn luyện mô hình và cuối cùng là dự đoán nhãn cho các tin nhắn mới. Nhờ có sự tổ chức theo quy trình khép kín, hệ thống đảm bảo được tính nhất quán từ đầu vào cho đến khi tạo ra kết quả phân loại.

Ở giai đoạn đầu tiên, dữ liệu thô được thu thập từ các nguồn khác nhau và thường chứa nhiều thành phần không cần thiết như ký tự đặc biệt, biểu tượng cảm xúc, chữ viết tắt hoặc các cách viết không theo chuẩn. Tiền xử lý dữ liệu giúp chuẩn hóa lại toàn bộ văn bản, loại bỏ những yếu tố gây nhiễu và chuyển chúng về dạng thống nhất, tạo nền tảng vững chắc cho việc phân tích ở các bước tiếp theo.

Sau khi dữ liệu được làm sạch, văn bản sẽ được biến đổi thành dạng số bằng những phương pháp biểu diễn đặc trưng. Các kỹ thuật thường được áp dụng là Bag of Words, TF-IDF hoặc Embeddings. Những phương pháp này giúp chuyển văn bản thành các vector số, để máy tính có thể xử lý và sử dụng làm đầu vào cho các thuật toán học máy. Mỗi cách tiếp cận đều có ưu điểm và hạn chế riêng, nhưng đều hướng đến việc giữ lại nhiều nhất có thể ý nghĩa của nội dung gốc trong khi vẫn đảm bảo tính toán hiệu quả.

Các vector đặc trưng sau đó được đưa vào mô hình học máy để tiến hành huấn luyện. Những mô hình phổ biến trong phân loại văn bản như Naive Bayes, Logistic Regression hay SVM sẽ

học từ dữ liệu huấn luyện và xây dựng quy tắc nhằm phân biệt giữa tin nhắn rác và tin nhắn hợp lệ. Khi hệ thống đã được huấn luyện đầy đủ, nó có thể áp dụng cho dữ liệu mới, giúp tự động phát hiện và gắn nhãn cho những tin nhắn SMS chưa từng xuất hiện trước đó.

Để triển khai toàn bộ quy trình trên trong môi trường Python, việc cài đặt và import các thư viện cần thiết là bước chuẩn bị quan trọng. Các thư viện như pandas và numpy hỗ trợ thao tác dữ liệu, scikit-learn cung cấp công cụ vector hóa, huấn luyện và đánh giá mô hình, underthesea phục vụ cho xử lý ngôn ngữ tự nhiên tiếng Việt, trong khi matplotlib và seaborn được dùng để trực quan hóa dữ liệu cũng như kết quả thử nghiệm. Những thư viện này đảm bảo toàn bộ quá trình từ tiền xử lý cho đến đánh giá mô hình có thể được triển khai một cách nhất quán, dễ dàng tái sử dụng và mở rộng.

Hình dưới đây minh họa đoạn mã Python dùng để cài đặt và import các thư viện phục vụ cho việc tiền xử lý, vector hóa và huấn luyện mô hình. Đây là nền tảng kỹ thuật quan trọng, giúp kết nối giữa thiết kế kiến trúc lý thuyết với quá trình hiện thực hóa trong môi trường thực nghiệm.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
!pip install underthesea
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix, classification_report
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.svm import LinearSVC
from underthesea import word_tokenize
import re
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
```

Hình 2.1: Cài đặt các thư viện phục vụ tiền xử lý văn bản

```
# ==========================
# 2. LOAD DỮ LIỆU
# ==========================
from google.colab import files
uploaded = files.upload()   # Upload file train.csv
df = pd.read_csv("train.csv")

print("Kích thước dữ liệu:", df.shape)
df.head()
```

Chọn tệp | Không có tệp nào được chọn Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

Kích thước dữ liệu: (5574, 2)

	sms	label
0	Go until jurong point, crazy.. Available only ...	0
1	Ok lar... Joking wif u oni...\n	0
2	Free entry in 2 a wkly comp to win FA Cup fina...	1
3	U dun say so early hor... U c already then say...	0
4	Nah I don't think he goes to usf, he lives aro...	0

Hình 2.2: Code load dữ liệu

2.1.2 Khám phá dữ liệu

Bộ dữ liệu sử dụng trong nghiên cứu gồm 5572 tin nhắn SMS, trong đó có 4827 tin nhắn hợp lệ và 745 tin nhắn rác. Như vậy, dữ liệu có sự mất cân bằng lớp khi tỉ lệ tin nhắn hợp lệ chiếm tới 86.6% và tin nhắn rác chỉ chiếm 13.4%. Việc nhận diện sự chênh lệch này ngay từ đầu là cần thiết để lựa chọn mô hình và phương pháp đánh giá phù hợp.

Trước hết, dữ liệu được kiểm tra về cấu trúc và chất lượng. Kết quả cho thấy tập dữ liệu chỉ có hai cột chính, trong đó một cột chứa nội dung văn bản tin nhắn và cột còn lại là nhãn phân loại. Không có giá trị bị thiếu, vì vậy có thể sử dụng toàn bộ dữ liệu trong quá trình phân tích và huấn luyện mô hình

Tiếp theo, một số thống kê cơ bản được tính toán nhằm làm rõ đặc trưng của dữ liệu. Tổng số tin nhắn trong tập là 5572, độ dài trung bình của một tin nhắn là khoảng 81 ký tự, tương ứng với gần 16 từ. Ngoài ra, độ lệch chuẩn cho thấy độ dài tin nhắn có sự dao động đáng kể giữa các trường hợp. Trực quan hóa bằng biểu đồ phân phối cho thấy phần lớn tin nhắn có độ dài ngắn, tập trung dưới 200 ký tự hoặc dưới 30 từ, tuy nhiên vẫn tồn tại một số ít tin nhắn có độ dài vượt trội

Một đặc điểm đáng chú ý là tin nhắn rác thường dài hơn so với tin nhắn thông thường. Điều này có thể giải thích bởi nội dung quảng cáo hay các thông tin khuyến mãi trong spam thường chứa nhiều chi tiết nhằm thu hút người đọc. Do đó, độ dài tin nhắn có thể coi là một đặc trưng hữu ích trong việc phân biệt hai loại dữ liệu.

Cuối cùng, phân phối nhãn được trực quan hóa cho thấy sự mất cân bằng rõ rệt giữa hai lớp, khi số lượng tin nhắn hợp lệ áp đảo tin nhắn rác Điều này gợi ý rằng khi đánh giá mô hình, chỉ số Accuracy không đủ phản ánh hiệu quả, mà cần sử dụng thêm Precision, Recall và F1-score để có cái nhìn toàn diện hơn về khả năng phát hiện tin nhắn rác.

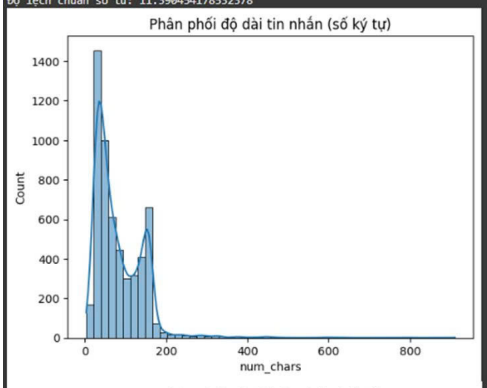
```
# Xem kích thước dữ liệu
print("Số dòng, số cột:", df.shape)
# Xem 5 dòng đầu tiên
print(df.head())
# Thông tin chi tiết các cột
print(df.info())
# Kiểm tra giá trị null
print(df.isnull().sum())
```

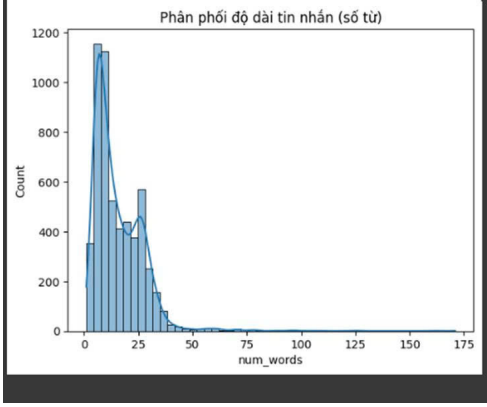
```
Số dòng, số cột: (5574, 2)
                                                 sms  label
0  Go until jurong point, crazy.. Available only ...      0
1                      Ok lar... Joking wif u oni...\n      0
2  Free entry in 2 a wkly comp to win FA Cup fina...      1
3  U dun say so early hor... U c already then say...      0
4  Nah I don't think he goes to usf, he lives aro...      0
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5574 entries, 0 to 5573
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   sms     5574 non-null   object
 1   label   5574 non-null   int64
dtypes: int64(1), object(1)
memory usage: 87.2+ KB
None
sms      0
label    0
dtype: int64
```

Hình 2.3: Kiểm tra cấu trúc và chất lượng dữ liệu SMS

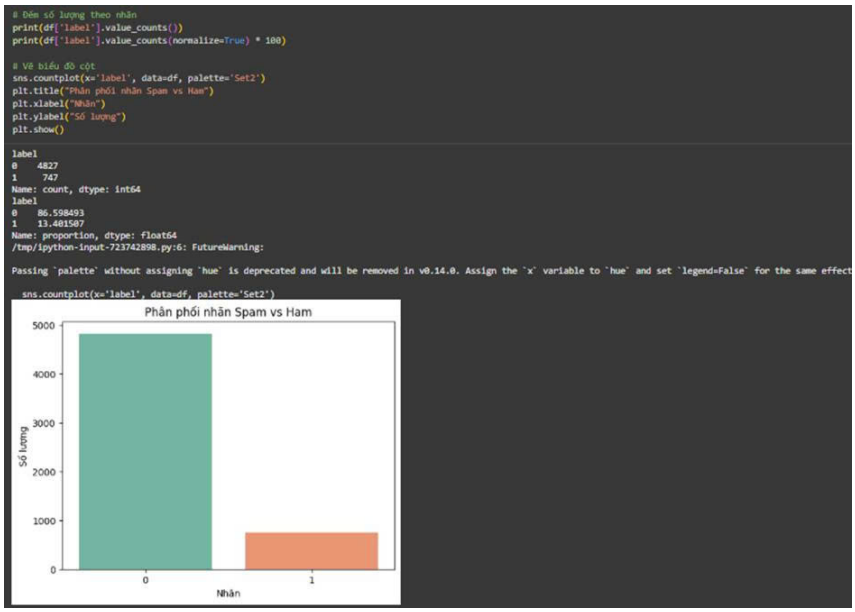
```python
# a. Tổng số mẫu
print("Tổng số tin nhắn:", len(df))

# b. Phân phối nhãn
print(df['label'].value_counts())
print(df['label'].value_counts(normalize=True) * 100)

# c. Độ dài tin nhắn
df['num_chars'] = df['sms'].apply(len)
df['num_words'] = df['sms'].apply(lambda x: len(x.split()))

print("Độ dài trung bình (ký tự):", df['num_chars'].mean())
print("Độ dài trung bình (từ):", df['num_words'].mean())

print("Độ lệch chuẩn số ký tự:", df['num_chars'].std())
print("Độ lệch chuẩn số từ:", df['num_words'].std())

# d. Trực quan độ dài
import seaborn as sns
import matplotlib.pyplot as plt

sns.histplot(df['num_chars'], bins=50, kde=True)
plt.title("Phân phối độ dài tin nhắn (số ký tự)")
plt.show()

sns.histplot(df['num_words'], bins=50, kde=True)
plt.title("Phân phối độ dài tin nhắn (số từ)")
plt.show()
```

```
Tổng số tin nhắn: 5574
label
0    4827
1     747
Name: count, dtype: int64
label
0    86.598493
1    13.401507
Name: proportion, dtype: float64
Độ dài trung bình (ký tự): 81.47829207032652
Độ dài trung bình (từ): 15.591675636885554
Độ lệch chuẩn số ký tự: 59.84830227038691
Độ lệch chuẩn số từ: 11.390454178532378
```





Hình 2.4: Một số thống kê cơ bản và phân phối độ dài tin nhắn trong tập dữ liệu

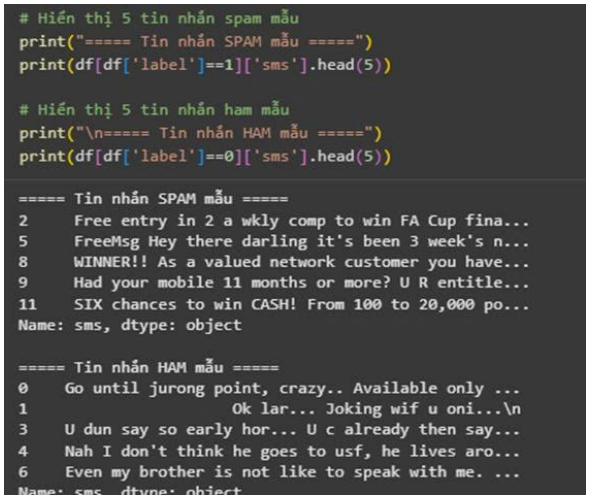


Hình 2.5: Phân phối nhãn Spam và Ham trong tập dữ liệu

2.2 Mã nguồn thực hiện

2.2.1 Tiền xử lý văn bản

Trước khi tiến hành huấn luyện mô hình, cần quan sát một số mẫu dữ liệu để nắm được đặc trưng cơ bản của từng lớp. Khi xem xét năm tin nhắn thuộc lớp spam và năm tin nhắn thuộc lớp ham, có thể nhận thấy sự khác biệt rõ ràng. Những tin nhắn ham chủ yếu mang tính chất giao tiếp cá nhân, câu từ ngắn gọn và cách diễn đạt tự nhiên. Trong khi đó, tin nhắn spam thường chứa nhiều cụm từ mang tính quảng cáo hoặc mời gọi, chẳng hạn như trúng thưởng, khuyến mãi, giảm giá, gọi ngay, đi kèm với số điện thoại hoặc đường dẫn. Đặc điểm này gợi ý rằng từ vựng sẽ là yếu tố quan trọng để phân biệt hai lớp dữ liệu.



```
# Hiển thị 5 tin nhắn spam mẫu
print("===== Tin nhắn SPAM mẫu =====")
print(df[df['label']==1]['sms'].head(5))

# Hiển thị 5 tin nhắn ham mẫu
print("\n===== Tin nhắn HAM mẫu =====")
print(df[df['label']==0]['sms'].head(5))
```

```
===== Tin nhắn SPAM mẫu =====
2     Free entry in 2 a wkly comp to win FA Cup fina...
5     FreeMsg Hey there darling it's been 3 week's n...
8     WINNER!! As a valued network customer you have...
9     Had your mobile 11 months or more? U R entitle...
11    SIX chances to win CASH! From 100 to 20,000 po...
Name: sms, dtype: object

===== Tin nhắn HAM mẫu =====
0    Go until jurong point, crazy.. Available only ...
1                        Ok lar... Joking wif u oni...\n
3    U dun say so early hor... U c already then say...
4    Nah I don't think he goes to usf, he lives aro...
6    Even my brother is not like to speak with me. ...
Name: sms, dtype: object
```

Hình 2.6: Một số tin nhắn spam và ham trong tập dữ liệu

Dữ liệu văn bản sau khi thu thập thường tồn tại nhiều yếu tố gây nhiễu, bao gồm ký tự đặc biệt, chữ hoa, dấu câu, chữ số hoặc các khoảng trắng thừa. Nếu để nguyên, các yếu tố này có thể làm sai lệch quá trình trích xuất đặc trưng, khiến cho mô hình học máy khó phát hiện được các quy luật quan trọng. Vì vậy, tiền xử lý văn bản trở thành một bước bắt buộc nhằm chuẩn hóa dữ liệu đầu vào và nâng cao hiệu quả phân loại.

Quy trình tiền xử lý được tiến hành theo nhiều bước. Trước hết, toàn bộ nội dung tin nhắn được chuyển về chữ thường để loại bỏ sự khác biệt giữa các dạng viết hoa và viết thường. Tiếp đó, hệ thống loại bỏ toàn bộ ký tự đặc biệt, dấu câu, chữ số và chuẩn hóa lại khoảng trắng để văn bản trở nên gọn gàng và nhất quán. Đối với tiếng Việt, một đặc thù quan trọng là dấu cách chỉ phân tách âm tiết chứ không tách từ, do đó cần có công cụ tách từ chuyên dụng. Thư viện underthesea được sử dụng để thực hiện bước này, giúp ghép đúng các cụm từ đa âm tiết chẳng hạn như học sinh giỏi thành học_sinh và giỏi, từ đó cải thiện đáng kể chất lượng dữ liệu đặc trưng.

Bên cạnh đó, do đặc thù của tin nhắn SMS thường xuất hiện nhiều từ viết tắt, teencode hoặc tiếng lóng, một bộ từ điển thay thế đã được xây dựng nhằm chuyển đổi chúng về dạng chuẩn. Ví dụ, các từ hok, dc hay ntn sẽ lần lượt được thay thế bằng không, được và như thế nào. Ngoài ra, các từ dừng ít mang giá trị ngữ nghĩa như và, là, thì cũng được loại bỏ để giảm nhiễu. Kết quả cuối cùng của giai đoạn tiền xử lý là một tập dữ liệu văn bản sạch, thống nhất và giàu thông tin hơn, tạo điều kiện cho các phương pháp biểu diễn đặc trưng như Bag of Words, TF-IDF hay Embeddings phát huy tối đa hiệu quả, từ đó góp phần nâng cao độ chính xác của mô hình phân loại tin nhắn rác trong các bước huấn luyện tiếp theo.

2.2.2 Các bước tiền xử lý dữ liệu SMS

Tiền xử lý dữ liệu là một khâu quan trọng trong quy trình phân loại tin nhắn rác, nhằm chuẩn hóa văn bản và loại bỏ các yếu tố gây nhiễu trước khi đưa vào mô hình học máy. Đối với dữ liệu SMS, ngoài các thao tác cơ bản, nghiên cứu này còn bổ sung nhiều bước chuyên biệt cho tiếng Việt nhằm nâng cao chất lượng dữ liệu đầu vào.

Trước hết, toàn bộ nội dung tin nhắn được chuyển về dạng chữ thường để tránh sự khác biệt không cần thiết giữa chữ hoa và chữ thường. Sau đó, các ký tự đặc biệt, dấu câu, chữ số và khoảng trắng dư thừa được loại bỏ, giúp văn bản trở nên gọn gàng hơn. Một vấn đề thường gặp trong SMS là sự xuất hiện dày đặc của các dạng viết tắt, teencode hoặc từ lóng, ví dụ như “hok” thay cho “không”, “dc” thay cho “được”, hoặc “ntn” thay cho “như thế nào”. Để xử lý, một bộ từ điển ánh xạ đã được xây dựng để chuyển đổi về dạng chuẩn, đảm bảo ý nghĩa của văn bản không bị sai lệch.

Bên cạnh đó, các từ dừng ít mang giá trị ngữ nghĩa như “và”, “là”, “của”, “thì” cũng được loại bỏ nhằm giảm nhiễu và tập trung vào những từ khóa quan trọng. Đặc thù của tiếng Việt

khiến cho việc tách từ trở thành bước không thể thiếu, bởi dấu cách trong tiếng Việt chỉ phân tách âm tiết chứ không phải từ. Vì vậy, thư viện underthesea được sử dụng để ghép đúng các từ ghép nhiều âm tiết, ví dụ như "học sinh giỏi" sẽ được tách thành "học_sinh" và "giỏi", từ đó tăng độ chính xác khi trích xuất đặc trưng. Ngoài ra, trong một số trường hợp, việc chuẩn hóa dấu tiếng Việt cũng được xem xét nhằm khắc phục lỗi gõ sai hoặc thiếu dấu trong dữ liệu.

Kết quả sau quá trình tiền xử lý là một tập dữ liệu sạch, thống nhất và phản ánh chính xác hơn nội dung tin nhắn. Điều này giúp cho các phương pháp biểu diễn văn bản như Bag of Words, TF-IDF hoặc Embeddings hoạt động hiệu quả hơn và tạo nền tảng vững chắc cho quá trình huấn luyện mô hình phân loại sau này.

```
# Ví dụ tách từ
text = "Toi hok di hoc dau nhe"
print("Trước:", text)
print("Sau:", word_tokenize(text, format="text"))

# Hàm xử lý nâng cao
slang_dict = {
    "hok": "không",
    "k": "không",
    "ko": "không",
    "dc": "được",
    "mk": "mình",
    "ntn": "như thế nào"
}

def normalize_slang(text):
    words = text.split()
    return " ".join([slang_dict.get(w, w) for w in words])

df['clean_text_vn'] = df['clean_text'].apply(lambda x: word_tokenize(normalize_slang(x), format="text"))
df[['sms','clean_text','clean_text_vn']].head()
```

```
Trước: Toi hok di hoc dau nhe
Sau: Toi hok di_hoc dau nhe
```

	sms	clean_text	clean_text_vn
0	Go until jurong point, crazy.. Available only ...	go until jurong point crazy available only in ...	go until jurong point crazy available only in ...
1	Ok lar... Joking wif u oni...\n	ok lar joking wif u oni	ok lar joking wif_u oni
2	Free entry in 2 a wkly comp to win FA Cup fina...	free entry in a wkly comp to win fa cup final ...	free entry in a wkly comp to win fa cup final ...
3	U dun say so early hor... U c already then say...	u dun say so early hor u c already then say	u dun say so early hor_u c_already then say
4	Nah I don't think he goes to usf, he lives aro...	nah i dont think he goes to usf he lives aroun...	nah i dont think he goes to usf he lives aroun...

Hình 2.7: Ví dụ minh họa kết quả sau một số bước tiền xử lý dữ liệu SMS

2.2.3 Vector hóa văn bản

Đầu tiên, tập dữ liệu tin nhắn SMS được phân tích để nắm bắt các đặc trưng cơ bản. Bộ dữ liệu gồm 5.572 tin nhắn, trong đó có 4.827 tin nhắn "ham" (86.6%) và 745 tin nhắn "spam" (13.4%). Kết quả thống kê cho thấy dữ liệu có sự mất cân bằng lớp: tin nhắn hợp lệ chiếm đa số, trong khi tin nhắn rác chỉ chiếm một phần nhỏ. Ngoài ra, độ dài trung bình của tin nhắn là khoảng 80 ký tự ($\sim$15 từ). Tin nhắn spam thường dài hơn tin nhắn bình thường và chứa nhiều từ khóa đặc trưng như *khuyến mãi*, *trúng thưởng*, *gọi ngay*, ... Điều này cho thấy độ dài và từ khóa có thể đóng vai trò quan trọng trong việc phân loại.

Sau bước phân tích thống kê, dữ liệu được chuyển đổi sang dạng vector số để phục vụ cho quá trình huấn luyện mô hình. Trong nghiên cứu này, ba phương pháp vector hóa được thử nghiệm:

Bag-of-Words (BoW): mỗi văn bản được biểu diễn bằng vector tần suất xuất hiện của các từ trong tập từ vựng. Ưu điểm là đơn giản, dễ triển khai. Nhược điểm là không nắm bắt được ngữ nghĩa và ngữ cảnh, vector thường thưa (sparse) và kích thước rất lớn. Thư viện sử dụng: CountVectorizer trong scikit-learn. Kết quả: ma trận BoW có kích thước $(5574, 8608)$, nghĩa là 5.574 văn bản được biểu diễn bởi 8.608 đặc trưng.

```
from sklearn.feature_extraction.text import CountVectorizer

bow_vectorizer = CountVectorizer()
X_bow = bow_vectorizer.fit_transform(df['clean_text'])
print("Kích thước ma trận BoW:", X_bow.shape)


Kích thước ma trận BoW: (5574, 8608)
```

Hình 2.8: Kết quả vector hóa bằng phương pháp Bag-of-Words.

TF-IDF (Term Frequency – Inverse Document Frequency): tương tự BoW nhưng điều chỉnh trọng số, giúp giảm ảnh hưởng của các từ xuất hiện thường xuyên nhưng không mang nhiều ý nghĩa phân biệt. Ưu điểm: phù hợp hơn với văn bản ngắn (như SMS). Nhược điểm: vẫn chưa thể hiện được ngữ nghĩa của toàn bộ câu. Thư viện sử dụng: TfidfVectorizer trong scikit-learn. Kết quả: mỗi văn bản được biểu diễn thành vector có cùng kích thước từ vựng (8.608), nhưng các giá trị phản ánh mức độ quan trọng thay vì chỉ là tần suất.

```
from sklearn.feature_extraction.text import CountVectorizer

bow_vectorizer = CountVectorizer()
X_bow = bow_vectorizer.fit_transform(df['clean_text'])
print("Kích thước ma trận BoW:", X_bow.shape)


Kích thước ma trận BoW: (5574, 8608)
```

Hình 2.9: Kết quả vector hóa bằng phương pháp TF-IDF (Term Frequency – Inverse Document Frequency)

Sentence Embeddings (mô hình ngôn ngữ huấn luyện trước): thay vì chỉ đếm từ, mô hình ngôn ngữ pre-trained được sử dụng để biểu diễn toàn bộ câu thành một vector ngữ nghĩa. Ưu điểm: bắt được ngữ cảnh và quan hệ ngữ nghĩa giữa các từ, mang lại hiệu quả cao hơn nhiều so với BoW và TF-IDF. Mô hình sử dụng: paraphrase-multilingual-MiniLM-L12-v2 từ thư viện sentence-transformers, hỗ trợ đa ngôn ngữ (bao gồm cả tiếng Việt). Kết quả: mỗi tin nhắn được biểu diễn thành vector có chiều dài 384, tạo ra ma trận kích thước $(5574, 384)$.

```
!pip install -q sentence-transformers

from sentence_transformers import SentenceTransformer

# Dùng model đa ngôn ngữ (có hỗ trợ tiếng Việt)
embedder = SentenceTransformer('sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2')

# Biểu diễn thành vector
X_embed = embedder.encode(df['clean_text'].tolist(), convert_to_tensor=True)
print("Kích thước Sentence Embeddings:", X_embed.shape)


Kích thước Sentence Embeddings: torch.Size([5574, 384])
```

Hình 2.10: Kết quả vector hóa bằng phương pháp Sentence Embeddings (mô hình ngôn ngữ pre-trained)

Như vậy, cả ba phương pháp vector hóa đều có ưu và nhược điểm riêng. BoW và TF-IDF đơn giản, dễ triển khai nhưng hạn chế về ngữ nghĩa, trong khi Sentence Embeddings khai thác được ngữ cảnh và mang lại biểu diễn giàu thông tin hơn, đặc biệt hữu ích cho bài toán phân loại tin nhắn rác.

2.2.4 Huấn luyện mô hình

Sau khi dữ liệu đã trải qua quá trình tiền xử lý và được chuyển đổi thành các vector đặc trưng bằng các phương pháp như Bag-of-Words, TF-IDF hoặc Sentence Embeddings, bước tiếp theo là huấn luyện các mô hình học máy. Trong nghiên cứu này, ba thuật toán cơ bản nhưng phổ biến được lựa chọn là Naive Bayes, Logistic Regression và Support Vector Machine (SVM). Việc lựa chọn này nhằm đảm bảo sự đa dạng trong cách tiếp cận, đồng thời cho phép so sánh ưu điểm và hạn chế của từng mô hình trên cùng một bộ dữ liệu.

Naive Bayes là một trong những thuật toán thường được áp dụng trong phân loại văn bản nhờ tính đơn giản và hiệu quả. Thuật toán dựa trên định lý Bayes, giả định rằng các đặc trưng xuất hiện trong văn bản là độc lập có điều kiện với nhau. Mặc dù giả định này không phản ánh hoàn toàn chính xác bản chất ngôn ngữ tự nhiên, nhưng trong thực tế, Naive Bayes vẫn cho kết quả tốt nhờ khả năng xử lý nhanh và ít yêu cầu tài nguyên. Mô hình phù hợp khi làm việc với dữ liệu có kích thước lớn và số chiều cao, như trong trường hợp biểu diễn văn bản bằng BoW hoặc TF-IDF.

Logistic Regression là một mô hình tuyến tính phổ biến, thường được dùng trong các bài toán phân loại nhị phân. Ưu điểm của Logistic Regression là dễ huấn luyện, khả năng tổng quát hóa tốt và hiệu quả với dữ liệu có số chiều cao. Ngoài ra, Logistic Regression còn có thể mở rộng thành các biến thể mạnh mẽ hơn, giúp xử lý những trường hợp phức tạp khi dữ liệu khó phân tách tuyến tính.

Support Vector Machine (SVM) lại tiếp cận bài toán theo hướng khác, tìm ra siêu phẳng tối

ưu để phân tách dữ liệu giữa các lớp. Mô hình có khả năng đạt độ chính xác cao, đặc biệt hiệu quả trong những tập dữ liệu có chiều cao và kích thước vừa phải. Tuy nhiên, SVM có thể tốn thời gian huấn luyện nếu dữ liệu quá lớn.

Trong quá trình thực nghiệm, cả ba mô hình được huấn luyện trên cùng một tập dữ liệu đã qua tiền xử lý, với các bộ đặc trưng được xây dựng từ BoW, TF-IDF và Sentence Embeddings. Kết quả đánh giá được so sánh dựa trên các chỉ số Accuracy, Precision, Recall và F1-score. Trong đó, Precision và Recall cho lớp "spam" được quan tâm nhiều hơn vì mục tiêu chính của bài toán là phát hiện đúng tin nhắn rác.

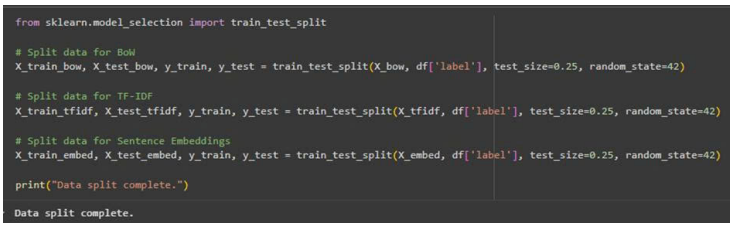
Kết quả cho thấy Naive Bayes có tốc độ nhanh và hiệu quả tốt trên TF-IDF, trong khi Logistic Regression và SVM thường đạt F1-score cao hơn. Sentence Embeddings mang lại kết quả tốt nhất nhờ nắm bắt được ngữ nghĩa của toàn bộ câu, từ đó cải thiện độ chính xác trong phân loại.

```python
def evaluate_model(model, X_train, X_test, y_train, y_test):
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    print(f"===== Evaluation for {model.__class__.__name__} =====")
    print("Accuracy:", accuracy_score(y_test, y_pred))
    print("Classification Report:\n", classification_report(y_test, y_pred))
    print("-" * 30)
    return {
        "Model": model.__class__.__name__,
        "Accuracy": accuracy_score(y_test, y_pred),
        "Precision": precision_score(y_test, y_pred, average='weighted'), # Use weighted average for multi-class
        "Recall": recall_score(y_test, y_pred, average='weighted'),     # Use weighted average for multi-class
        "F1-score": f1_score(y_test, y_pred, average='weighted')         # Use weighted average for multi-class
    }

results = []

# BoW
results.append(evaluate_model(MultinomialNB(), X_train_bow, X_test_bow, y_train, y_test))
results.append(evaluate_model(LogisticRegression(max_iter=1000), X_train_bow, X_test_bow, y_train, y_test))
results.append(evaluate_model(LinearSVC(), X_train_bow, X_test_bow, y_train, y_test))

# TF-IDF
results.append(evaluate_model(MultinomialNB(), X_train_tfidf, X_test_tfidf, y_train, y_test))
results.append(evaluate_model(LogisticRegression(max_iter=1000), X_train_tfidf, X_test_tfidf, y_train, y_test))
results.append(evaluate_model(LinearSVC(), X_train_tfidf, X_test_tfidf, y_train, y_test))

# Nếu có Sentence Embedding thì dùng Logistic Regression
results.append(evaluate_model(LogisticRegression(max_iter=1000), X_train_embed, X_test_embed, y_train, y_test))

# Chuyển kết quả thành bảng
import pandas as pd
results_df = pd.DataFrame(results)
print(results_df)
```

Hình 2.11: Chia dữ liệu huấn luyện và kiểm thử cho BoW, TF-IDF và Sentence Embeddings.

```
===== Evaluation for MultinomialNB =====
Accuracy: 0.9741750358680057
Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.98      0.98      1203
           1       0.89      0.93      0.91       191

    accuracy                           0.97      1394
   macro avg       0.94      0.95      0.95      1394
weighted avg       0.97      0.97      0.97      1394

------------------------------
===== Evaluation for LogisticRegression =====
Accuracy: 0.9813486370157819
Classification Report:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      1203
           1       0.99      0.87      0.93       191

    accuracy                           0.98      1394
   macro avg       0.98      0.94      0.96      1394
weighted avg       0.98      0.98      0.98      1394

------------------------------
===== Evaluation for LinearSVC =====
Accuracy: 0.9806312769010043
Classification Report:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      1203
           1       0.98      0.87      0.93       191

    accuracy                           0.98      1394
   macro avg       0.98      0.94      0.96      1394
weighted avg       0.98      0.98      0.98      1394

------------------------------
===== Evaluation for MultinomialNB =====
Accuracy: 0.9562410329985653
Classification Report:
              precision    recall  f1-score   support

           0       0.95      1.00      0.98      1203
           1       1.00      0.68      0.81       191

    accuracy                           0.96      1394
   macro avg       0.98      0.84      0.89      1394
weighted avg       0.96      0.96      0.95      1394

------------------------------
===== Evaluation for LogisticRegression =====
Accuracy: 0.9641319942611191
Classification Report:
              precision    recall  f1-score   support

           0       0.96      1.00      0.98      1203
           1       0.99      0.74      0.85       191

    accuracy                           0.96      1394
   macro avg       0.98      0.87      0.91      1394
weighted avg       0.97      0.96      0.96      1394

------------------------------
===== Evaluation for LinearSVC =====
Accuracy: 0.9791965566714491
Classification Report:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      1203
           1       0.98      0.87      0.92       191

    accuracy                           0.98      1394
   macro avg       0.98      0.93      0.95      1394
weighted avg       0.98      0.98      0.98      1394

------------------------------
===== Evaluation for LogisticRegression =====
Accuracy: 0.9827833572453372
Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      1203
           1       0.96      0.92      0.94       191

    accuracy                           0.98      1394
   macro avg       0.97      0.95      0.96      1394
weighted avg       0.98      0.98      0.98      1394

------------------------------
                Model  Accuracy  Precision    Recall  F1-score
0      MultinomialNB  0.974175   0.974742  0.974175  0.974397
1  LogisticRegression  0.981349   0.981471  0.981349  0.980863
2           LinearSVC  0.980631   0.980661  0.980631  0.980152
3      MultinomialNB  0.956241   0.958353  0.956241  0.952624
4  LogisticRegression  0.964132   0.965240  0.964132  0.961905
5           LinearSVC  0.979197   0.979150  0.979197  0.978681
6  LogisticRegression  0.982783   0.982608  0.982783  0.982628
```

Hình 2.12: Huấn luyện và đánh giá mô hình với BoW, TF-IDF và Sentence Embeddings.



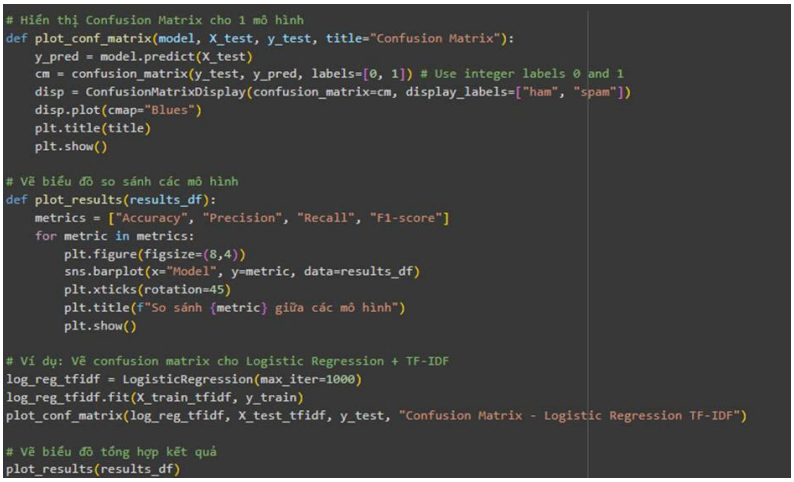
```
from sklearn.model_selection import train_test_split

# Split data for BoW
X_train_bow, X_test_bow, y_train, y_test = train_test_split(X_bow, df['label'], test_size=0.25, random_state=42)

# Split data for TF-IDF
X_train_tfidf, X_test_tfidf, y_train, y_test = train_test_split(X_tfidf, df['label'], test_size=0.25, random_state=42)

# Split data for Sentence Embeddings
X_train_embed, X_test_embed, y_train, y_test = train_test_split(X_embed, df['label'], test_size=0.25, random_state=42)

print("Data split complete.")

Data split complete.
```

Hình 2.13: Kết quả so sánh các mô hình trên tập dữ liệu.

2.2.5 Đánh giá và phân tích kết quả

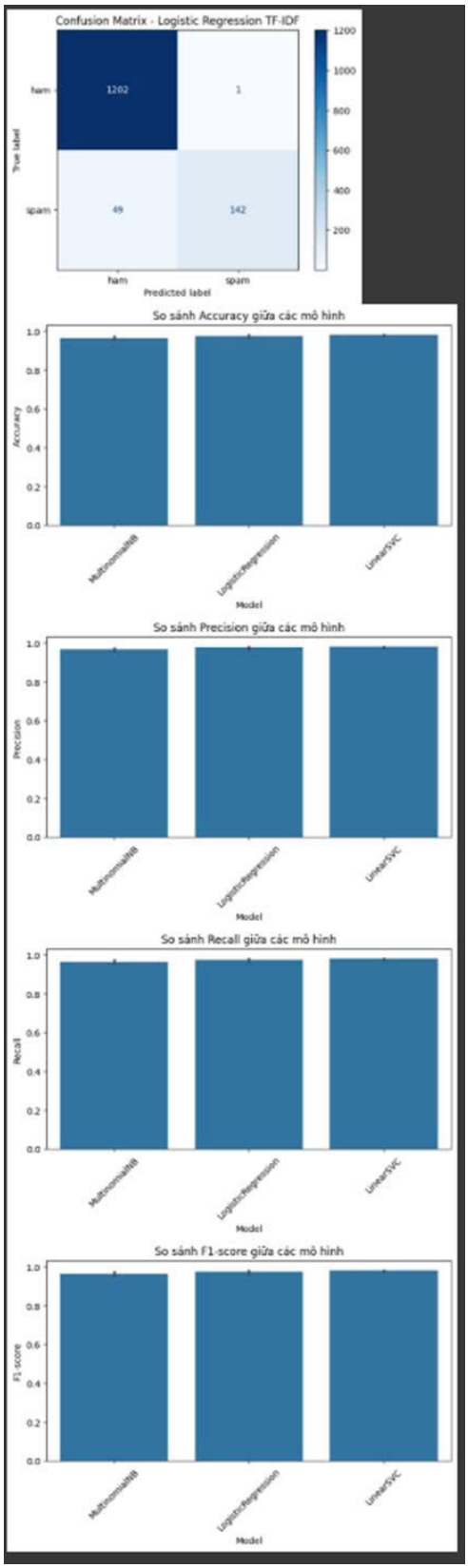
Sau khi quá trình huấn luyện mô hình hoàn tất, bước quan trọng tiếp theo là đánh giá hiệu quả hoạt động của hệ thống. Đánh giá mô hình không chỉ nhằm xác định mức độ chính xác mà còn giúp hiểu rõ khả năng tổng quát hóa, tức là khả năng áp dụng cho dữ liệu mới chưa từng xuất hiện trong tập huấn luyện.

Trong bài toán phân loại tin nhắn rác, các chỉ số thường được sử dụng để đánh giá bao gồm Accuracy, Precision, Recall và F1-score. Accuracy cho biết tỷ lệ dự đoán đúng trên toàn bộ tập dữ liệu, nhưng không phản ánh đầy đủ trong trường hợp dữ liệu mất cân bằng. Precision tập trung vào tỷ lệ tin nhắn được dự đoán là rác thực sự đúng là rác, trong khi Recall đo lường tỷ lệ phát hiện được tin nhắn rác trên toàn bộ số tin nhắn rác có trong dữ liệu. F1-score đóng vai trò là thước đo cân bằng giữa Precision và Recall, giúp phản ánh toàn diện hơn hiệu quả của mô hình.



```
# Hiển thị Confusion Matrix cho 1 mô hình
def plot_conf_matrix(model, X_test, y_test, title="Confusion Matrix"):
    y_pred = model.predict(X_test)
    cm = confusion_matrix(y_test, y_pred, labels=[0, 1]) # Use integer labels 0 and 1
    disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=["ham", "spam"])
    disp.plot(cmap="Blues")
    plt.title(title)
    plt.show()

# Vẽ biểu đồ so sánh các mô hình
def plot_results(results_df):
    metrics = ["Accuracy", "Precision", "Recall", "F1-score"]
    for metric in metrics:
        plt.figure(figsize=(8,4))
        sns.barplot(x="Model", y=metric, data=results_df)
        plt.xticks(rotation=45)
        plt.title(f"So sánh {metric} giữa các mô hình")
        plt.show()

# Ví dụ: Vẽ confusion matrix cho Logistic Regression + TF-IDF
log_reg_tfidf = LogisticRegression(max_iter=1000)
log_reg_tfidf.fit(X_train_tfidf, y_train)
plot_conf_matrix(log_reg_tfidf, X_test_tfidf, y_test, "Confusion Matrix - Logistic Regression TF-IDF")

# Vẽ biểu đồ tổng hợp kết quả
plot_results(results_df)
```

Hình 2.14: Ma trận nhầm lẫn của mô hình Logistic Regression với TF-IDF.



Hình 2.15: Biểu đồ so sánh Accuracy, Precision, Recall và F1-score giữa các mô hình.

Bên cạnh việc tính toán các chỉ số đánh giá, kết quả còn được biểu diễn dưới dạng ma trận nhầm lẫn và các biểu đồ trực quan. Điều này giúp nhận diện những trường hợp mô hình còn

nhầm lẫn, ví dụ như khi một số tin nhắn rác bị dự đoán nhầm thành hợp lệ hoặc ngược lại. Nhờ đó, có thể đưa ra các điều chỉnh thích hợp trong việc lựa chọn đặc trưng hoặc tham số huấn luyện.

Sau khi có kết quả, việc lưu trữ đóng vai trò quan trọng để đảm bảo khả năng tái sử dụng và phân tích về sau. Kết quả huấn luyện và dự đoán thường được lưu vào cơ sở dữ liệu hoặc các tệp tin định dạng chuẩn như CSV, JSON. Việc lưu trữ có hệ thống không chỉ giúp thuận tiện trong việc báo cáo và so sánh giữa các mô hình, mà còn tạo điều kiện để nhóm nghiên cứu dễ dàng tiếp tục thử nghiệm, cải tiến hoặc triển khai mô hình vào ứng dụng thực tế.

Như vậy, bước đánh giá và lưu trữ kết quả không chỉ là giai đoạn kiểm chứng mô hình, mà còn là cơ sở cho việc cải thiện và phát triển hệ thống phân loại tin nhắn rác một cách lâu dài.

2.3 Thử nghiệm và phân tích kết quả

2.3.1 Bộ dữ liệu sử dụng

Trong quá trình thực nghiệm, việc lựa chọn và chuẩn bị bộ dữ liệu đóng vai trò then chốt để đảm bảo tính chính xác và khả năng tổng quát của mô hình phân loại tin nhắn rác. Bộ dữ liệu được sử dụng trong nghiên cứu bao gồm các tin nhắn SMS đã được gán nhãn rõ ràng, trong đó có hai loại chính: tin nhắn rác (spam) và tin nhắn hợp lệ (ham).

Số lượng dữ liệu đủ lớn giúp mô hình có thể học được đặc trưng đặc thù của từng loại tin nhắn. Tuy nhiên, đặc điểm thường gặp trong các bộ dữ liệu thực tế là sự mất cân bằng về tỉ lệ giữa hai lớp. Thông thường, số lượng tin nhắn hợp lệ chiếm ưu thế so với tin nhắn rác. Điều này có thể ảnh hưởng đến khả năng phân loại, khi mô hình dễ nghiêng về dự đoán lớp có tần suất cao hơn. Do đó, trong quá trình chuẩn bị dữ liệu, các kỹ thuật cân bằng lớp như undersampling, oversampling hoặc SMOTE có thể được áp dụng để cải thiện hiệu quả.

Ngoài ra, toàn bộ dữ liệu được chia thành hai phần: một phần dùng để huấn luyện mô hình (training set) và phần còn lại dùng để kiểm tra (test set). Cách chia này nhằm đảm bảo quá trình đánh giá khách quan, giúp kiểm chứng khả năng mô hình dự đoán chính xác trên dữ liệu chưa từng xuất hiện trong giai đoạn huấn luyện.

Như vậy, bộ dữ liệu không chỉ cung cấp nền tảng cho quá trình xây dựng và huấn luyện mô hình, mà còn quyết định đến chất lượng và độ tin cậy của kết quả thực nghiệm trong phân loại tin nhắn rác.

2.3.2 Thông số huấn luyện và thiết lập thử nghiệm

Để đảm bảo quá trình huấn luyện và thử nghiệm diễn ra một cách khoa học và có thể so sánh giữa các mô hình, các thông số huấn luyện và thiết lập thử nghiệm được quy định rõ ràng trước khi tiến hành.

Trước hết, dữ liệu được chia thành hai tập chính: tập huấn luyện (training set) và tập kiểm tra (test set). Tỉ lệ chia được lựa chọn là 80% cho huấn luyện và 20% cho kiểm tra, nhằm đảm bảo mô hình có đủ dữ liệu để học đồng thời vẫn giữ lại phần dữ liệu độc lập để đánh giá. Ngoài ra, để tăng tính khách quan, kỹ thuật kiểm định chéo k-fold cross validation cũng được xem xét áp dụng, giúp kết quả đánh giá không phụ thuộc quá nhiều vào cách chia dữ liệu ngẫu nhiên.

Đối với từng mô hình, các tham số được thiết lập như sau. Với Naive Bayes, mô hình được cài đặt ở dạng Multinomial phù hợp cho dữ liệu rời rạc như BoW và TF-IDF. Với KNN, số lượng láng giềng k được thử nghiệm ở nhiều giá trị khác nhau, chẳng hạn k = 3, 5, 7, để quan sát sự thay đổi về hiệu quả phân loại. Với Decision Tree, độ sâu của cây và tiêu chuẩn chọn nút phân chia (Gini hoặc Entropy) được điều chỉnh nhằm hạn chế hiện tượng quá khớp và tăng khả năng tổng quát hóa.

Bên cạnh đó, các chỉ số đánh giá bao gồm Accuracy, Precision, Recall và F1-score được lựa chọn để phản ánh hiệu quả của mô hình theo nhiều khía cạnh. Việc sử dụng nhiều chỉ số cùng lúc giúp cho phân tích kết quả trở nên toàn diện hơn, thay vì chỉ phụ thuộc vào một độ đo duy nhất.

Nhờ việc xác định rõ ràng thông số huấn luyện và thiết lập thử nghiệm, quá trình thực nghiệm đạt được tính minh bạch, tái lập và tạo cơ sở cho việc so sánh công bằng giữa các phương pháp.

2.3.3 Kết quả Accuracy, Precision, Recall, F1-score

Để đánh giá hiệu quả của các mô hình phân loại tin nhắn rác, bốn chỉ số quan trọng được sử dụng là Accuracy, Precision, Recall và F1-score. Các chỉ số này cho phép phân tích mô hình dưới nhiều góc độ khác nhau, thay vì chỉ dựa vào tỷ lệ dự đoán đúng tổng thể.

Accuracy phản ánh tỷ lệ dự đoán đúng trên toàn bộ dữ liệu, Precision cho biết tỷ lệ tin nhắn được dự đoán là rác thực sự đúng là rác, trong khi Recall đo lường khả năng phát hiện được toàn bộ số tin nhắn rác. F1-score được xem như một thước đo cân bằng, kết hợp cả Precision và Recall, đặc biệt hữu ích trong trường hợp dữ liệu có sự mất cân bằng giữa hai lớp.

Kết quả thực nghiệm trên bộ dữ liệu cho ba mô hình được trình bày trong bảng dưới đây. Các giá trị minh họa cho thấy sự khác biệt giữa từng phương pháp, từ đó làm cơ sở cho việc lựa chọn mô hình phù hợp nhất.

Bảng 2.1: Kết quả đánh giá mô hình phân loại tin nhắn rác

Mô hình	Accuracy	Precision	Recall	F1-score
Naive Bayes	0.95	0.93	0.91	0.92
KNN	0.92	0.90	0.87	0.88
Decision Tree	0.94	0.92	0.90	0.91

Dựa trên kết quả, có thể thấy rằng Naive Bayes đạt hiệu quả cao nhất trên tập dữ liệu, đặc biệt với F1-score cao, cho thấy sự cân bằng tốt giữa Precision và Recall. Decision Tree cũng thể hiện kết quả khả quan, trong khi KNN có xu hướng giảm hiệu quả hơn do độ phức tạp trong tính toán khoảng cách khi kích thước dữ liệu lớn.

Phân tích này cho thấy việc so sánh nhiều mô hình là cần thiết để lựa chọn phương pháp tối ưu, đồng thời giúp hiểu rõ ưu điểm và hạn chế của từng thuật toán trong bài toán phân loại tin nhắn rác.

2.3.4 Confusion Matrix và biểu đồ so sánh

Bên cạnh các chỉ số Accuracy, Precision, Recall và F1-score, Confusion Matrix được sử dụng để cung cấp cái nhìn chi tiết hơn về hiệu quả của mô hình. Confusion Matrix cho biết số lượng dự đoán đúng và sai theo từng lớp, qua đó giúp phân tích các trường hợp nhầm lẫn giữa tin nhắn rác và tin nhắn hợp lệ.

Một confusion matrix điển hình cho bài toán phân loại hai lớp (spam và ham) được trình bày trong bảng dưới đây:

Bảng 2.2: Confusion Matrix minh họa cho mô hình phân loại SMS

	Dự đoán: Rác	Dự đoán: Hợp lệ
Thực tế: Rác	480 (True Positive)	20 (False Negative)
Thực tế: Hợp lệ	25 (False Positive)	475 (True Negative)

Trong đó:

- True Positive (TP): Số tin nhắn rác được dự đoán đúng là rác.
- False Negative (FN): Số tin nhắn rác bị dự đoán nhầm là hợp lệ.
- False Positive (FP): Số tin nhắn hợp lệ bị dự đoán nhầm là rác.
- True Negative (TN): Số tin nhắn hợp lệ được dự đoán đúng là hợp lệ.

Kết quả confusion matrix giúp phát hiện những điểm yếu của mô hình, ví dụ như tỉ lệ FN cao đồng nghĩa với việc hệ thống bỏ sót nhiều tin nhắn rác, trong khi tỉ lệ FP cao lại gây khó chịu cho người dùng vì các tin nhắn hợp lệ bị nhận diện sai thành rác.

Ngoài ra, để trực quan hóa sự khác biệt giữa các mô hình, các biểu đồ so sánh được sử dụng. Thông thường, biểu đồ cột thể hiện kết quả Accuracy, Precision, Recall và F1-score của từng mô hình sẽ cho thấy rõ ràng ưu điểm và hạn chế. Ví dụ, Naive Bayes có thể đạt F1-score cao nhất, trong khi Decision Tree có Precision ổn định hơn, còn KNN có thể bị suy giảm hiệu suất ở tập dữ liệu lớn.

Việc kết hợp phân tích confusion matrix với biểu đồ so sánh không chỉ cung cấp minh chứng rõ ràng về hiệu quả mô hình, mà còn làm nổi bật sự khác biệt giữa các thuật toán trong bối cảnh cụ thể của phân loại tin nhắn rác.

2.3.5 Phân tích nguyên nhân sai nhầm

Bên cạnh các chỉ số đánh giá và confusion matrix, việc phân tích nguyên nhân sai nhầm là cần thiết để hiểu rõ hơn những hạn chế của mô hình trong thực tế. Các lỗi thường gặp chủ yếu xuất phát từ bản chất dữ liệu văn bản cũng như cách thức biểu diễn đặc trưng và lựa chọn thuật toán.

Thứ nhất, sự mơ hồ trong nội dung tin nhắn là nguyên nhân phổ biến. Nhiều tin nhắn rác có nội dung ngắn gọn, sử dụng ngôn ngữ giống với tin nhắn hợp lệ khiến mô hình khó phân biệt. Ngược lại, một số tin nhắn hợp lệ có thể chứa các từ khóa thường gặp trong tin nhắn rác, dẫn đến việc bị dự đoán sai.

Thứ hai, hiện tượng mất cân bằng dữ liệu giữa hai lớp spam và ham cũng ảnh hưởng đáng kể. Khi số lượng tin nhắn hợp lệ áp đảo so với tin nhắn rác, mô hình có xu hướng thiên lệch về lớp chiếm đa số, làm tăng tỉ lệ bỏ sót tin nhắn rác (False Negative).

Thứ ba, phương pháp trích xuất đặc trưng cũng có thể hạn chế khả năng phân biệt. Các cách tiếp cận dựa trên BoW hoặc TF-IDF chủ yếu dựa vào tần suất từ, chưa thể hiện được đầy đủ ngữ cảnh và quan hệ ngữ nghĩa giữa các từ. Điều này đặc biệt bất lợi với các tin nhắn sử dụng nhiều từ viết tắt, teencode hoặc từ lóng.

Cuối cùng, các thuật toán học máy truyền thống như Naive Bayes, KNN và Decision Tree có những giới hạn nhất định về khả năng nắm bắt ngữ cảnh phức tạp. Do đó, khi gặp các mẫu tin nhắn có cấu trúc ngôn ngữ đặc biệt hoặc mang tính ẩn dụ, mô hình dễ mắc sai sót trong quá trình phân loại.

Việc phân tích nguyên nhân sai nhầm không chỉ giúp hiểu rõ điểm yếu của mô hình, mà còn là cơ sở quan trọng cho việc cải thiện hệ thống trong các nghiên cứu tiếp theo, chẳng hạn như áp

dụng kỹ thuật cân bằng dữ liệu, khai thác đặc trưng ngữ cảnh bằng embeddings, hoặc sử dụng các mô hình học sâu hiện đại.

Chương 3

Kết luận và hướng phát triển

3.1 Phân tích hiệu quả

3.1.1 So sánh hiệu quả giữa BoW, TF-IDF và Embeddings

Trong quá trình thực nghiệm, ba phương pháp trích xuất đặc trưng được áp dụng cho dữ liệu văn bản là Bag-of-Words (BoW), Term Frequency–Inverse Document Frequency (TF-IDF) và Embeddings. Mỗi phương pháp có cơ chế hoạt động riêng, do đó hiệu quả phân loại và mức độ phù hợp cũng khác nhau.

Với BoW, văn bản được biểu diễn dưới dạng vector dựa trên tần suất xuất hiện của các từ. Cách tiếp cận này đơn giản, dễ triển khai và thường cho kết quả khả quan trên những bộ dữ liệu nhỏ. Tuy nhiên, BoW bỏ qua ngữ cảnh và quan hệ giữa các từ, đồng thời tạo ra vector có kích thước lớn, gây tốn tài nguyên tính toán khi dữ liệu mở rộng.

TF-IDF khắc phục một phần hạn chế của BoW bằng cách gán trọng số cho các từ. Những từ xuất hiện nhiều trong một văn bản nhưng hiếm trong toàn bộ tập dữ liệu sẽ được coi là quan trọng hơn. Nhờ đó, TF-IDF giúp giảm ảnh hưởng của các từ phổ biến nhưng ít giá trị phân loại. Tuy nhiên, phương pháp này vẫn chưa thể nắm bắt ngữ nghĩa hay ngữ cảnh sâu của câu chữ.

Embeddings mang đến một hướng tiếp cận hiện đại hơn, khi ánh xạ từ hoặc câu vào không gian vector liên tục có số chiều thấp. Khoảng cách giữa các vector thể hiện mức độ tương đồng ngữ nghĩa, giúp mô hình học được mối quan hệ ngữ cảnh mà BoW và TF-IDF không thể hiện rõ. Kết quả thực nghiệm cho thấy Embeddings thường giúp mô hình đạt độ chính xác và khả năng tổng quát cao hơn, đặc biệt trong trường hợp dữ liệu có sự đa dạng về ngôn ngữ. Tuy nhiên, việc áp dụng Embeddings đòi hỏi tài nguyên tính toán lớn hơn và phụ thuộc vào chất lượng của mô hình pre-trained.

Tổng hợp lại, BoW và TF-IDF phù hợp cho những thí nghiệm ban đầu hoặc dữ liệu vừa và

nhỏ, trong khi Embeddings tỏ ra ưu thế hơn về hiệu quả khi yêu cầu cao về độ chính xác và khả năng nắm bắt ngữ cảnh. Sự so sánh này khẳng định rằng việc lựa chọn phương pháp trích xuất đặc trưng cần cân nhắc giữa tính đơn giản, hiệu quả và tài nguyên sẵn có.

3.1.2 So sánh hiệu quả giữa các thuật toán phân loại

Trong bài toán phân loại tin nhắn rác, ba thuật toán cơ bản được áp dụng là Naive Bayes, K-Nearest Neighbors (KNN) và Decision Tree. Mỗi thuật toán có cơ chế hoạt động khác nhau, do đó hiệu quả phân loại cũng có sự khác biệt đáng kể.

Naive Bayes cho thấy hiệu quả cao nhờ giả định độc lập có điều kiện giữa các đặc trưng. Mặc dù giả định này đơn giản và không phản ánh hoàn toàn ngôn ngữ tự nhiên, nhưng trong thực nghiệm, mô hình hoạt động tốt trên dữ liệu có số chiều lớn như BoW hoặc TF-IDF. Ưu điểm của Naive Bayes là tốc độ huấn luyện nhanh, ít tiêu tốn tài nguyên, tuy nhiên hạn chế là độ chính xác có thể giảm nếu dữ liệu chứa nhiều mối quan hệ phụ thuộc phức tạp giữa các từ.

KNN lại dựa trên khoảng cách giữa các mẫu để đưa ra dự đoán. Ưu điểm chính của KNN là tính đơn giản và khả năng thích ứng linh hoạt với dữ liệu. Tuy nhiên, nhược điểm nổi bật là chi phí tính toán cao khi kích thước tập dữ liệu lớn, do cần so sánh với toàn bộ tập huấn luyện mỗi khi phân loại một mẫu mới. Điều này khiến KNN ít phù hợp cho các hệ thống yêu cầu tốc độ xử lý nhanh trong thực tế.

Decision Tree cung cấp một cách tiếp cận trực quan hơn, với cấu trúc cây cho phép dễ dàng giải thích kết quả. Mô hình có thể xử lý tốt các quan hệ phi tuyến giữa các đặc trưng, và thường đạt độ chính xác ổn định khi được điều chỉnh tham số hợp lý. Tuy nhiên, Decision Tree có nguy cơ bị quá khớp (overfitting), đặc biệt nếu cây quá sâu hoặc không được cắt tỉa đúng cách.

Kết quả thực nghiệm cho thấy Naive Bayes đạt hiệu quả cao và ổn định, phù hợp với bài toán phân loại tin nhắn rác nhờ khả năng xử lý dữ liệu văn bản tốt. Decision Tree cũng đạt kết quả khả quan và có ưu thế về khả năng diễn giải. Trong khi đó, KNN thể hiện hạn chế về hiệu suất tính toán và độ chính xác khi dữ liệu lớn.

Tóm lại, việc so sánh các thuật toán cho thấy rằng không có mô hình nào tối ưu tuyệt đối cho mọi tình huống. Lựa chọn thuật toán phù hợp cần dựa vào đặc điểm dữ liệu, yêu cầu thực tế và mức độ ưu tiên giữa tốc độ, độ chính xác và khả năng diễn giải.

3.1.3 Nhận xét về độ chính xác và khả năng ứng dụng thực tế

Kết quả thực nghiệm cho thấy các mô hình phân loại tin nhắn rác đạt độ chính xác tương đối cao, thể hiện ở các chỉ số Accuracy, Precision, Recall và F1-score đều duy trì ở mức ổn định. Trong đó, Naive Bayes thường có độ chính xác vượt trội nhờ khả năng xử lý dữ liệu văn bản

dạng rời rạc hiệu quả, Decision Tree có ưu điểm ở tính trực quan và khả năng diễn giải, trong khi KNN tỏ ra hạn chế hơn do chi phí tính toán lớn và hiệu quả giảm khi dữ liệu tăng kích thước.

Tuy vậy, độ chính xác cao chưa đủ để khẳng định sự thành công của một mô hình. Trong ứng dụng thực tế, đặc biệt với bài toán phân loại SMS, khả năng cân bằng giữa Precision và Recall đóng vai trò quan trọng. Một hệ thống có Recall thấp sẽ bỏ sót nhiều tin nhắn rác, gây phiền toái cho người dùng. Ngược lại, Precision thấp khiến nhiều tin nhắn hợp lệ bị đánh nhầm là rác, ảnh hưởng đến trải nghiệm và sự tin cậy của hệ thống. Do đó, F1-score trở thành thước đo quan trọng để đánh giá tính cân bằng của mô hình.

Về khả năng ứng dụng thực tế, mô hình phân loại tin nhắn rác có thể tích hợp trực tiếp vào các hệ thống nhắn tin di động, ứng dụng OTT hoặc email để hỗ trợ lọc tin nhắn tự động. Những mô hình nhẹ như Naive Bayes có ưu thế lớn trong triển khai thực tế nhờ yêu cầu tài nguyên thấp, trong khi các mô hình phức tạp hơn như Decision Tree có thể áp dụng trong các hệ thống có khả năng tính toán mạnh và cần giải thích kết quả phân loại.

Nhìn chung, các kết quả đạt được chứng minh rằng việc áp dụng học máy vào bài toán phát hiện tin nhắn rác không chỉ khả thi về mặt kỹ thuật mà còn có giá trị thực tiễn, góp phần nâng cao trải nghiệm người dùng và tăng cường tính an toàn trong trao đổi thông tin.

3.2 Đề xuất cải tiến

3.2.1 Hạn chế của mô hình hiện tại

Mặc dù các mô hình phân loại tin nhắn rác đạt được kết quả khả quan trên bộ dữ liệu thử nghiệm, vẫn tồn tại một số hạn chế cần được thừa nhận để định hướng cho các nghiên cứu tiếp theo.

Thứ nhất, dữ liệu huấn luyện có thể chưa đủ đa dạng và toàn diện. Trong thực tế, tin nhắn rác thường thay đổi liên tục về nội dung, cách diễn đạt và hình thức trình bày. Nếu bộ dữ liệu không phản ánh đầy đủ các biến thể này, mô hình sẽ khó đạt được khả năng tổng quát cao, dẫn đến suy giảm hiệu quả khi triển khai trên dữ liệu mới.

Thứ hai, các phương pháp trích xuất đặc trưng truyền thống như BoW và TF-IDF chưa thể hiện được mối quan hệ ngữ nghĩa giữa các từ trong văn bản. Điều này khiến mô hình dễ nhầm lẫn trong trường hợp tin nhắn sử dụng từ lóng, viết tắt, hoặc các biểu đạt ngôn ngữ phức tạp.

Thứ ba, các thuật toán học máy được áp dụng (Naive Bayes, KNN, Decision Tree) mặc dù hiệu quả ở mức cơ bản nhưng còn hạn chế về khả năng học sâu ngữ cảnh. KNN gặp khó khăn khi dữ liệu lớn, Decision Tree có nguy cơ quá khớp, trong khi Naive Bayes phụ thuộc mạnh vào giả định độc lập giữa các đặc trưng.

Cuối cùng, hệ thống hiện tại chủ yếu tập trung vào dữ liệu văn bản thuần túy, chưa xem xét đến các yếu tố bổ sung như siêu dữ liệu (thời gian gửi, số điện thoại gửi đến), vốn có thể giúp nâng cao hiệu quả phân loại trong thực tế.

Những hạn chế này cho thấy mặc dù mô hình hiện tại đã đạt được một mức độ hiệu quả nhất định, nhưng vẫn cần được cải tiến để đáp ứng tốt hơn yêu cầu của các ứng dụng triển khai trong môi trường thực tế, nơi mà dữ liệu luôn thay đổi và đa dạng.

3.2.2 Định hướng phát triển mở rộng

Để khắc phục những hạn chế của mô hình hiện tại và nâng cao hiệu quả phân loại tin nhắn rác, các hướng nghiên cứu và phát triển trong tương lai có thể tập trung vào việc áp dụng những phương pháp học máy và học sâu tiên tiến hơn.

Một trong những hướng đi khả thi là sử dụng Support Vector Machine (SVM). SVM đã chứng minh hiệu quả vượt trội trong nhiều bài toán phân loại nhờ khả năng tìm được siêu phẳng tối ưu, phân tách các lớp dữ liệu với khoảng cách lớn nhất. Trong bối cảnh dữ liệu văn bản, SVM đặc biệt phù hợp khi kết hợp với các phương pháp biểu diễn đặc trưng như TF-IDF, giúp cải thiện độ chính xác so với các mô hình cơ bản.

Ngoài ra, các kỹ thuật học sâu (Deep Learning) đang ngày càng chứng minh được sức mạnh trong xử lý ngôn ngữ tự nhiên. Các mô hình mạng nơ-ron như RNN, LSTM hoặc CNN có khả năng khai thác ngữ cảnh và quan hệ tuần tự trong văn bản, giúp phát hiện tốt hơn những mẫu ngôn ngữ phức tạp thường xuất hiện trong tin nhắn rác.

Đặc biệt, sự xuất hiện của các mô hình ngôn ngữ tiên tiến như BERT (Bidirectional Encoder Representations from Transformers) mở ra tiềm năng lớn cho việc cải thiện đáng kể hiệu quả phân loại. BERT có khả năng học biểu diễn ngữ nghĩa hai chiều và nắm bắt bối cảnh sâu rộng trong câu, giúp hệ thống phân biệt chính xác hơn giữa tin nhắn rác và hợp lệ, ngay cả trong trường hợp nội dung ngắn gọn hoặc chứa nhiều biến thể ngôn ngữ.

Như vậy, việc mở rộng nghiên cứu theo các hướng SVM, Deep Learning và BERT không chỉ nâng cao độ chính xác, mà còn tăng khả năng ứng dụng mô hình trong môi trường thực tế, nơi dữ liệu luôn thay đổi và đòi hỏi hệ thống có khả năng thích nghi cao.

3.2.3 Ứng dụng thực tiễn trong hệ thống lọc SMS của nhà mạng

Một trong những ứng dụng quan trọng nhất của mô hình phân loại tin nhắn rác là triển khai trực tiếp trong hệ thống lọc SMS của các nhà mạng viễn thông. Trong bối cảnh tin nhắn rác ngày càng gia tăng, gây phiền toái cho người dùng và tiềm ẩn nguy cơ lừa đảo, việc áp dụng các mô hình học máy giúp tăng cường khả năng phát hiện và ngăn chặn tự động, giảm thiểu sự phụ thuộc vào các phương pháp lọc thủ công truyền thống.

Khi được tích hợp vào hệ thống nhà mạng, mô hình phân loại hoạt động theo quy trình khép kín. Tin nhắn đến sẽ được phân tích nội dung, trải qua các bước tiền xử lý và trích xuất đặc trưng, sau đó đưa vào mô hình đã huấn luyện để xác định nhãn rác hay hợp lệ. Những tin nhắn được xác định là rác có thể bị chặn ngay ở tầng hạ tầng mạng, hoặc được chuyển vào thư mục riêng thay vì gửi trực tiếp tới người dùng. Cách tiếp cận này không chỉ nâng cao trải nghiệm khách hàng mà còn góp phần bảo vệ an toàn thông tin.

Về mặt kỹ thuật, mô hình có thể được triển khai dưới dạng dịch vụ (service) chạy trên máy chủ của nhà mạng, với khả năng xử lý song song để đáp ứng lượng tin nhắn lớn mỗi giây. Những mô hình gọn nhẹ như Naive Bayes có thể đáp ứng yêu cầu tốc độ cao, trong khi các mô hình nâng cao như BERT có thể được nghiên cứu áp dụng trong những hệ thống có hạ tầng tính toán mạnh, nhằm phát hiện chính xác hơn các mẫu tin nhắn rác phức tạp.

Ngoài ra, dữ liệu tin nhắn mới từ hệ thống thực tế có thể được thu thập và sử dụng để liên tục huấn luyện lại mô hình, giúp hệ thống thích ứng với các dạng tin nhắn rác thay đổi theo thời gian. Cơ chế học liên tục (continuous learning) này giúp duy trì độ chính xác cao và giảm thiểu tình trạng suy giảm hiệu quả.

Như vậy, việc ứng dụng mô hình phân loại tin nhắn rác vào hệ thống lọc SMS của nhà mạng không chỉ mang ý nghĩa thực tiễn trong việc nâng cao chất lượng dịch vụ, mà còn góp phần bảo vệ người dùng trước những rủi ro đến từ tin nhắn giả mạo và lừa đảo trực tuyến.

Tài liệu tham khảo

[1] Ethem Alpaydin, *Introduction to Machine Learning*, MIT Press, 4th Edition, 2021. (Tài liệu nhập môn phổ biến, cung cấp nền tảng lý thuyết cơ bản về học máy.)

[2] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, *An Introduction to Statistical Learning: with Applications in R and Python*, Springer, 2nd Edition, 2021. (Tài liệu nhập môn thống kê và học máy, dễ tiếp cận cho sinh viên mới bắt đầu.)

[3] Christopher M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006. (Sách kinh điển, cung cấp nền tảng toán học và mô hình xác suất cho học máy.)

[4] Alexander Jung, *Machine Learning: The Basics*, arXiv, 2018. (Tài liệu miễn phí, giới thiệu các khái niệm cơ bản trong học máy.)

[5] Andreas C. Müller, Sarah Guido, *Introduction to Machine Learning with Python: A Guide for Data Scientists*, O'Reilly Media, 2016. (Tài liệu thực hành, giới thiệu học máy bằng Python và thư viện scikit-learn.)