UỶ BAN NHÂN DÂN

THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC SÀI GÒN

BÁO CÁO TIỂU LUẬN

XỬ LÝ NGÔN NGỮ TỰ NHIÊN

TÊN ĐỀ TÀI: ỨNG DỤNG MÔ HÌNH NAIVE BAYES, SUPPORT VECTOR MACHINE, LOGISTIC REGRESSION VÀO PHÂN LOẠI EMAIL RÁC

Thông tin sinh viên:

3121411171 Vũ Bình Phước DCT121C3

Giảng viên: Vũ Ngọc Thanh Sang

Thành phố Hồ Chí Minh, ngày 26 tháng 12 năm 2024

MỤC LỤC

DANH MỤC BẢNG BIỂU, HÌNH ẢNH

DANH MỤC CÁC KÝ HIỆU, VIẾT TẮT

Chữ viết tắt	Nguyên nghĩa
NLP	Natural language processing
SVM	Support Vector Machine
TF-IDF	Term Frequency-Inverse Document Frequency
UCE	Unsolicited Commercial Email
AI	Artificial Intelligence

TÓM TẮT ĐỀ TÀI

Phát triển hệ thống tự động phân loại email thành hợp lệ (ham) hoặc rác (spam) bằng cách ứng dụng các kỹ thuật NLP, giúp giảm phiền toái và nâng cao bảo mật thông tin.

Trình bày nội dung, kỹ thuật và phương pháp thực hiện đề tài. Trong quá trình thực hiện chúng ta cần có các bước như sau: thực hiện thu thập dữ liệu. sử dụng các phương pháp, kỹ thuật tiền xử lý dữ liệu, xây dựng các mô hình (Naive Bayes, SVM, Logistic Regression), đánh giá hiệu quả của từng mô hình thông qua các số liệu cùng biểu đồ, chọn mô hình hiệu quả nhất.

Từ kết quả thu được, chúng ta có thể thấy được mô hình nào hiệu quả nhất, cũng như tìm ra được hướng phát triển cho tương lai.

MỞ ĐẦU

Email rác có thể mang đến một số rủi ro, không an toàn về bảo mật như: lừa đảo, phát tán mã độc, ăn cắp danh tính. Ngoài ra, chúng còn làm hao tốn tài nguyên hệ thống cũng như làm ảnh hưởng đến thời gian, trải nghiệm của người dùng. Vì vậy để cải thiện trải nghiệm người dùng cũng như đảm bảo tính an toàn thông tin, việc phân loại Email rác trở nên cần thiết hơn bao giờ hết.

Mục tiêu mà chúng ta cần làm được là phân loại được đâu là Email hợp lệ (ham) và đâu là Email rác (spam) từ nhiều mô hình, dữ liệu khác nhau. Từ đó tìm ra mô hình có hiệu quả cao nhất nhằm ứng dụng vào thực tế.

Trong thực tế, có rất nhiều các dịch vụ Email tận dụng các mô hình của NLP vào việc phân loại, lọc, sắp xếp Email nhằm nâng cao trải nghiệm người dùng cũng như đảm bảo tính an toàn bảo mật trong quá trình giao tiếp Email giữa các người dùng.

Quá trình thực hiện và ứng dụng các mô hình của NLP vào thực tế vẫn còn nhiều khó khăn, hạn chế như: không hiểu được ngữ cảnh, chưa xử lý được nhiều ngôn ngữ, dữ liệu huấn luyện bị thiên lệch (nhiều Ham hơn Spam)...

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1.1 Email rác là gì?

Email rác thường là các thư thương mại không mong muốn (UCE). Đó là những thư được gửi hàng loạt đến nhiều người nhận cùng một lúc, chúng thường có nội dung và mục đích như: quảng cáo, tiếp thị đa cấp, quà tặng, thị trường chứng khoán...

Mục đích của các thư này đa số là để kiếm lời nhuận từ những việc như quảng cáo, tiếp thị đa cấp. Và trong số đó có một số việc vi phạm pháp luật như lừa đảo, phát tán mã độc, ăn cắp thông tin cá nhân hoặc tấn công DDoS.

Một người sử dụng Email bình thường vì sao lại nhận được rất nhiều Email rác, nguyên nhân dẫn đến việc này phần lớn là do người dùng có thể đã để địa chỉ Email, thông tin Email của mình rỏ rỉ trên mạng xã hội hoặc trên các Website không đáng tin cậy. Ngoài ra, người dùng còn có thể đã để rỏ rỉ thông tin Email của mình từ nhiều việc khác nữa.

Việc có quá nhiều Email rác có thể gây phiền toái, mất tập trung hoặc làm chúng ta mất thời gian để tìm xem Email nào là rác và Email nào là chúng ta cần. Nếu các Email chỉ gửi với mục đích quảng cáo hay tiếp thị thì chúng chỉ mang đến những phiền phức trên nhưng nếu chúng được gửi đến với mục đích lừa đảo, phát tán mã độc thì điều này rất nguy hiểm. Những mã độc này có thể làm tốn tài nguyên thiết bị của bạn, làm thiết bị của bạn trở nên đình trệ, chúng còn có thể ăn cắp thông tin cá nhân từ thiết bị của bạn. Vì vậy việc ảnh hưởng của mã độc và lừa đảo đều mang đến những tổn hại về tinh thần cũng như vật chất của chúng ta.

Để giảm thiểu những việc này xảy ra từ các Email rác, các dịch vụ Email như Gmail, Outlook đều có các công cụ lọc thư rác hiệu quả. Ngoài việc phát triển lọc thư rác, họ còn phát triển về bảo mật để có thể bảo vệ thông tin Email của người dùng. [1]

1.2 Xử lý ngôn ngữ tự nhiên (NLP) là gì?

Xử lý ngôn ngữ tự nhiên (NLP) là một lĩnh vực khoa học máy tính, đặc biệt là trong trí tuệ nhân tạo (AI), chuyên về trang bị cho máy tính khả năng hiểu văn bản và ngôn ngữ nói giống như con người.

NLP tích hợp ngôn ngữ học tính toán, sử dụng các mô hình dựa trên quy tắc của ngôn ngữ con người, với các mô hình thống kê, machine learning và deep learning. Những công nghệ kết hợp này cho phép máy tính xử lý ngôn ngữ của con người, dù ở dạng văn bản hay giọng nói và nắm bắt được ý nghĩa đầy đủ của nó, bao gồm cả ý định và cảm xúc của người nói hoặc người viết.

Các ứng dụng của NLP rất đa dạng, từ dịch ngôn ngữ và phản hồi các lệnh nói cho đến tóm tắt nhanh chóng lượng văn bản đa dạng, thường là theo thời gian thực. Rất có thể bạn đã bắt gặp công nghệ NLP thông qua hệ thống GPS điều khiển bằng giọng nói, trợ lý số, phần mềm chuyển giọng nói thành văn bản và chatbot dịch vụ khách hàng, cùng với các ứng dụng thân thiện với người tiêu dùng khác. Ngoài mục đích sử dụng trên, NLP ngày càng đóng vai trò quan trọng trong các giải pháp doanh nghiệp, tối ưu hóa hoạt động kinh doanh, nâng cao năng suất của nhân viên và đơn giản hóa các quy trình kinh doanh quan trọng.[2]

1.3 Ứng dụng NLP trong phân loại Email rác

Mặc dù tính năng phát hiện Email rác (spam) có thể không được coi là một ứng dụng nổi bật NLP, nhưng các công nghệ hàng đầu tận dụng khả năng phân loại văn bản của NLP để xem xét kỹ lưỡng các Email để tìm các mẫu ngôn ngữ cho thấy Email rác hoặc lừa đảo. Điều này bao gồm việc xác định việc sử dụng quá nhiều thuật ngữ tài chính, ngữ pháp ít đặc trưng, ngôn ngữ mang tính đe dọa, mức độ khẩn cấp không phù hợp... Việc phát hiện thư rác đã được các chuyên gia xem là vấn đề “gần như đã được giải quyết”, mặc dù trải nghiệm của từng cá nhân có thể khác nhau.[3]

1.4 Vai trò và lợi ích của việc lọc Email rác

Việc lọc Email rác giúp loại bỏ những thư không mong muốn, nâng cao chất lượng hộp thư đến và tiết kiệm thời gian cho người dùng khi xử lý email hàng ngày.

Chúng giúp ngăn chặn các Email lừa đảo, chứa mã độc, hoặc cố gắng đánh cắp danh tính, giảm nguy cơ bị mất dữ liệu cá nhân. Đồng thời, giúp bảo vệ người dùng khỏi các cuộc tấn công mạng.

Từ đó ngăn chặn việc nhận các email quảng cáo không phù hợp hoặc lặp lại nhiều lần, giúp hộp thư luôn sạch sẽ và tập trung vào những thông tin quan trọng.

Đối với cá nhân và tổ chức, việc giảm thiểu lượng Email không cần thiết giúp tập trung tốt hơn vào các công việc quan trọng. Điều này đặc biệt hữu ích cho doanh nghiệp khi phân loại và xử lý email khách hàng một cách nhanh chóng và chính xác.

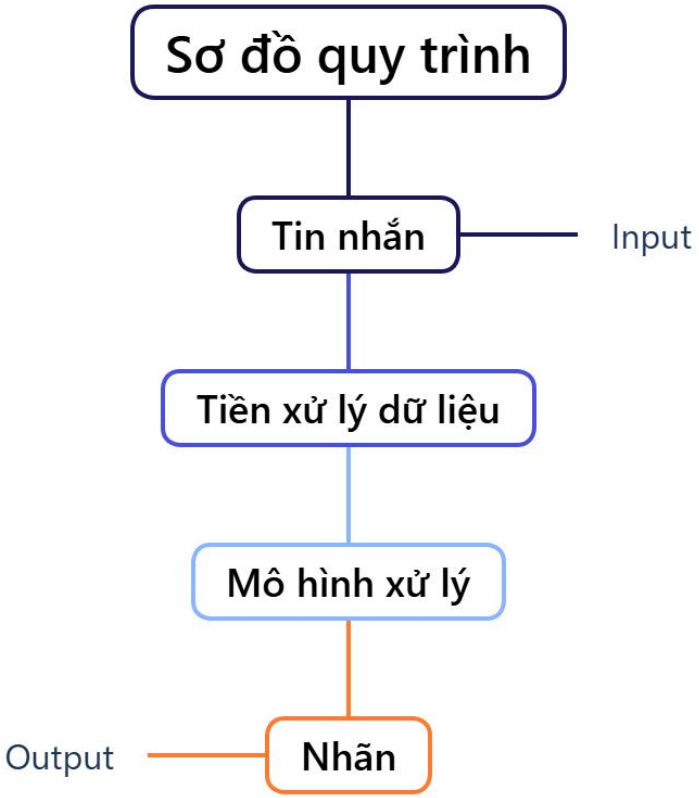
Việc giảm số lượng email rác giúp tiết kiệm chi phí lưu trữ và quản lý email, đồng thời giảm tải cho hệ thống máy chủ.

Sử dụng các công cụ lọc Email hiệu quả không chỉ cải thiện quy trình làm việc mà còn thể hiện tính chuyên nghiệp trong quản lý thông tin liên lạc.

Lọc Email rác không chỉ là biện pháp bảo vệ thông tin mà còn giúp nâng cao hiệu quả công việc và trải nghiệm người dùng. Sử dụng các công cụ và phương pháp lọc hiệu quả là yếu tố quan trọng để quản lý email một cách chuyên nghiệp.

CHƯƠNG 2: MÔ HÌNH ĐỀ XUẤT

Quy trình bắt đầu với input là các tin nhắn. Sau đó, dữ liệu được đưa qua bước tiền xử lý, bao gồm làm sạch, chuẩn hóa và chuyển đổi dữ liệu thành dạng phù hợp để phân tích. Tiếp theo, dữ liệu đã được xử lý sẽ được đưa vào mô hình xử lý, nơi áp dụng các thuật toán hoặc mô hình học máy để phân tích và phân loại. Kết quả cuối cùng là nhãn, đại diện cho đầu ra của hệ thống, giúp xác định là tin nhắn spam hay tin nhắn hợp lệ.

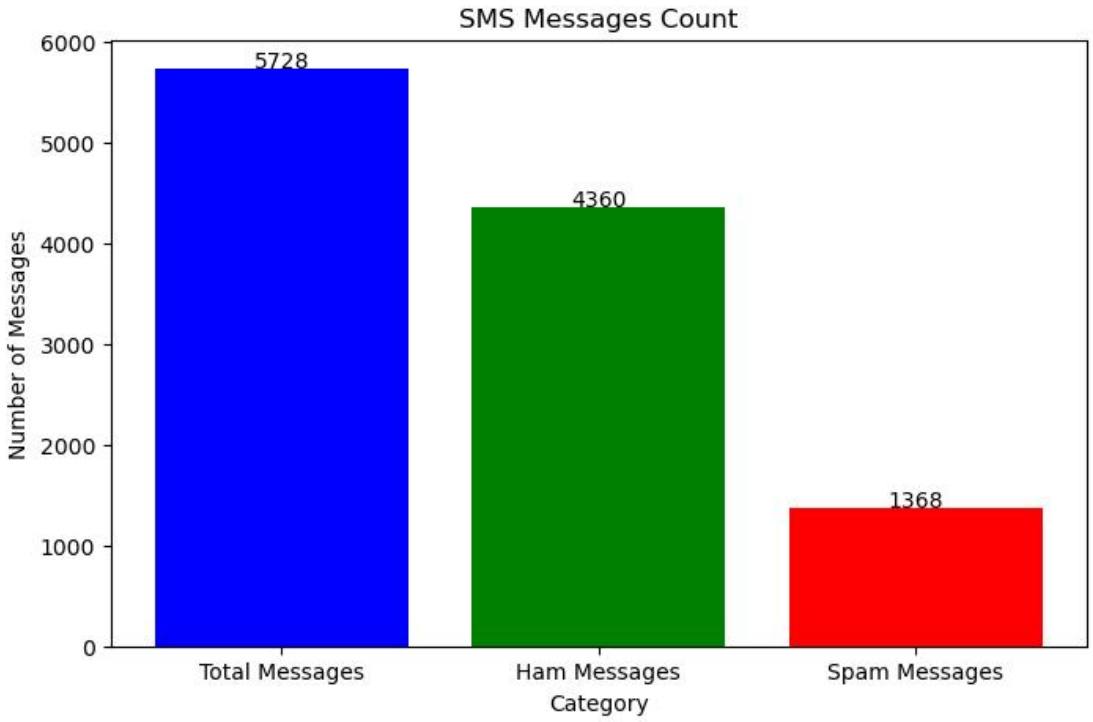


Hình 1 – Sơ đồ quy trình

2.1 Mô tả dữ liệu

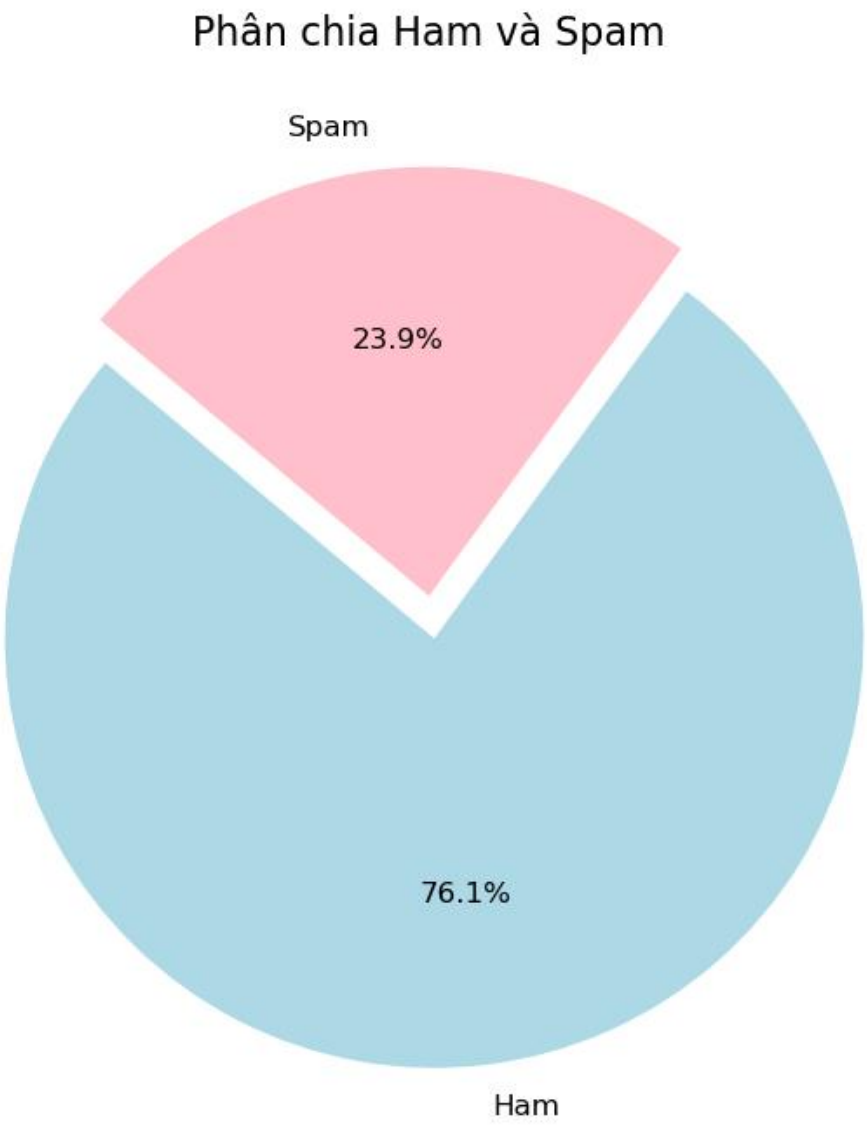
Dữ liệu mà chúng ta dùng cho việc train và test mô hình là một tập hợp các tin nhắn đã được gán nhãn, được thu thập cho nghiên cứu về tin nhắn rác.

Trong đó dữ liệu bao gồm một tập tin nhắn bằng tiếng Anh với tổng cộng 5728 tin nhắn, được gán nhãn là "ham" (hợp lệ) hoặc "spam" (tin nhắn rác).



Biểu đồ 1 – Biểu đồ cột thống kê dữ liệu

Theo biểu đồ 1, chúng ta có thể thấy các tin nhắn Ham chiếm đa số với số lượng là 4360 trên tổng số 5728 tin nhắn, còn tin nhắn Spam là 1368 tin nhắn trên tổng 5728 tin nhắn.



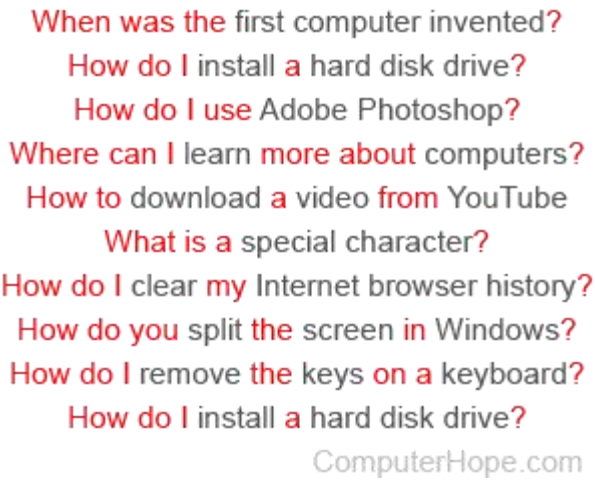
Biểu đồ 2 – Biểu đồ tròn thống kê tỉ lệ Email có nhãn Ham và Spam

Từ biểu đồ tròn 2 ta có thể thấy tỉ lệ phần trăm của Ham và Spam chiếm trong tổng số tin nhắn của dữ liệu. Ham chiếm 76.1% và Spam chiếm 23.9%.

2.2 Tiền xử lý dữ liệu

Stopwords

Stopwords là những từ phổ biến không đóng góp đáng kể vào ý nghĩa đối với việc xử lý và phân tích nội dung của văn bản. Các từ này thường bao gồm: and, or, is, the, but…



Hình 2 – Hình minh họa Stopwords

Trong NLP, những từ này thường được loại bỏ để tối ưu hiệu suất, tập trung vào thông tin quan trọng, tăng độ chính xác. [4]

Converting text to lowercase

Lowercase (viết thường) là một kỹ thuật cơ bản được sử dụng trong quá trình tiền xử lý văn bản để đảm bảo tính nhất quán và đơn giản hóa việc phân tích.

Raw	Lowercased
Canada CanadA CANADA	canada
TOMCAT Tomcat toMcat	tomcat

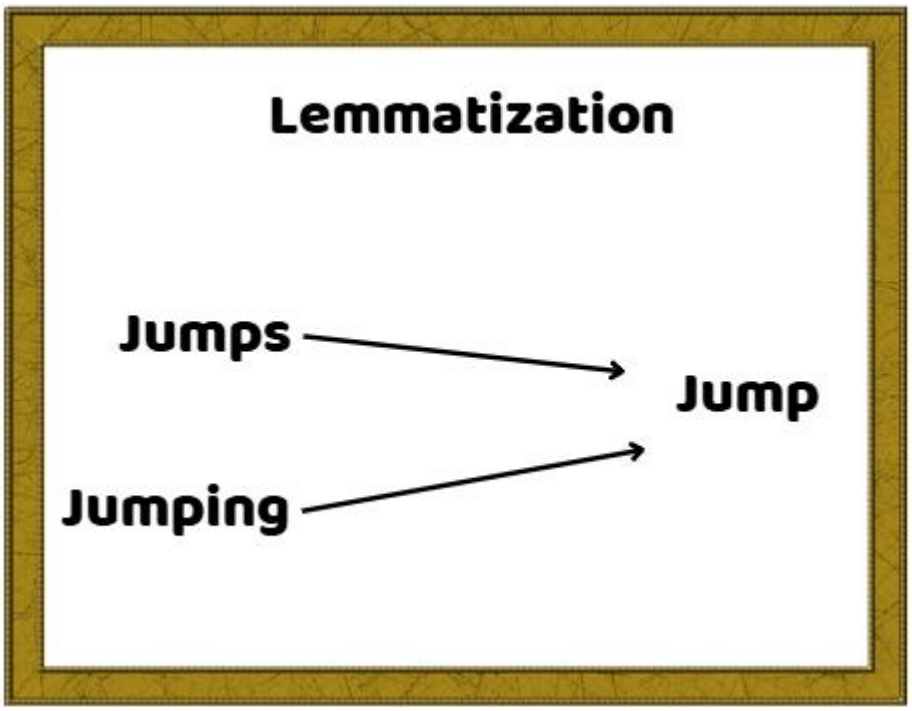
Hình 3 – Hình minh họa Converting text to lowercase

Việc sử dụng Lowercase giúp cho văn bản được đồng nhất về một kiểu là viết thường, từ đó giúp tăng hiệu quả xử lý văn bản trong việc tìm kiếm hay phân loại.[4]

Lemmatization

Lemmatization là một kỹ thuật tiền xử lý văn bản, giúp chuyển đổi các từ về dạng cơ sở hoặc dạng từ điển của chúng, gọi là lemmas. Khác với stemming,

có thể tạo ra các từ không đầy đủ hoặc bị rút gọn, lemmatization duy trì độ chính xác ngữ nghĩa bằng cách xem xét bối cảnh ngữ pháp và loại từ. Ví dụ, cả "jumps" và "jumping" sẽ được chuyển thành "jump", bảo toàn ý nghĩa của chúng.[4]



Hình 4 – Hình minh họa Lemmatization

Train Test Split

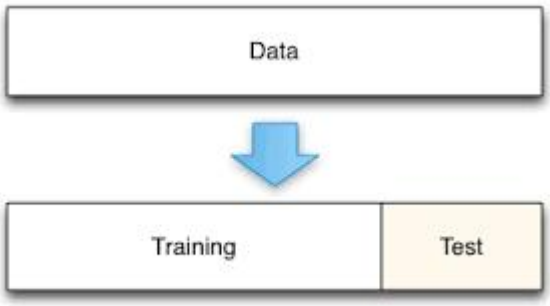
Sử dụng để chia dữ liệu thành hai tập: tập huấn luyện (train set) và tập kiểm tra (test set).

Chúng ta cần chia dữ liệu thành các đặc trưng (X) và nhãn (y). Dữ liệu sẽ được chia thành bốn phần: X_train, X_test, y_train và y_test.

- X_train và y_train được sử dụng để huấn luyện và tối ưu hóa mô hình.
- X_test và y_test được sử dụng để kiểm tra xem mô hình có dự đoán đúng nhãn (labels) hay không.

Train set là một tập dữ liệu được sử dụng để huấn luyện mô hình. Đây là tập dữ liệu mà mô hình được "nhìn thấy" và học hỏi trong quá trình đào tạo.

Test set là một tập hợp con riêng biệt được sử dụng để đánh giá chính xác mức độ phù hợp của mô hình cuối cùng.

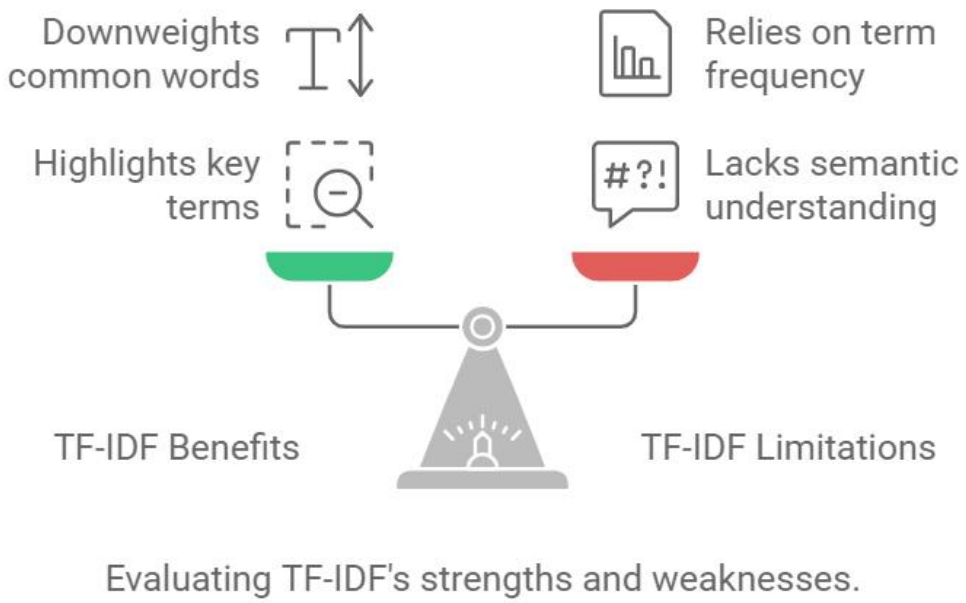


Hình 5 – Hình minh họa Train Test Split

Term Frequency-Inverse Document Frequency

Là một kỹ thuật biểu diễn văn bản được sử dụng để chuyển đổi dữ liệu văn bản thành vectơ số phục vụ cho các mô hình học máy. TF-IDF giúp xác định mức độ quan trọng của các từ trong một tài liệu so với toàn bộ tập dữ liệu.

- **TF**: Đo lường mức độ thường xuyên của một từ xuất hiện trong một tài liệu. TF được tính bằng tỉ lệ giữa số lần xuất hiện của từ và tổng số từ trong tài liệu. Giá trị TF càng cao, từ đó càng quan trọng trong tài liệu.
- **IDF**: Đo lường mức độ quan trọng của một từ trong toàn bộ tập dữ liệu (corpus). IDF làm giảm trọng số của các từ phổ biến (ví dụ: "và", "là") và tăng trọng số của các từ hiếm.
- **TF-IDF Score:** Là tích của TF và IDF, biểu thị mức độ liên quan của một từ trong một tài liệu so với toàn bộ tập dữ liệu. Các từ có tần suất cao trong một tài liệu nhưng hiếm trong toàn bộ tập dữ liệu sẽ có điểm TF-IDF cao.



Hình 6 – Hình minh họa TF-IDF

TF-IDF giúp giảm trọng số các từ phổ biến nhưng không mang nhiều ý nghĩa, đồng thời làm nổi bật các từ đặc trưng trong tài liệu, phù hợp cho các bài toán như phân loại văn bản, phân tích ngữ nghĩa, và trích xuất từ khóa.[4]

2.3 Mô hình xử lý

2.3.1 Mô hình Naive Bayes

a. Định nghĩa

Naive Bayes là một thuật toán phân loại đơn giản nhưng hiệu quả, dựa trên **định lý Bayes**. Định lý Bayes cho phép chúng ta tính toán xác suất của một sự kiện dựa trên các bằng chứng đã biết. Trong Naive Bayes, giả định quan trọng nhất là **các đặc trưng đầu vào là độc lập có điều kiện** khi biết lớp mục tiêu, mặc dù giả định này không luôn đúng trong thực tế. Sự đơn giản hóa này giúp việc tính toán xác suất trở nên dễ dàng hơn. Công thức định lý Bayes:

$$P(C|x) = \frac{P(x|C).P(C)}{P(x)} \quad (1)$$

Trong đó:

- P(C|x): Xác suất thuộc lớp C khi đã biết các đặc trưng x.
- P(x|C): Xác suất có các đặc trưng x khi thuộc lớp C.
- P(C): Xác suất của lớp C (prior probability).
- P(x): Xác suất của các đặc trưng x (evidence).

Các biến thể:

- Gaussian Naive Bayes: Sử dụng khi các đặc trưng là liên tục.
- Multinomial Naive Bayes: Phù hợp với dữ liệu rời rạc, ví dụ như tần suất từ trong văn bản.
- Bernoulli Naive Bayes: Sử dụng với dữ liệu nhị phân, như sự xuất hiện hay không xuất hiện của một từ.[5]

b. Naive Bayes hoạt động như thế nào?

Giả sử chúng ta muốn phân loại email là spam hay không spam dựa trên các từ xuất hiện trong email. Nếu từ "khuyến mãi" xuất hiện nhiều, xác suất để email đó là spam sẽ cao hơn. Cụ thể: Có 10 email, 5 email là spam và 5 email là ham. Có 6 email chứa từ “khuyến mãi” gồm 5 email spam và 1 email ham.

Vậy ta có các xác xuất:

P(Spam) = 0.5 (50% email là spam).

P(Ham) = 0.5.

P(“khuyến mãi) = 0.6

P("khuyến mãi" | Spam) = 1.

P("khuyến mãi" | Ham) = 0.2.

Sử dụng công thức, chúng ta có thể tính toán và đưa ra dự đoán dựa trên xác suất.

P(Spam|"khuyến mãi") = 0.83.

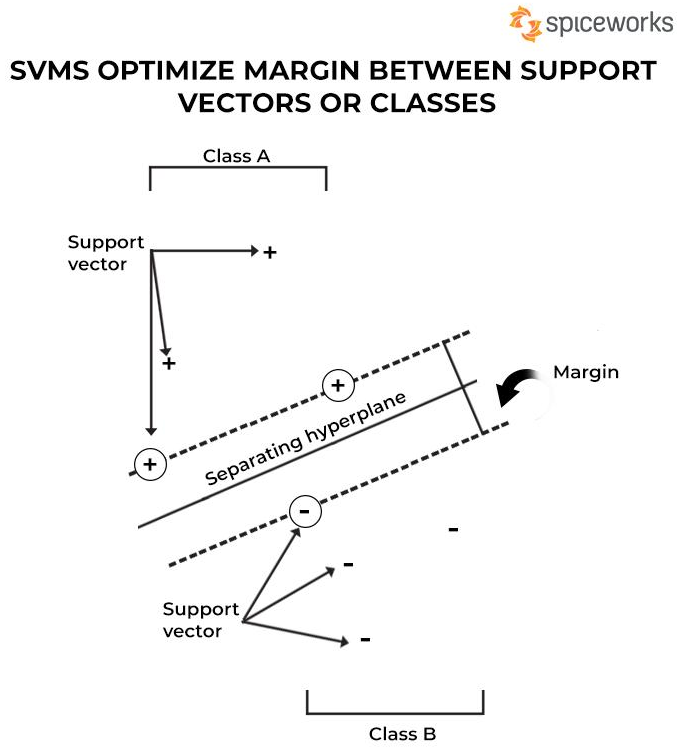
P(Ham|"khuyến mãi") = 0.17.

Vì xác xuất của một email spam có từ “khuyến mãi” là 83% và lớn hơn một email ham có từ “khuyến mãi” là 17%. Nên ở các dự đoán tiếp theo khi xuất hiện từ “khuyến mãi” thì mô hình của chủng ta sẽ dự đoán đó là một email spam.

2.3.2 Mô hình Support Vector Machine

a. Định nghĩa

Support Vector Machine (SVM) là một thuật toán học máy sử dụng các mô hình học có giám sát để giải quyết các bài toán phức tạp như phân loại, hồi quy và phát hiện ngoại lệ. SVM tìm cách xây dựng một **mặt siêu phẳng (hyperplane)** tối ưu để phân tách các lớp dữ liệu trong không gian đặc trưng. Mục tiêu chính của SVM là **tối đa hóa khoảng cách (margin)** giữa siêu phẳng và các điểm dữ liệu gần nhất của mỗi lớp, được gọi là **vector hỗ trợ (support vectors)**.



Hình 7 – Hình minh họa mô tả khái niệm SVM

Như được thấy trong hình 7, khoảng cách (margin) là chiều rộng tối đa của phần không gian song song với siêu phẳng mà không có vector hỗ trợ nào nằm bên trong. Những siêu phẳng như vậy dễ xác định hơn với các bài toán tuyến tính. Công thức phân loại tuyến tính như sau:

$$f = f(x, w, b) = sign(w . x + b) \quad (2)$$

▪ Nếu f = + 1, x thuộc Lớp 1.

▪ Nếu f = − 1, x thuộc Lớp 2.

▪ Ta gọi f là một bộ phân loại tuyến tính vì w.x + b = 0 là một đường thẳng (mặt siêu phẳng).

Tuy nhiên, trong các vấn đề hoặc tình hình thực tế, thuật toán SVM cố gắng tối ưu hóa khoảng cách giữa các vector hỗ trợ, dẫn đến khả năng phân loại sai đối với một số điểm dữ liệu nhỏ.

SVM được thiết kế chủ yếu cho các bài toán phân loại nhị phân. Tuy nhiên, với sự gia tăng các vấn đề phân loại đa lớp phức tạp đòi hỏi tính toán cao, người ta xây dựng và kết hợp nhiều bộ phân loại nhị phân để tạo thành SVM có khả năng thực hiện phân loại đa lớp thông qua cách tiếp cận nhị phân.[6-8]

Đặc điểm nổi bật của SVM:

- Hỗ trợ bài toán phi tuyến tính: Khi dữ liệu không thể tách biệt tuyến tính, SVM sử dụng các hàm hạt nhân (kernel functions) để ánh xạ dữ liệu sang không gian nhiều chiều hơn, nơi dữ liệu có thể phân tách được. Một số hàm hạt nhân phổ biến: Hàm hạt nhân tuyến tính, Gaussian RBF, Hàm hạt nhân bậc đa thức.
- Linh hoạt: SVM có thể áp dụng cho cả bài toán phân loại nhị phân và phân loại đa lớp (thông qua phương pháp one-vs-one hoặc one-vs-rest).
- Khả năng khái quát hóa tốt: Nhờ tối ưu hóa khoảng cách, SVM thường tránh được hiện tượng quá khớp (overfitting) trên dữ liệu huấn luyện.

b. SVM hoạt động như thế nào?

Để hiểu rõ cách SVM hoạt động, hãy xem xét một ví dụ đơn giản với dữ liệu tuyến tính.

Phân loại dữ liệu tuyến tính:

Giả sử chúng ta có hai nhóm dữ liệu với các đặc trưng x và y, được biểu diễn bằng màu **đỏ** và **đen**. SVM hoạt động bằng cách tìm một siêu phẳng (hyperplane) phân tách dữ liệu thành hai nhóm sao cho:

- **Khoảng cách (margin)** giữa siêu phẳng và các điểm dữ liệu gần nhất của hai nhóm là lớn nhất.
- **Vector hỗ trợ (support vectors)** là các điểm dữ liệu gần siêu phẳng nhất, quyết định vị trí của siêu phẳng.

Phân loại dữ liệu phi tuyến tính:

Khi dữ liệu không thể tách biệt tuyến tính, SVM sử dụng **hàm hạt nhân (kernel functions)** để ánh xạ dữ liệu sang không gian nhiều chiều, nơi dữ liệu có thể phân tách dễ dàng hơn. Ví dụ:

- Với dữ liệu hình tròn trong không gian 2D, hàm hạt nhân Gaussian RBF có thể ánh xạ dữ liệu vào không gian 3D, nơi các điểm nằm trên mặt cầu và dễ phân tách.

2.3.3 Mô hình Logistic Regression

a. Định nghĩa

Là một thuật toán học máy có giám sát thực hiện các nhiệm vụ phân loại nhị phân bằng cách dự đoán xác suất của một kết quả, sự kiện hoặc quan sát. Mô hình này đưa ra một kết quả nhị phân hoặc phân đôi, giới hạn trong hai kết quả có thể xảy ra: 0 và 1.

Logistic Regression phân tích mối quan hệ giữa một hoặc nhiều biến độc lập và phân loại dữ liệu vào các lớp rời rạc. Nó được sử dụng rộng rãi trong mô hình dự đoán, trong đó mô hình ước tính xác suất toán học liệu một đối tượng có thuộc về một danh mục cụ thể hay không.

Công thức Linear Regression:

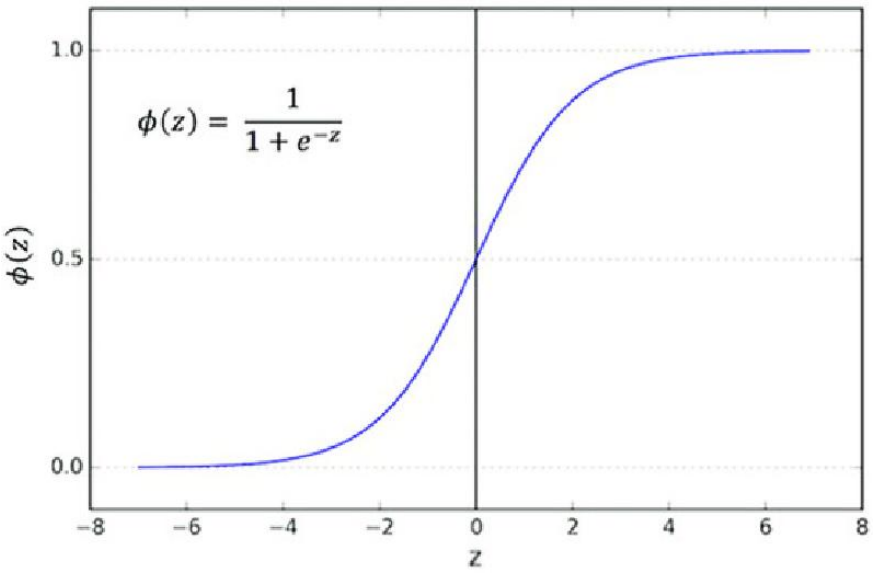
$$f(x) = w^T x \tag{3}$$

Đầu ra của Logistic regression thường được viết chung dưới dạng:

$$f(x) = \Theta(w^T x) \tag{4}$$

Θ(.) được gọi là một hàm kích hoạt.

Và để đầu ra là một biến độc lập nằm trong khoảng [0,1]. Ta cần sử dụng sigmoid function.



Hình 8 – Hình minh họa mô tả sigmoid function

Công thức sigmoid function:

$$\emptyset(z) = \frac{1}{1 + e^{-z}} \tag{5}$$

∅(z): đầu ra của hàm (nằm trong khoảng từ 0 đến 1)

Z: đầu vào của hàm

e: số tự nhiên logarit

b. Logistic Regression hoạt động như thế nào?

Ví dụ: 0 – đại diện cho lớp tiêu cực; 1 – đại diện cho lớp tích cực.

$P(y=1|x) = \emptyset(z)$

Xem P là xác xuất của output y =1 với giá trị x mà ta cung cấp.

Áp dụng một ngưỡng ranh giới = 0.5. Khi đó:

- P >= 0.5, thì thuộc lớp tích cực.
- P < 0.5 thì thuộc lớp tiêu cực.

Một số ví dụ về các phân loại và trường hợp mà phản hồi nhị phân được mong đợi hoặc ngụ ý là: xác định xác suất bị nhồi máu cơ tim, khả năng nhập học vào một trường đại học, nhận diện thư rác.[6-8]

Các loại Logistic Regression:

Logistic Regression nhị phân:

- Hồi quy logistic nhị phân dự đoán mối quan hệ giữa các biến phụ thuộc nhị phân và độc lập. Một số ví dụ về đầu ra của loại hồi quy này có thể là thành công/thất bại, 0/1 hoặc đúng/sai.

Logistic Regression đa thức:

- Biến phụ thuộc phân loại có hai hoặc nhiều kết quả rời rạc trong loại hồi quy đa thức. Điều này ngụ ý rằng loại hồi quy này có nhiều hơn hai kết quả có thể xảy ra.

Logistic Regression thứ tự:

- Hồi quy logistic thứ tự áp dụng khi biến phụ thuộc ở trạng thái có thứ tự (tức là thứ tự). Biến phụ thuộc (y) chỉ định thứ tự với hai hoặc nhiều loại hoặc cấp độ.

CHƯƠNG 3: CÁC BƯỚC CÀI ĐẶT

Cài đặt thư viện

a. Thư viện pandas

Thư viện Pandas trong Python thường được sử dụng để: xử lý dữ liệu, phân tích dữ liệu, chuyển đổi dữ liệu, thao tác dữ liệu dạng bảng.

Để thực hiện cài cho thư viện này, có thể nhập câu lệnh như sau trong cmd:

```
pip install pandas
```

b. Thư viện sklearn

Thư viện sklearn trong Python thường được sử dụng để: học máy, xử lý dữ liệu, đánh giá mô hình, pipeline.

Để thực hiện cài cho thư viện này, có thể nhập câu lệnh như sau trong cmd:

```
pip install scikit-learn
```

c. Thư viện nltk

Thư viện nltk trong Python thường được sử dụng để: tiền xử lý văn bản, phân tích ngôn ngữ, tính toán văn bản, xây dựng mô hình NLP.

Để thực hiện cài cho thư viện này, có thể nhập câu lệnh như sau trong cmd:

```
pip install nltk
```

d. Thư viện matplotlib

Thư viện matplotlib trong Python thường được sử dụng để: vẽ biểu đồ, trực quan hóa dữ liệu, tùy chỉnh đồ thị.

Để thực hiện cài cho thư viện này, có thể nhập câu lệnh như sau trong cmd:

```
pip install matplotlib
```

CHƯƠNG 4: THỰC NGHIỆM VÀ PHÂN TÍCH KẾT QUẢ

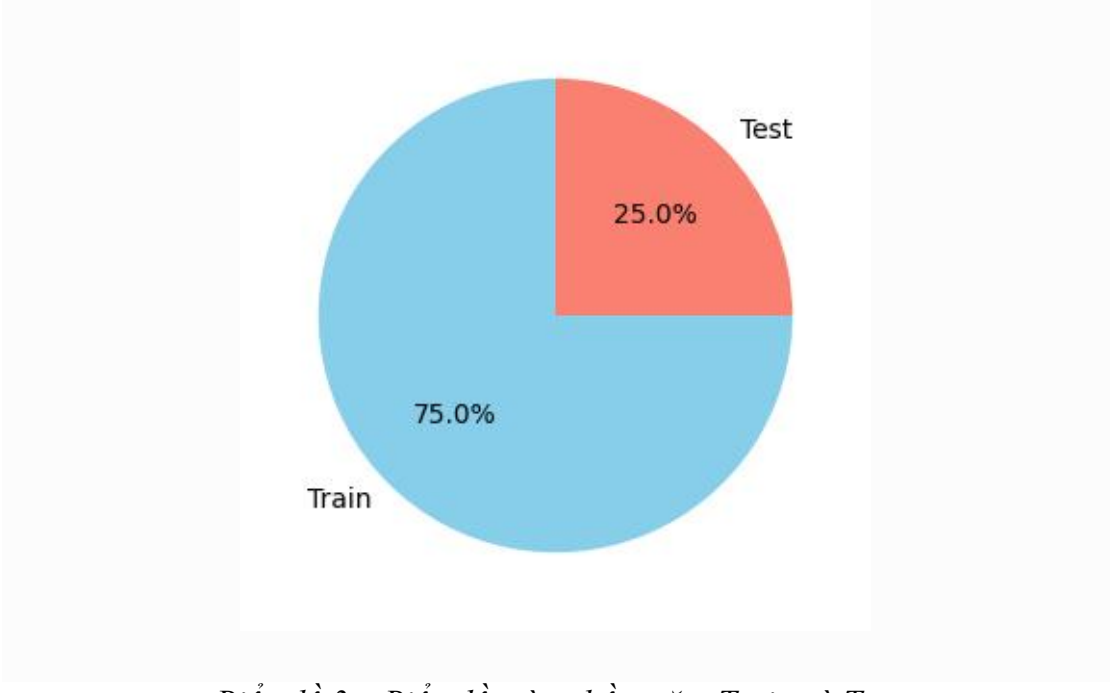
4.1 Tiền xử lý dữ liệu

Thực hiện xóa dấu câu, chuyển thành chữ thường, tách từ, loại bỏ stopwords và lemmatize.

Bảng 1 – Bảng so sánh trước và sau khi thực hiện tiền xử lý

Trước	Sau
The guy at the car shop who was flirting with me got my phone number from the paperwork and called and texted me. I'm nervous because of course now he may have my address. Should i call his boss and tell him, knowing this may get him fired	guy car shop flirting got phone number paperwork called texted im nervous course may address call bos tell knowing may get fired

Thực hiện chia dữ liệu:



Biểu đồ 3 – Biểu đồ tròn phần trăm Train và Test

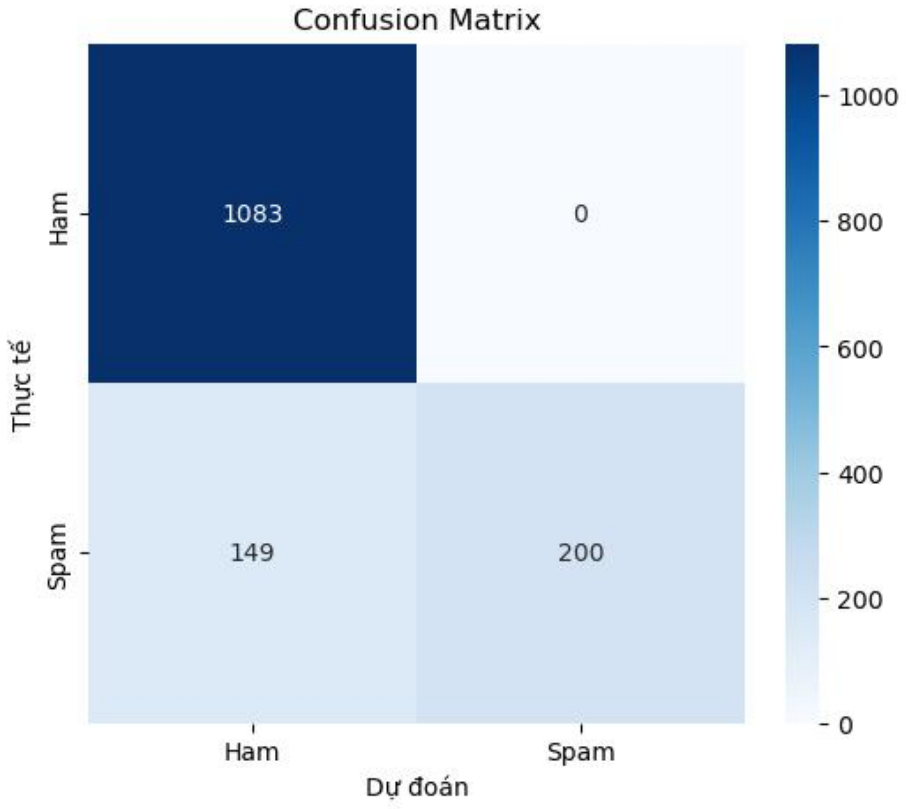
Tổng dữ liệu được chia ra làm hai phần là Train và Test. Phần Train chiếm 75% tương ứng với 4296 dòng dữ liệu. Phần Test chiếm 25% tương ứng với 1432 dòng dữ liệu.

4.2 Áp dụng mô hình xử lý

4.2.1 Sử dụng mô hình Naive Bayes với biến thể Multinomial Naive Bayes

Bảng 2 – Bảng báo cáo phân loại Naive Bayes

	Precision	Recall	F1-score	Support
0 (Ham)	0.88	1.00	0.94	1083
1 (Spam)	1.00	0.57	0.73	349
Accuracy			0.9	1432
Macro avg	0.94	0.79	0.83	1432

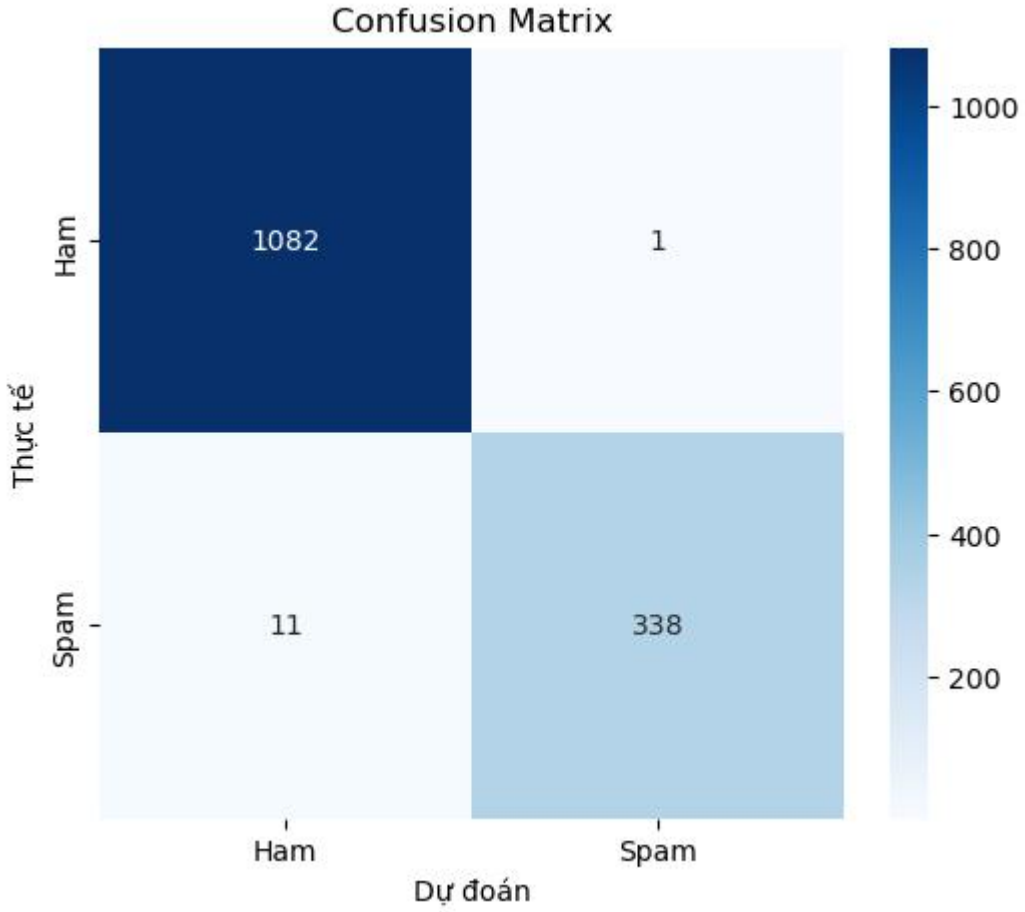


Biểu đồ 4 – Biểu đồ ma trận nhầm lẫn của mô hình Naive Bayes

4.2.2 Sử dụng mô hình Support Vector Machine

Bảng 3 – Bảng báo cáo phân loại SVM

	Precision	Recall	F1-score	Support
0 (Ham)	0.99	1.00	0.99	1083
1 (Spam)	1.00	0.97	0.98	349
Accuracy			0.99	1432
Macro avg	0.99	0.98	0.99	1432

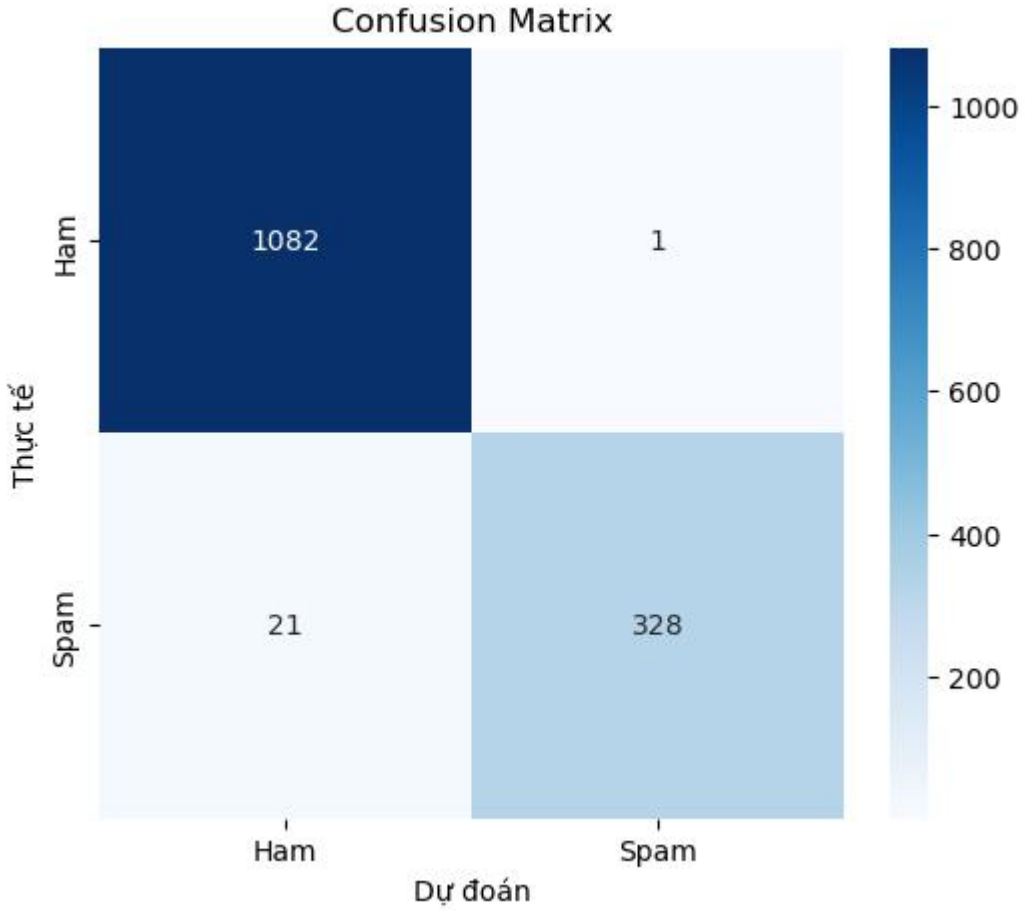


Biểu đồ 5 – Biểu đồ ma trận nhầm lẫn của mô hình SVM

4.2.3 Sử dụng mô hình Logistic Regression

Bảng 4 – Bảng báo cáo phân loại Logistic Regression

	Precision	Recall	F1-score	Support
0 (Ham)	0.98	1.00	0.99	1083
1 (Spam)	1.00	0.94	0.97	349
Accuracy			0.98	1432
Macro avg	0.99	0.97	0.98	1432



Biểu đồ 6 – Biểu đồ ma trận nhầm lẫn của mô hình Logistic Regression

4.3 Phân tích kết quả

Bảng 5 – Bảng so sánh kết quả theo từng lớp

Mô hình	Phân loại	Precision	Recall	F1-score
Naive Bayes	0 (Ham)	0.88	1.00	0.94
	1 (Spam)	1.00	0.57	0.73
SVM	0 (Ham)	0.99	1.00	0.99
	1 (Spam)	1.00	0.97	0.98
Logistic Regression	0 (Ham)	0.98	1.00	0.99
	1 (Spam)	1.00	0.94	0.97

Bảng 6 – Bảng so sánh kết quả tổng quan

Mô hình	Accuraccy	Macro Avg		
		Precision	**Recall**	**F1-score**
Naive Bayes	0.9	0.94	0.79	0.83
SVM	0.99	0.99	0.98	0.99
Logistic Regression	0.98	0.99	0.97	0.98

Dựa trên so sánh ở bảng 5 và bảng 6, ta có các đánh giá sau:

Naive Bayes

- **Đối với lớp 0 (Ham):**
  - Precision: 0.88 cho thấy 88% các tin nhắn được dự đoán là Ham là chính xác.
  - Recall: 1.00 tất cả các tin nhắn Ham thực sự đều được nhận diện chính xác.

- F1-score: 0.94 phản ánh sự cân bằng tốt giữa Precision và Recall.

- **Đối với lớp 1 (Spam):**

  - Precision: 1.00, nghĩa là tất cả các tin nhắn được dự đoán là Spam đều đúng.
  - Recall: 0.57, chỉ ra rằng 57% các tin nhắn Spam thực sự được nhận diện.
  - F1-score: 0.73, thấp hơn lớp 0, cho thấy mô hình gặp khó khăn hơn với việc nhận diện Spam.

- **Macro average**:
  - Precision: 0.94, cho thấy mô hình dự đoán chính xác cao ở cả hai lớp (Ham và Spam).
  - Recall: 0.79, thấp hơn so với Precision, nghĩa là mô hình bỏ sót nhiều tin nhắn Spam.
  - F1-score: 0.83, phản ánh hiệu quả tổng thể của mô hình, nhưng vẫn kém hơn SVM.

SVM

- **Đối với lớp 0 (Ham):**

  - Precision: 0.99, rất cao, nghĩa là gần như tất cả các dự đoán Ham đều đúng.
  - Recall: 1.00, tất cả các tin nhắn Ham thực sự đều được phát hiện.
  - F1-score: 0.99, cao hơn Naive Baye.

- **Đối với lớp 1 (Spam):**

  - Precision: 1.00, chính xác tuyệt đối.
  - Recall: 0.97, cải thiện đáng kể so với Naive Bayes.
  - F1-score: 0.98, cao nhất cho lớp Spam trong ba mô hình.

- **Macro average**:
    - Precision: 0.99, rất cao.
    - Recall: 0.98, cao nhất, cho thấy SVM nhận diện cả hai lớp tốt hơn so với các mô hình khác.
    - F1-score: 0.99, vượt trội trong việc cân bằng Precision và Recall.

Logistic Regression

- **Đối với lớp 0 (Ham):**
    - Precision: 0.98, cao hơn Naive Bayes nhưng thấp hơn SVM.
    - Recall: 1.00, tương tự hai mô hình kia.
    - F1-score: 0.99, ngang với SVM.
- **Đối với lớp 1 (Spam):**
    - Precision: 1.00, rất cao.
    - Recall: 0.94, cao hơn Naive Bayes nhưng thấp hơn SVM.
    - F1-score: 0.97, cải thiện so với Naive Bayes nhưng thấp hơn SVM.
- **Macro average**:
    - Precision: 0.99, rất cao.
    - Recall: 0.97, cao hơn Naive Bayes nhưng thấp hơn SVM.
    - F1-score: 0.98, cao hơn Naive Bayes.

Bảng 7 – Bảng so sánh kết quả thu được từ các mô hình

Mô hình	Số lượng Ham dự đoán đúng	Số lượng Spam dự đoán đúng	Số lượng Ham dự đoán sai thành Spam	Số lượng Spam dự đoán sai thành Ham
Naive Bayes	1083	200	0	149
SVM	1082	338	1	11
Logistic Regression	1082	328	1	21

Thông qua bảng 7 ta có thể thấy mô hình SVM có số lượng dự đoán Ham, Spam đúng cao nhất và sai ít nhất.

Mô hình SVM là lựa chọn tốt nhất do đạt độ chính xác cao nhất thông qua các chỉ số đã so sánh ở bảng 5, 6, 7 và số lượng dự đoán đúng, sai của Ham và Spam.

Còn hai mô hình là Naive Bayes và Logistic Regression có độ các thông số tương đồng nhau nhưng vẫn kém hơn mô hình SVM.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 Kết luận

Thông qua đề tài này đã chứng minh hiệu quả của các mô hình NLP hiện đại trong việc phân loại email rác. Mô hình được xây dựng không chỉ đạt được độ chính xác cao trong việc phân loại các loại email rác khác nhau mà còn có khả năng thích ứng với các dạng spam mới xuất hiện. Kết quả này mở ra nhiều tiềm năng ứng dụng trong thực tế, giúp người dùng tiết kiệm thời gian, bảo vệ thông tin cá nhân và nâng cao hiệu quả làm việc. Trong tương lai, việc kết hợp các mô hình NLP với các kỹ thuật học sâu tiên tiến hứa hẹn sẽ mang đến những đột phá mới trong việc chống lại spam và bảo mật thông tin.

5.2 Hướng phát triển

Trong tương lai chúng ta cần phải thu thập thêm các dữ liệu như: tên người gửi, tên người nhận, tiêu đề, tệp đính kèm... Từ đó chúng ta có thể đa dạng hóa dữ liệu song song với đó ta có thể tăng độ chính xác của mô hình từ việc có thêm đối tượng để chọn lọc.

Phát triển mô hình với đa ngôn ngữ, đặc biệt là đối với tiếng Việt của chúng ta.

Kết hợp với các kỹ thuật bảo mật khác, phát hiện các kiểu spam mới và phức tạp hơn. Đặc biệt là tối ưu hóa đối với dữ liệu lớn.

TÀI LIỆU THAM KHẢO

[1] T. Hoa. "Thư rác (Spam) là gì? Cách chặn thư rác trong Gmail." VietNamBiz. https://vietnambiz.vn/thu-rac-spam-la-gi-cach-chan-thu-rac-trong-gmail-20190909143353946.htm (accessed 20/11, 2024).

[2] D. N. T. Thùy. "Natural Language Processing (NLP) là gì và nó có ứng dụng như thế nào?" Viblo. https://viblo.asia/p/natural-language-processing-nlp-la-gi-va-no-co-ung-dung-nhu-the-nao-GAWVpMA3405?fbclid=IwZXh0bgNhZW0CMTEAAR0h-a7RE1N9tYSRc5kICoPfdt8UA1smMRLUFMW8WdRF1taAWqTLXfCVP0Y_aem_0DKDAqg_MApff6qLASW-yA (accessed 20/11, 2024).

[3] V. Pham. "Ứng dụng NLP trong việc lọc thư rác, spam email – Xử lý ngôn ngữ tự nhiên." https://clickdigital.website/ung-dung-nlp-trong-viec-loc-spam-email/ (accessed 20/11, 2024).

[4] J. P. Jiawei Han , Hanghang Tong, *Data Mining: Concepts and Techniques* fourth edition ed. Morgan Kaufmann (in English), 17/10/2022, p. 752.

[5] S. G. Andreas C. Müller, *Introduction to Machine Learning with Python*. O'Reilly Media (in English), 25/9/2016, p. 389.

[6] T. T. Đạt. "Machine Learning." https://sites.google.com/site/ttdat88/courses/machine-learning-masters-course?authuser=0 (accessed 01/12, 2024).

[7] T. H. Julian Avila *scikit-learn Cookbook*, Second Edition ed. Packt Publishing (in English), 15/11/2017, p. 374.

[8] M. Swamynathan, *Mastering Machine Learning with Python in Six Steps*. Apress (in English), 05/06/2017, p. 379.

PHỤ LỤC

```
# Import thư viện
import pandas as pd
import string
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import
TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import (
    accuracy_score,
    classification_report,
    confusion_matrix
)
import matplotlib.pyplot as plt
# Tiền xử lý dữ liệu
lemmatizer = WordNetLemmatizer()
STOPWORDS = stopwords.words("english")

def text_process(mess):
    # Xóa dấu câu
    nopunc = [char for char in mess if char not in
string.punctuation]
    nopunc = "".join(nopunc)
```

```
    # Chuyển thành chữ thường, tách từ, loại bỏ stopwords và lemmatize
    words = [
        lemmatizer.lemmatize(word.lower())
        for word in nopunc.split()
        if word.lower() not in STOPWORDS
    ]
    return " ".join(words)
sms["clean_msg"] = sms.message.apply(text_process)
```

```
# Train Test Split

X = sms.clean_msg
y = sms.label_num
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=1)

tfidf_vectorizer = TfidfVectorizer()
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)
# Mô hình xử lý

# Naive Bayes
naive_bayes_model = MultinomialNB()
naive_bayes_model.fit(X_train_tfidf, y_train)  # Huấn luyện mô hình
y_pred_naive_bayes =
naive_bayes_model.predict(X_test_tfidf)
```

```
# Hiển thị thông số
print("Model: Naive Bayes")
print("Accuracy:", accuracy_score(y_test,
y_pred_naive_bayes))
print("\nClassification Report:\n",
classification_report(y_test, y_pred_naive_bayes))
print("\nConfusion Matrix:\n", confusion_matrix(y_test,
y_pred_naive_bayes))
```

```
# SVM
svm_model = SVC(probability=True)
svm_model.fit(X_train_tfidf, y_train)  # Huấn luyện mô
hình
y_pred_svm = svm_model.predict(X_test_tfidf)

# Hiển thị thông số
print("Model: SVM")
print("Accuracy:", accuracy_score(y_test, y_pred_svm))
print("\nClassification Report:\n",
classification_report(y_test, y_pred_svm))
print("\nConfusion Matrix:\n", confusion_matrix(y_test,
y_pred_svm))

# Logistic Regression
logistic_model = LogisticRegression()
logistic_model.fit(X_train_tfidf, y_train)
y_pred_logistic = logistic_model.predict(X_test_tfidf)
```

```
# Hiển thị thông số
print("Model: Logistic Regression")
print("Accuracy:", accuracy_score(y_test, y_pred_logistic))
print("\nClassification Report:\n",
classification_report(y_test, y_pred_logistic))
print("\nConfusion Matrix:\n", confusion_matrix(y_test,
y_pred_logistic))
```