

BỘ CÔNG THƯƠNG

TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP. HCM

KHOA CÔNG NGHỆ THÔNG TIN



ĐỒ ÁN CHUYÊN NGÀNH

NGHIÊN CỨU VÀ ỨNG DỤNG THUẬT TOÁN RANDOM FOREST ĐỂ PHÁT HIỆN EMAIL LỪA ĐẢO (PHISHING EMAIL)

GIẢNG VIÊN HƯỚNG DẪN

Nguyễn Thị Hồng Thảo

SINH VIÊN THỰC HIỆN

2033216597 – Phạm Minh Tú – 12DHBM09

2033216424 – Mai Quốc Hùng – 12DHBM03

2033210022 – Đinh Lê Gia Bảo – 12DHBM01

TP. HỒ CHÍ MINH, tháng...năm 20...

BỘ CÔNG THƯƠNG

TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP. HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN



ĐỒ ÁN CHUYÊN NGÀNH

NGHIÊN CỨU VÀ ỨNG DỤNG THUẬT TOÁN RANDOM FOREST ĐỂ PHÁT HIỆN EMAIL LỪA ĐẢO (PHISHING EMAIL)

GIẢNG VIÊN HƯỚNG DẪN

Nguyễn Thị Hồng Thảo

SINH VIÊN THỰC HIỆN

2033216597 – Phạm Minh Tú – 12DHBM09

2033216424 – Mai Quốc Hùng – 12DHBM03

2033210022 – Đinh Lê Gia Bảo – 12DHBM01

TP. HỒ CHÍ MINH, tháng...năm 20...

LỜI CẢM ƠN

Lời đầu tiên, chúng em xin được gửi lời cảm ơn chân thành nhất đến cô Nguyễn Thị Hồng Thảo. Trong quá trình học tập và tìm hiểu môn Đồ án chuyên ngành, chúng em đã nhận được rất nhiều sự quan tâm, giúp đỡ, hướng dẫn tâm huyết và tận tình từ cô. Cô đã giúp chúng em tích lũy thêm nhiều kiến thức về môn học này để có thể hoàn thành được bài tiểu luận về đề tài tìm hiểu về thuật toán Random Forest, Machine Learning.

Trong quá trình làm bài chắc chắn khó tránh khỏi những thiếu sót. Do đó, chúng em kính mong nhận được những lời góp ý của cô để bài tiểu luận của chúng em ngày càng hoàn thiện hơn.

Chúng em xin chân thành cảm ơn!

LỜI CAM ĐOAN

Nhóm chúng em xin cam đoan đây là công trình nghiên cứu của riêng chúng em và được sự hướng dẫn khoa học của cô Nguyễn Thị Hồng Thảo. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong Khóa luận Đồ án chuyên ngành còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kỳ sự gian lận nào chúng em xin hoàn toàn chịu trách nhiệm về nội dung Khóa luận Đồ án chuyên ngành của mình. Trường Đại học Công Thương Thành phố Hồ Chí Minh không liên quan đến những vi phạm tác quyền, bản quyền do gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, tháng...năm 20...

Sinh viên thực hiện Đồ án

(Ký và ghi rõ họ tên)

This image shows a full page of white paper with horizontal dotted lines, typical of primary school writing paper. The lines are evenly spaced and run across the entire width of the page. There are no margins, text, or other markings on the paper.

TP. Hồ Chí Minh, tháng...năm 20...

(Ký và ghi rõ họ tên)

This image shows a full page of white paper with horizontal dotted lines, typical of primary school writing paper. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

(Ký và ghi rõ họ tên)

BẢNG PHÂN CÔNG CÔNG VIỆC

Họ và tên	Nội dung công việc	Mức độ đóng góp đề tài (%)
Đinh Lê Gia Bảo	1, 3, 6, 7, 8	30%
Phạm Minh Tú	1, 2, 3, 4, 7, 8	30%
Mai Quốc Hùng	1, 2, 4, 6, 7, 8	40%

Nội dung chi tiết công việc

STT	Nội dung công việc
1	– Tìm hiểu cơ bản về khái niệm học máy và mô hình Random Forest
2	– Thu thập dữ liệu: các tập dữ liệu phishing email từ Kaggle, SpamAssassin, hoặc Enron corpus – Phân tích đặc điểm của email lừa đảo (từ khóa, link, người gửi...)
3	- Tiền xử lý dữ liệu: loại bỏ stop words, stemming, chuẩn hóa văn bản, vector hóa (TF-IDF hoặc Bag-of-Words) – Gán nhãn và chia tập train/test
4	– Triển khai thuật toán Random Forest. – Huấn luyện mô hình với dữ liệu đã xử lý
5	– Triển khai thuật toán Random Forest. – Huấn luyện mô hình với dữ liệu đã xử lý
6	– Đánh giá thực nghiệm
7	– Tiếp tục thực hiện, viết báo cáo, báo cáo với giảng viên
8	– Hoàn chỉnh báo cáo. – Thiết kế bài báo cáo trên PowerPoint. – Nộp báo cáo.

TÓM TẮT

Đề tài nhóm chúng em thực hiện tập trung vào lĩnh vực an toàn thông tin, cụ thể là trong việc phát hiện và ngăn chặn email lừa đảo (phishing email) – một trong những mối đe dọa phổ biến và nguy hiểm trong môi trường mạng hiện nay. Mục tiêu chính của nghiên cứu là tìm hiểu chuyên sâu về thuật toán Random Forest – một kỹ thuật học máy mạnh mẽ, thường được ứng dụng trong các bài toán phân loại – để xây dựng một mô hình có khả năng phân biệt giữa email hợp lệ và email lừa đảo.

Trong quá trình thực hiện, nhóm đã tiến hành thu thập và xử lý tập dữ liệu gồm nhiều đặc trưng liên quan đến nội dung và tiêu đề email, sau đó áp dụng các kỹ thuật tiền xử lý dữ liệu, trích xuất đặc trưng và huấn luyện mô hình Random Forest. Nhờ vào khả năng xử lý tốt dữ liệu có nhiều chiều và tránh overfitting, Random Forest được lựa chọn như một giải pháp hiệu quả để phát hiện các hành vi lừa đảo trong email.

Ngoài ra, nhóm cũng so sánh hiệu suất của mô hình Random Forest với một số thuật toán học máy khác như Naive Bayes và Decision Tree để đánh giá mức độ chính xác, độ nhạy và độ đặc hiệu trong việc phát hiện email lừa đảo. Mục tiêu cuối cùng của đề tài là triển khai một công cụ phát hiện email phishing dựa trên mô hình học máy, góp phần nâng cao khả năng bảo vệ người dùng khỏi các cuộc tấn công lừa đảo qua email trong thực tế.

MỤC LỤC

LỜI CẢM ƠN.....	1
LỜI CAM ĐOAN	2
BẢNG PHÂN CÔNG CÔNG VIỆC	5
TÓM TẮT.....	6
MỤC LỤC	7
MỤC LỤC HÌNH ẢNH	9
LÝ DO CHỌN ĐỀ TÀI.....	1
MỤC TIÊU ĐỀ TÀI.....	2
Ý NGHĨA CỦA ĐỀ TÀI.....	3
MỞ ĐẦU	4
CHƯƠNG 1: EMAIL VÀ EMAIL LỪA ĐẢO	5
1.1. Phishing là gì?	5
1.2. Lịch sử của Email lừa đảo.....	6
1.3. Các loại phishing email phổ biến	6
1.4.2. Đối với doanh nghiệp.....	8
1.5. Đặc điểm nhận dạng email lừa đảo	9
1.6. Sử dụng học máy để phát hiện email lừa đảo	10
1.7. Lợi ích và hạn chế của việc sử dụng thuật toán trong phát hiện email lừa đảo	11
1.7.1. Lợi ích	11
1.7.2. Hạn Chế	12
CHƯƠNG 2: THUẬT TOÁN RANDOM FOREST VÀ CÁC KHÁI NIỆM LIÊN QUAN	16
2.1 Machine Learning là gì?.....	16
2.2. Phương pháp học trong machine learning.....	16
2.2.1. Phương pháp học có giám sát (Supervised Learning)	16
2.2.2. Phương pháp học không giám sát (Unsupervised Learning).....	17
2.2.3. Phương pháp học bán giám sát (Semi-Supervised Learning).....	17
2.3. Random Forest là gì?.....	17
2.4. Xây dựng thuật toán Random Forest	18
2.5. Nguyên lý hoạt động	20
2.6. Ưu điểm của Random Forest.....	20
2.7. Nhược điểm của Random Forest.....	20
2.8. Ứng dụng của Random Forest.....	20
2.9. Mã giả.....	21

CHƯƠNG 3: TRIỂN KHAI & THỰC NGHIỆM.....	25
3.1. Chuẩn bị môi trường	25
3.1.1. Hệ điều hành và công cụ.....	25
3.1.2. Thư viện	25
3.1.3. Bộ dữ liệu dataset.....	26
3.2. Triển khai và thực nghiệm	26
3.3. Kiểm thử.....	36
3.3.1. Kiểm thử với bộ dữ liệu SpamAssassin.....	36
3.3.2. Kiểm thử với bộ dữ liệu Kaggle	39
3.4. So sánh kết quả khi xử lý từ bộ dữ liệu SpamAssassin, Kaggle	41
KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....	42
Kết luận	42
Hướng phát triển	42
TÀI LIỆU THAM KHẢO	44

MỤC LỤC HÌNH ẢNH

Hình 1.1. Báo cáo những email lừa đảo	8
Hình 1.2. Mô hình đặc tả về Random Forest.....	11
Hình 1.3. Email lừa đảo kèm link.....	13
Hình 1.4. Email với các keyword lừa đảo	14
Hình 1.5. Tên miền không có chứng chỉ bảo mật SSL.....	14
Hình 1.6. Trang thật nhìn rất chuyên nghiệp và đầy đủ thông tin.....	15
Hình 2.1. Random Forest được phát triển dựa trên Cây quyết định.....	18
Hình 2.2. Cơ chế Random Sampling with Replacement.....	19
Hình 3.1. Khởi tạo thư viện	25
Hình 3.2. Tiến hành tiền xử lý dữ liệu.....	27
Hình 3.3. Tiến hành Vector hóa dữ liệu	28
Hình 3.4. Tiến hành Gán nhãn cho các dữ liệu	29
Hình 3.5. Tiến hành chia dữ liệu để huấn luyện.....	30
Hình 3.6. Tiến hành khởi tạo và huấn luyện theo tập dữ liệu đã chia với bộ dữ liệu SpamAssassin	30
Hình 3.7. Tiến hành khởi tạo và huấn luyện theo tập dữ liệu đã chia với bộ dữ liệu Kaggle	31
Hình 3.7. Đánh giá mô hình trên tập dữ liệu test.....	31
Hình 3.10. Đánh giá mô hình Random Forest cho bộ dữ liệu SpamAssassin.....	35
Hình 3.11. Đánh giá mô hình Random Forest cho bộ dữ liệu Kaggle	35
Hình 3.12. Hàm nhận diện email phishing (1)	36
Hình 3.13. Hàm nhận diện email phishing (2)	36
Hình 3.14. Kiểm thử với email spam cho bộ dữ liệu SpamAssassin	37
Hình 3.15. Kết quả email spam với bộ dữ liệu SpamAssassin.....	37
Hình 3.16. Kiểm thử với email ham cho bộ dữ liệu SpamAssassin.....	37
Hình 3.17. Kết quả email ham với bộ dữ liệu SpamAssassin	38
Hình 3.18. Kiểm thử với email spam nhưng mô hình nhận định là ham cho bộ dữ liệu SpamAssassin	38
Hình 3.19. Kết quả với email spam nhưng mô hình nhận định là ham cho bộ dữ liệu SpamAssassin	38
Hình 3.20. Kiểm thử với email ham nhưng mô hình nhận định là spam cho bộ dữ liệu SpamAssassin	39
Hình 3.21. Kết quả với email ham nhưng mô hình nhận định là spam cho bộ dữ liệu SpamAssassin	39
Hình 3.22. Kiểm thử với email spam mẫu cho bộ dữ liệu Kaggle.....	39
Hình 3.23. Kết quả email spam với bộ dữ liệu Kaggle	40
Hình 3.24. Kiểm thử với email ham mẫu cho bộ dữ liệu Kaggle	40
Hình 3.25. Kết quả email ham với bộ dữ liệu Kaggle.....	40
Hình 3.26. Kiểm thử với email spam nhưng mô hình nhận định là ham cho bộ dữ liệu Kaggle.....	40
Hình 3.27. Kết quả với email spam nhưng mô hình nhận định là ham cho bộ dữ liệu Kaggle	41
Hình 3.28. Kiểm thử với email ham nhưng mô hình nhận định là spam với bộ dữ liệu Kaggle	41
Hình 3.29. Kết quả với email ham nhưng mô hình nhận định là spam với bộ dữ liệu Kaggle.....	41

LÝ DO CHỌN ĐỀ TÀI

Sự bùng nổ công nghệ đã biến internet thành một phần không thể thiếu trong cuộc sống hiện đại. Các thiết bị thông minh được kết nối liên tục, giúp việc trao đổi thông tin diễn ra nhanh chóng hơn bao giờ hết. Trong đó, thư điện tử (email) đóng vai trò quan trọng trong giao tiếp cá nhân lẫn công việc. Nếu như trước đây, thư từ truyền thống đòi hỏi giấy mực và mất nhiều thời gian để vận chuyển, thì nay, chỉ với một cú nhấp chuột, email có thể được gửi đi tức thì mà không mất chi phí vận chuyển.

Tuy nhiên, song song với sự tiện lợi, email cũng tiềm ẩn nhiều rủi ro an ninh nghiêm trọng. Tin tặc có thể lợi dụng các lỗ hổng để chặn bắt, chỉnh sửa nội dung hoặc đánh cắp thông tin cá nhân. Điều này đòi hỏi các phương pháp bảo mật tiên tiến như chữ ký số và mã hóa dữ liệu để đảm bảo an toàn trong quá trình gửi nhận email.

Một trong những mối đe dọa phổ biến nhất hiện nay là phishing email – hình thức tấn công lừa đảo qua email. Kẻ xấu có thể mạo danh các tổ chức uy tín, doanh nghiệp hoặc cá nhân nhằm đánh lừa người dùng cung cấp thông tin nhạy cảm như tài khoản ngân hàng, mật khẩu hoặc thông tin cá nhân. Đặc biệt, trong bối cảnh nhiều công ty chuyển sang làm việc từ xa do ảnh hưởng của đại dịch Covid-19, tần suất trao đổi email ngày càng tăng, tạo điều kiện cho các cuộc tấn công phishing diễn ra mạnh mẽ hơn.

Không dừng lại ở đó, các chiêu trò lừa đảo ngày càng tinh vi, từ giả mạo thông báo bảo mật, kêu gọi đóng góp từ thiện đến gửi đường dẫn chứa mã độc. Những hành vi này không chỉ gây tổn thất tài chính mà còn làm suy giảm niềm tin của con người vào các nền tảng số.

Vậy làm thế nào để phát hiện và ngăn chặn những email lừa đảo này? Giải pháp hiệu quả đang được áp dụng rộng rãi chính là Machine Learning. Bằng cách phân tích các đặc điểm của email, mô hình học máy có thể phân loại và nhận diện email giả mạo với độ chính xác cao. Việc ứng dụng trí tuệ nhân tạo trong bảo mật email không chỉ giúp giảm thiểu rủi ro mà còn góp phần tạo ra một môi trường internet an toàn hơn cho tất cả mọi người.

MỤC TIÊU ĐỀ TÀI

Đề tài "Nghiên cứu và ứng dụng thuật toán Random Forest để phát hiện email lừa đảo (Phishing Email)" hướng đến các mục tiêu cụ thể sau: nghiên cứu về Phishing Email.

Hiểu rõ về phishing email và tầm quan trọng của việc phát hiện các cuộc tấn công lừa đảo trong môi trường trực tuyến.

Nghiên cứu cách triển khai giải pháp Machine Learning, cụ thể là thuật toán Random Forest, vào hệ thống phát hiện email lừa đảo, tập trung vào việc bảo mật thông tin người dùng và hệ thống email doanh nghiệp.

Xây dựng giải pháp bảo vệ hệ thống email khỏi các mối đe dọa an ninh mạng, đảm bảo chỉ những email hợp lệ mới được gửi đến người dùng, góp phần giảm thiểu rủi ro mất cắp thông tin và tấn công lừa đảo trực tuyến.

Đánh giá hiệu quả của mô hình Random Forest trong việc phát hiện phishing email, từ đó hỗ trợ cải thiện hệ thống bảo mật email doanh nghiệp và cá nhân.

Đối tượng và phạm vi nghiên cứu:

- Đối tượng nghiên cứu: Giải pháp phát hiện phishing email bằng thuật toán Machine Learning, đặc biệt tập trung vào mô hình Random Forest và các yếu tố ảnh hưởng đến độ chính xác của mô hình này.

- Phạm vi nghiên cứu: Tập trung vào các email lừa đảo (phishing email) trong hệ thống email doanh nghiệp và cá nhân. Giới hạn trong việc thu thập, xử lý dữ liệu và đánh giá mô hình trên tập dữ liệu mẫu thay vì triển khai trực tiếp vào hệ thống thực tế.

- Phương pháp nghiên cứu: Nghiên cứu tài liệu, thu thập và phân tích các tài liệu liên quan đến phishing email, kỹ thuật tấn công lừa đảo qua email và các phương pháp phát hiện phishing email. Nghiên cứu về thuật toán Random Forest và cách áp dụng vào bài toán phân loại email.

Thực nghiệm:

- Thu thập tập dữ liệu chứa email hợp lệ và email lừa đảo từ các nguồn có sẵn.
- Tiền xử lý dữ liệu và trích xuất đặc trưng quan trọng để phục vụ huấn luyện mô hình.
- Huấn luyện mô hình Random Forest để phân loại email hợp lệ và phishing email.
- Đánh giá và phân tích
- Thực hiện các thử nghiệm để kiểm tra độ chính xác, độ nhạy, độ đặc hiệu của mô hình.
- Phân tích kết quả và so sánh với các phương pháp khác để đánh giá hiệu suất mô hình.

Ý NGHĨA CỦA ĐỀ TÀI

Tăng cường an ninh mạng: Giải pháp giúp phát hiện sớm các email lừa đảo, bảo vệ người dùng và doanh nghiệp khỏi các cuộc tấn công phishing.

Cải thiện hiệu quả phát hiện phishing email: Ứng dụng Machine Learning, đặc biệt là thuật toán Random Forest, giúp nâng cao độ chính xác trong việc phân loại email hợp lệ và phishing email.

Hỗ trợ doanh nghiệp và cá nhân bảo vệ thông tin: Việc nghiên cứu và phát triển hệ thống phát hiện phishing email có thể được triển khai trong thực tế để nâng cao khả năng bảo mật email và giảm thiểu rủi ro mất cắp thông tin.

MỞ ĐẦU

Lý do nhóm chúng em lựa chọn đề tài này bắt nguồn từ thực trạng trong thời đại công nghệ số phát triển mạnh mẽ, các hệ thống thư điện tử (email) đang đóng vai trò cực kỳ quan trọng trong việc trao đổi thông tin của cá nhân, tổ chức và doanh nghiệp. Tuy nhiên, cùng với sự phổ biến của email là sự gia tăng nhanh chóng của các cuộc tấn công mạng, đặc biệt là email lừa đảo (phishing email) – một trong những hình thức tấn công phổ biến và nguy hiểm nhất hiện nay. Những cuộc tấn công này không chỉ gây thiệt hại về tài chính mà còn ảnh hưởng nghiêm trọng đến bảo mật thông tin và quyền riêng tư của người dùng.

Chính vì vậy, nhóm chúng em quyết định thực hiện đề tài "Nghiên cứu và ứng dụng thuật toán Random Forest để phát hiện email lừa đảo (Phishing Email)". Mục tiêu của đề tài là tìm hiểu sâu về bản chất của phishing email và áp dụng các kỹ thuật của Machine Learning, cụ thể là thuật toán Random Forest, nhằm xây dựng một mô hình phát hiện email lừa đảo hiệu quả.

Với khả năng học và phân loại dữ liệu mạnh mẽ, Random Forest không chỉ giúp nâng cao độ chính xác trong việc phát hiện các email đáng ngờ, mà còn góp phần cải thiện hệ thống bảo mật của người dùng và tổ chức. Đề tài không chỉ mang lại giá trị học thuật trong lĩnh vực trí tuệ nhân tạo và an toàn thông tin, mà còn có ý nghĩa thực tiễn cao trong việc giảm thiểu rủi ro an ninh mạng và tăng cường lòng tin vào các hệ thống giao tiếp trực tuyến.

CHƯƠNG 1: EMAIL VÀ EMAIL LỪA ĐẢO

Email (Electronic Mail) là một phương thức trao đổi thư tín điện tử thông qua mạng Internet. Với tốc độ truyền tải nhanh chóng, chi phí thấp và khả năng lưu trữ tiện lợi, email đã trở thành công cụ giao tiếp phổ biến trong cả đời sống cá nhân lẫn môi trường học tập, làm việc và kinh doanh. Chỉ với một tài khoản email và kết nối Internet, người dùng có thể dễ dàng gửi và nhận thư từ, tài liệu hoặc thông tin với bạn bè, đồng nghiệp và đối tác ở bất kỳ đâu. Ngày nay, email gần như là một phần không thể thiếu trong hệ sinh thái công nghệ, được tích hợp vào các hệ thống quản lý, nền tảng học tập trực tuyến, ngân hàng điện tử, thương mại điện tử, và nhiều lĩnh vực khác. Tuy nhiên, chính vì sự phổ biến và vai trò quan trọng đó, email cũng trở thành mục tiêu tấn công của các tin tặc với nhiều hình thức khác nhau, trong đó phổ biến nhất là email lừa đảo (phishing email). Phishing email là hình thức tấn công giả mạo nhằm đánh lừa người dùng cung cấp thông tin cá nhân nhạy cảm như tài khoản đăng nhập, mật khẩu, thông tin ngân hàng,... Những email này thường ngụy trang dưới dạng thư từ tổ chức uy tín như ngân hàng, cơ quan nhà nước, trường học hay doanh nghiệp. Nếu người dùng không cảnh giác, rất dễ bị lừa đảo, dẫn đến mất thông tin, tiền bạc và gây ảnh hưởng nghiêm trọng đến cá nhân hoặc tổ chức. Do đó, việc nghiên cứu các giải pháp tự động nhằm phát hiện và ngăn chặn phishing email là một nhu cầu cấp thiết. Trong số các giải pháp đó, ứng dụng các thuật toán Machine Learning, đặc biệt là thuật toán Random Forest, đang cho thấy hiệu quả cao trong việc phân loại và nhận diện các email có dấu hiệu lừa đảo.

1.1. Phising là gì?

Phishing (Tấn công giả mạo) là hình thức tấn công mạng mà kẻ tấn công giả mạo thành một đơn vị uy tín lừa đảo người dùng cung cấp thông tin cá nhân cho chúng

Thông thường, tin tặc sẽ giả mạo thành các tổ chức uy tín như ngân hàng, trang web giao dịch trực tuyến, ví điện tử, công ty thẻ tín dụng nhằm đánh lừa người dùng chia sẻ những thông tin nhạy cảm như tài khoản và mật khẩu đăng nhập, mật khẩu giao dịch, thông tin thẻ tín dụng và các dữ liệu cá nhân có giá trị khác.

Phương thức tấn công này thường được thực hiện thông qua email hoặc tin nhắn giả mạo. Khi người dùng mở email và nhấp vào liên kết được đính kèm, họ sẽ được dẫn tới một trang đăng nhập giả mạo. Nếu không cảnh giác và nhập thông tin, người dùng sẽ ngay lập tức bị tin tặc chiếm đoạt dữ liệu.

Phishing là một hình thức tấn công đã được biết đến từ năm 1987. Thuật ngữ “Phishing” được hình thành từ sự kết hợp giữa hai khái niệm: “fishing for information” (câu thông tin) và

“phreaking” (hành vi gian lận hệ thống điện thoại). Từ “Phishing” mang ý nghĩa tương đồng với hành động “câu cá”, nhưng trong ngữ cảnh này là hành động “câu” thông tin người dùng một cách tinh vi và lừa đảo.

1.2. Lịch sử của Email lừa đảo

Email lừa đảo (phishing email) đã xuất hiện từ những ngày đầu của Internet và không ngừng biến đổi để đánh lừa người dùng. Năm 1995, thuật ngữ "phishing" chính thức ra đời khi nhóm hacker AOHell gửi hàng loạt email giả mạo đến người dùng AOL (America Online), yêu cầu họ cung cấp mật khẩu với lý do bảo trì hệ thống. Sang thập niên 2000, email lừa đảo trở nên tinh vi hơn nhắm vào các ngân hàng và dịch vụ tài chính, thường giả danh các tổ chức uy tín để đánh cắp thông tin. Đặc biệt, từ năm 2010 đến nay, với sự phát triển của mạng xã hội và công nghệ, hacker không chỉ sử dụng email mà còn kết hợp tin nhắn, website giả mạo, thậm chí AI để tạo ra những chiến dịch phishing có tính cá nhân hóa cao.

1.3. Các loại phishing email phổ biến

Phishing email là hình thức lừa đảo trực tuyến ngày càng tinh vi, nhắm vào nhiều đối tượng khác nhau. Dưới đây là 18 loại phishing email thường gặp trong thực tế:

- Giả mạo đơn đặt hàng: Email giả danh nhà cung cấp, đính kèm hóa đơn hoặc thông tin đơn hàng giả, dụ người dùng tải file độc hại.
- Giả mạo hóa đơn thanh toán: Tin tặc giả danh doanh nghiệp, yêu cầu thanh toán hóa đơn qua liên kết hoặc file đính kèm chứa malware.
- Giả mạo nâng cấp tài khoản email: Email trá hình từ "Google", "Microsoft" hoặc bộ phận IT công ty, yêu cầu cập nhật thông tin cá nhân qua trang web giả mạo.
- Giả mạo phí trả trước: Lừa người dùng thanh toán trước cho dịch vụ/sản phẩm không tồn tại, thường nhắm vào doanh nghiệp nhỏ.
- Giả mạo tài liệu Google Docs: Email giả danh người quen, dụ nạn nhân nhập thông tin đăng nhập Gmail trên trang web giả.
- Giả mạo thông báo PayPal: Tin tặc giả mạo PayPal, thông báo "tài khoản bị khóa" để chiếm đoạt thông tin tài chính.

- Giả mạo phòng Nhân sự (HR): Email giả mạo thông báo nội bộ từ HR, đính kèm file độc hại hoặc liên kết lừa đảo.
- Giả mạo cơ quan Nhà nước: Lợi dụng uy tín của cơ quan chính phủ để đe dọa hoặc yêu cầu cung cấp thông tin cá nhân.
- Giả mạo thông báo Dropbox: Email giả mạo thông báo chia sẻ file, dẫn đến trang web lừa đảo đánh cắp tài khoản.
- Giả mạo chi cục thuế: Thông báo nợ thuế hoặc hoàn thuế giả, yêu cầu nhập thông tin ngân hàng.
- Giả mạo thông báo xâm nhập tài khoản: Gây hoang mang bằng email cảnh báo "tài khoản bị đăng nhập bất thường" để chiếm quyền điều khiển.
- Giả mạo rút tiền: Yêu cầu xác minh thông tin tài khoản ngân hàng qua biểu mẫu giả mạo.
- Giả mạo trúng thưởng: Dụ dỗ bằng giải thưởng hấp dẫn (tiền mặt, xe hơi) để lừa đảo.
- Giả mạo nạn nhân: Tin tặc giả vờ là khách hàng bị lừa, đe dọa kiện cáo nếu không cung cấp thông tin.
- Giả mạo kiểm tra hệ thống: Email yêu cầu "xác minh tài khoản" do doanh nghiệp kiểm tra, thực chất là lừa đảo.
- Giả mạo người quen cũ: Mạo danh bạn bè/đồng nghiệp cũ để vay tiền hoặc dụ tải phần mềm độc hại.
- Giả mạo thông báo quá hạn thanh toán: Cảnh báo dịch vụ sắp bị ngừng để đánh cắp thông tin đăng nhập.
- Giả mạo thông báo quá tải dung lượng email: Tin tặc gửi email thông báo dung lượng email sắp đầy, yêu cầu người dùng nhấp vào liên kết "nâng cấp". Khi click, mã độc sẽ xâm nhập hệ thống.

1.4. Cách phòng chống Phishing

1.4.1. Đối với cá nhân

Để tránh bị hacker sử dụng tấn công phishing để lừa đảo internet, thu thập dữ liệu cá nhân và thông tin nhạy cảm cá nhân cần:

- Cảnh giác với các email có xu hướng thúc giục bạn nhập thông tin nhạy cảm. Cho dù lời kêu gọi có hấp dẫn thế nào đi chăng nữa thì vẫn nên kiểm tra kỹ càng. VD: bạn mới mua sắm online, đột nhiên có email từ ngân hàng tới đề nghị hoàn tiền cho bạn, chỉ cần nhập thông tin thẻ đã dùng để thanh toán. Có tin được không?

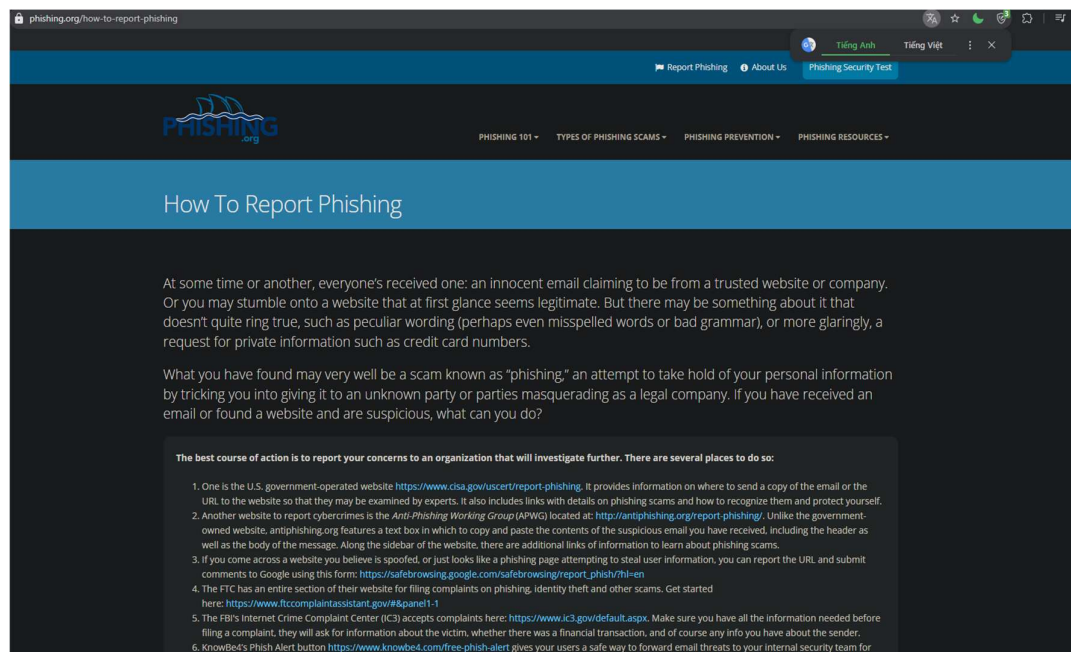
- Không click vào bất kỳ đường link nào được gửi qua email nếu bạn không chắc chắn 100% an toàn.

- Không bao giờ gửi thông tin bí mật qua email.

- Không trả lời những thư lừa đảo. Những kẻ gian lận thường gửi cho bạn số điện thoại để bạn gọi cho họ vì mục đích kinh doanh. Họ sử dụng công nghệ Voice over Internet Protocol. Với công nghệ này, các cuộc gọi của họ không bao giờ có thể được truy tìm.

- Sử dụng Tường lửa và phần mềm diệt virus. Hãy nhớ luôn cập nhật phiên bản mới nhất của các phần mềm này.

- Hãy chuyển tiếp các thư rác đến spam@uce.gov. cũng có thể gửi email tới reportphishing@antiphishing.org Tổ chức này giúp chống lại các phishing khác.



Hình 1.1. Báo cáo những email lừa đảo

1.4.2. Đối với doanh nghiệp

Training cho nhân viên để tăng kiến thức sử dụng internet an toàn. Thường xuyên tổ chức các buổi tập huấn, diễn tập các tình huống giả mạo.

Sử dụng dịch vụ G-suite dành cho doanh nghiệp, không nên sử dụng dịch vụ Gmail miễn phí vì dễ bị giả mạo.

Triển khai bộ lọc SPAM để phòng tránh thư rác, lừa đảo.

Luôn cập nhật các phần mềm, ứng dụng để tránh các lỗ hổng bảo mật có thể bị kẻ tấn công lợi dụng.

1.5. Đặc điểm nhận dạng email lừa đảo

Để phát hiện email lừa đảo, người dùng có thể áp dụng các phương pháp nhận diện thông qua những đặc điểm sau:

- **Kiểm tra tên miền của địa chỉ email:** Một trong những đặc điểm đầu tiên cần lưu ý là không có bất kỳ công ty nào sẽ liên lạc với bạn từ một tên miền công cộng như “@gmail.com”, “@yahoo.com”, hay “@outlook.com”. Các công ty, tổ chức hay tập đoàn thường sử dụng tên miền email riêng biệt của họ. Ví dụ, Google sẽ không bao giờ gửi email từ địa chỉ như “xyz@gmail.com”, mà thay vào đó, họ sẽ sử dụng tên miền của Google như “@google.com”.

Nếu tên miền sau ký tự “@” trong địa chỉ email trùng khớp với tên miền chính thức của công ty, thì đó là một email hợp pháp. Ngược lại, nếu tên miền không trùng khớp hoặc là tên miền công cộng, khả năng cao đây là email giả mạo.

- **Kiểm tra các mục Mailed-by, Signed-by, và Security:** Tin tặc có thể sử dụng các máy chủ giả mạo để làm giả địa chỉ email (ví dụ: "xyz@google.com"), tuy nhiên chúng không thể làm giả các chứng nhận bảo mật như mục "mailed-by", "signed-by" và "security". Những mục này sẽ cung cấp thông tin xác thực cho email.

Mailed-by và Signed-by chỉ ra rằng email đã được xác thực bằng phương thức SPF (Sender Policy Framework) và DKIM (DomainKeys Identified Mail), giúp người nhận xác minh tính hợp lệ của địa chỉ email.

Security cho biết liệu email có được mã hóa bằng chuẩn TLS hoặc SSL trong suốt quá trình gửi hay không, đảm bảo rằng không có bên thứ ba nào có thể nghe trộm hoặc thay đổi nội dung email.

Email hợp pháp từ các tổ chức, ngân hàng thường có đủ các mục trên, trong khi email lừa đảo thường không có hoặc thông tin được cung cấp rất mơ hồ.

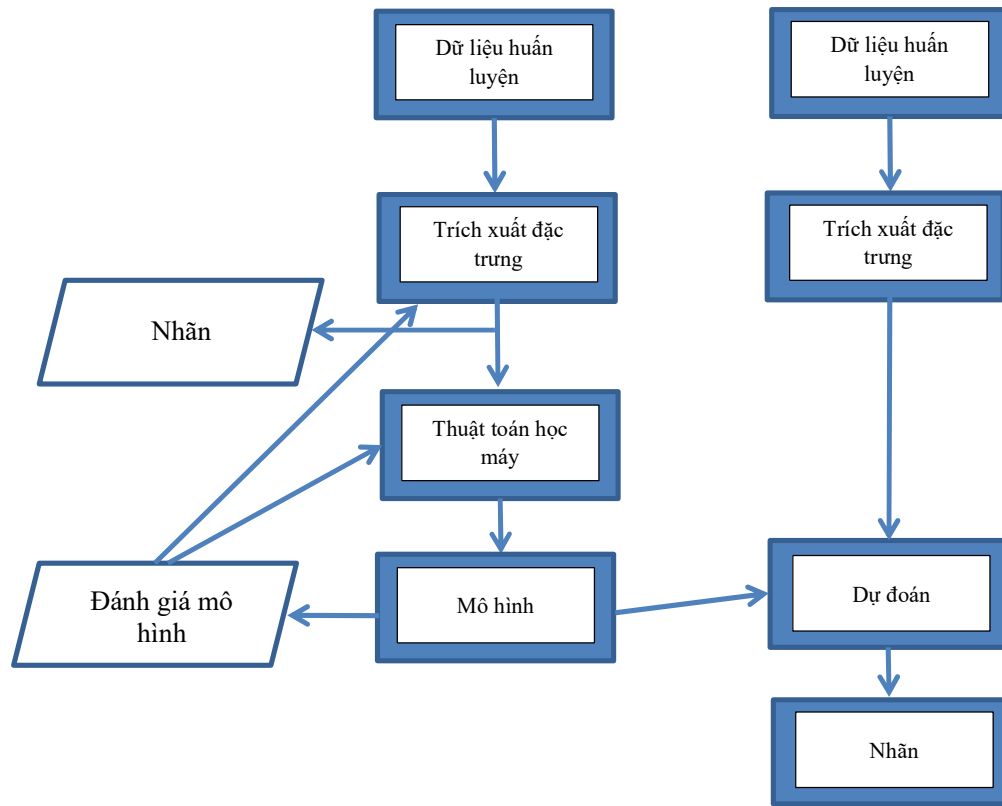
- **Nhận diện liên kết, nút bấm lừa đảo trong email:** Trước khi nhấp vào bất kỳ liên kết hoặc nút bấm nào trong email, bạn nên rê chuột lên chúng để kiểm tra địa chỉ thực của liên kết. Nếu liên kết dẫn đến trang web giả mạo, hoặc trang web không liên quan đến công ty chính thức, đó là dấu hiệu của một email lừa đảo. Các nút bấm trong email giả mạo cũng có thể dẫn bạn đến các trang web chứa mã độc hoặc các trang web lừa đảo.

- **Kiểm tra lỗi chính tả và ngữ pháp trong email:** Email lừa đảo thường chứa nhiều lỗi chính tả và ngữ pháp. Nếu nội dung email có nhiều lỗi hoặc cách trình bày lộn xộn, đây có thể là dấu hiệu cảnh báo. Các email từ các tổ chức uy tín thường rất kỹ lưỡng trong việc soát lỗi và có ngữ pháp chuẩn mực. Bên cạnh đó, nếu email chứa các từ khóa mà bạn thường thấy trong các email rác trước đây, đó là dấu hiệu cần chú ý.

- **Kiểm tra file đính kèm:** Email lừa đảo thường có các file đính kèm chứa mã độc. Nếu bạn nhận được email từ người gửi không quen thuộc và có file đính kèm, không nên mở chúng trừ khi bạn chắc chắn về độ an toàn của tệp đó. Phần lớn các file đính kèm trong email lừa đảo là các file có thể lây nhiễm virus vào hệ thống của bạn khi mở lên.

1.6. Sử dụng học máy để phát hiện email lừa đảo

Ngoài các phương pháp kiểm tra thủ công, hiện nay công nghệ học máy (machine learning) ngày càng được ứng dụng rộng rãi trong việc phát hiện email lừa đảo. Các mô hình học máy có thể được huấn luyện trên tập dữ liệu gồm hàng ngàn email hợp lệ và lừa đảo, từ đó tự động nhận diện các đặc điểm như ngôn ngữ, cấu trúc, địa chỉ người gửi, liên kết đáng ngờ, hay tệp đính kèm chứa mã độc. Nhờ khả năng học và thích nghi, hệ thống có thể phát hiện những email giả mạo mới mà chưa từng xuất hiện trước đó. Việc áp dụng học máy giúp tăng độ chính xác, giảm thiểu rủi ro và tiết kiệm thời gian cho người dùng cũng như tổ chức.



Hình 1.2. Mô hình đặc tả về Random Forest

1.7. Lợi ích và hạn chế của việc sử dụng thuật toán trong phát hiện email lừa đảo

1.7.1. Lợi ích

Tăng độ chính xác và hiệu quả: Một trong những lợi ích lớn nhất khi sử dụng thuật toán, đặc biệt là các thuật toán học máy như Random Forest, trong phát hiện email lừa đảo là khả năng xử lý lượng lớn dữ liệu mà con người không thể làm được trong thời gian ngắn. Các thuật toán học máy có thể nhận diện các mẫu email lừa đảo, phân tích nội dung, địa chỉ người gửi, liên kết, tệp đính kèm và các yếu tố khác một cách tự động và nhanh chóng. Điều này giúp nâng cao độ chính xác trong việc phân loại email và giảm thiểu tỷ lệ bỏ sót (false negative) khi so với việc làm thủ công.

Tự động hóa và tiết kiệm thời gian: Một lợi ích khác của việc sử dụng thuật toán trong việc phát hiện email lừa đảo là tính tự động hóa. Việc áp dụng các thuật toán giúp loại bỏ công việc kiểm tra thủ công, giảm tải công việc cho người dùng và các tổ chức, đồng thời có thể phát hiện email lừa đảo ngay lập tức. Những người dùng không phải mất công kiểm tra từng email, tránh việc bị lừa đảo và tiết kiệm thời gian cho các công việc khác.

Cập nhật và học hỏi liên tục: Các mô hình học máy như Random Forest có khả năng học hỏi từ các dữ liệu mới, giúp hệ thống tự động cập nhật và cải thiện khả năng nhận diện. Mỗi khi hệ thống được cung cấp dữ liệu mới về các mẫu email lừa đảo hoặc phương thức tấn công mới, thuật toán có thể điều chỉnh và học cách phát hiện những kiểu lừa đảo mới, từ đó trở nên chính xác và hiệu quả hơn theo thời gian. Điều này rất quan trọng trong bối cảnh tội phạm mạng và email lừa đảo luôn thay đổi để tránh bị phát hiện.

Tăng Cường bảo mật và bảo vệ người dùng: Việc sử dụng thuật toán trong phát hiện email lừa đảo đóng vai trò quan trọng trong việc bảo vệ người dùng khỏi những mối đe dọa tiềm ẩn từ các email giả mạo. Những email lừa đảo có thể chứa mã độc, phần mềm gián điệp, hoặc các cuộc tấn công phishing để lấy cắp thông tin cá nhân của người dùng. Bằng cách tự động phát hiện và ngăn chặn những email này trước khi người dùng mở chúng, thuật toán giúp nâng cao mức độ bảo mật cho các cá nhân và tổ chức, tránh được những rủi ro tài chính và bảo mật nghiêm trọng.

Giảm Thiểu chi phí cho tổ chức: Khi các thuật toán phát hiện email lừa đảo hoạt động hiệu quả, các tổ chức không cần phải phụ thuộc vào các biện pháp bảo mật đắt đỏ khác hoặc đội ngũ chuyên gia bảo mật. Hệ thống có thể tự động nhận diện và ngăn chặn các email lừa đảo mà không cần sự can thiệp thủ công. Điều này giúp tổ chức tiết kiệm chi phí liên quan đến việc quản lý bảo mật mạng và đào tạo nhân viên.

1.7.2. Hạn Chế

Yêu cầu dữ liệu huấn luyện lớn và chất lượng: Một trong những hạn chế lớn của việc sử dụng thuật toán học máy trong phát hiện email lừa đảo là yêu cầu về một lượng lớn và đa dạng dữ liệu huấn luyện. Các mô hình học máy cần được huấn luyện trên hàng triệu email hợp lệ và lừa đảo để có thể nhận diện được các mẫu hành vi giả mạo trong email. Tuy nhiên, không phải lúc nào cũng dễ dàng có đủ dữ liệu chất lượng để huấn luyện mô hình, đặc biệt là khi các hình thức email lừa đảo mới liên tục xuất hiện và các dữ liệu cũ trở nên lỗi thời.

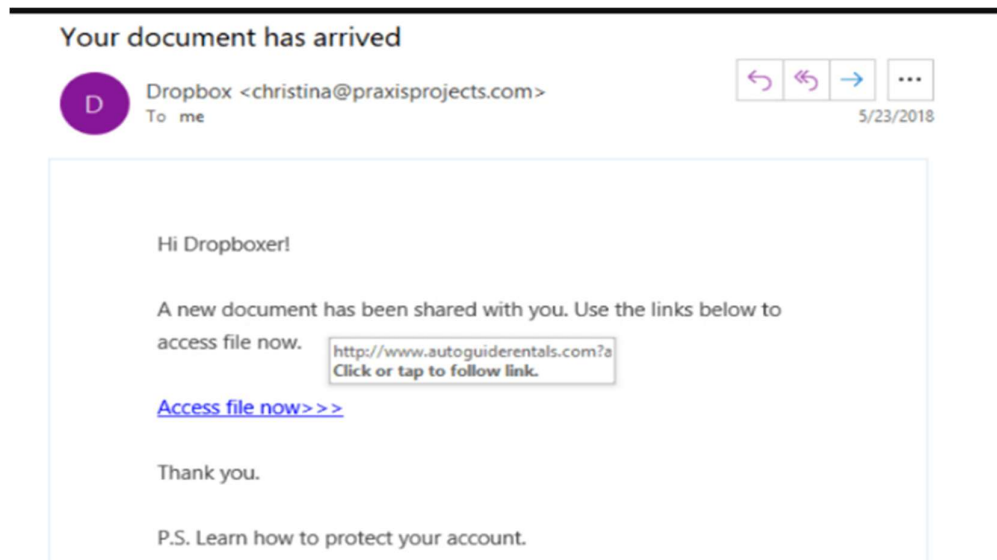
Khả Năng nhầm lẫn cao: Dù thuật toán học máy có thể rất mạnh mẽ, nhưng chúng cũng có thể gặp phải vấn đề "false positive" và "false negative". False positive xảy ra khi thuật toán nhận diện nhầm một email hợp lệ là lừa đảo, dẫn đến việc người dùng không nhận được email quan trọng. Ngược lại, false negative xảy ra khi thuật toán không phát hiện được một email lừa đảo, khiến người dùng rơi vào bẫy. Điều này có thể làm giảm độ tin cậy của hệ thống và gây khó khăn trong việc sử dụng.

Phụ Thuộc vào chất lượng dữ liệu: Thuật toán học máy chỉ mạnh mẽ khi có dữ liệu chất lượng. Nếu dữ liệu huấn luyện không đầy đủ, không chính xác, hoặc không cập nhật với các phương thức tấn công mới, mô hình có thể không thể phát hiện chính xác các email lừa đảo. Do đó, việc thu thập và duy trì dữ liệu huấn luyện chất lượng là một yếu tố quan trọng, nhưng cũng rất khó khăn và tốn kém.

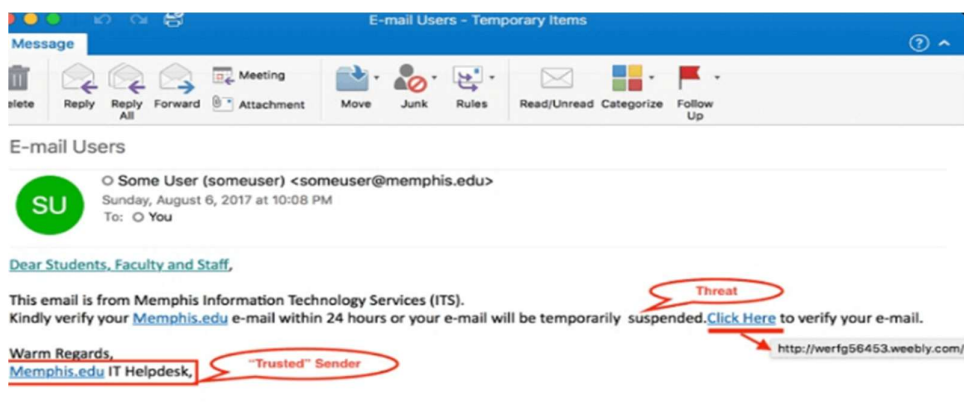
Đòi hỏi tài nguyên tính toán lớn: Các thuật toán học máy, đặc biệt là các mô hình phức tạp như Random Forest, yêu cầu tài nguyên tính toán lớn. Việc xử lý và phân tích một khối lượng lớn dữ liệu email có thể làm giảm hiệu suất của hệ thống nếu không có phần cứng phù hợp. Điều này có thể dẫn đến chi phí cao khi triển khai hệ thống bảo mật, đặc biệt đối với các tổ chức lớn hoặc các ứng dụng cần xử lý email trong thời gian thực.

Cần bảo trì và cập nhật thường xuyên: Một hạn chế khác của thuật toán học máy trong phát hiện email lừa đảo là việc cần bảo trì và cập nhật thường xuyên. Vì các phương thức tấn công email lừa đảo luôn thay đổi, hệ thống cần phải liên tục được điều chỉnh và cập nhật với các mối đe dọa mới. Việc này đòi hỏi các nhà quản lý phải đầu tư vào việc duy trì và làm mới các mô hình học máy, cũng như các hệ thống hỗ trợ để đảm bảo hiệu quả của thuật toán.

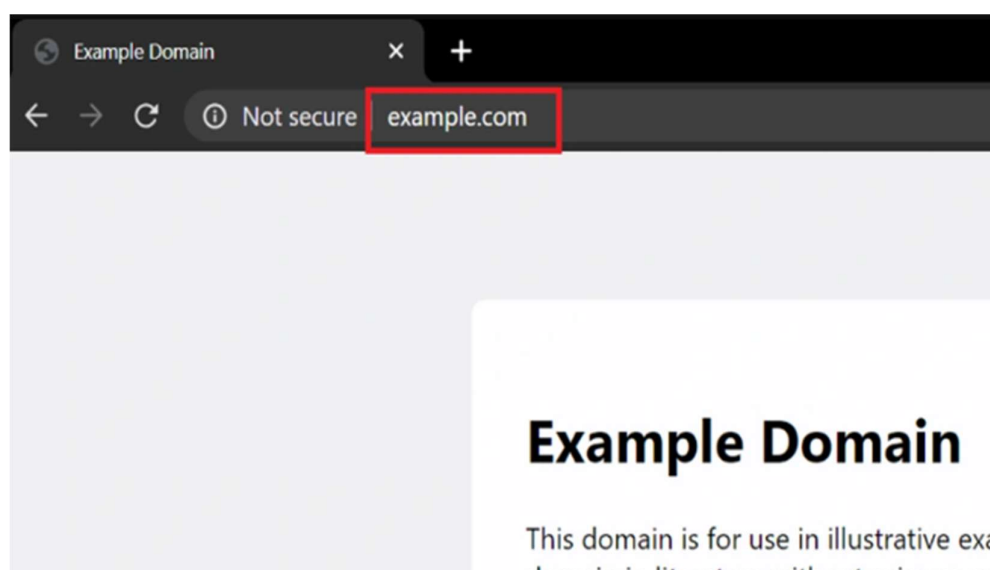
Khả năng tấn công vào thuật toán: Mặc dù học máy có thể rất hiệu quả trong việc phát hiện email lừa đảo, nhưng các hacker cũng có thể tìm cách tấn công vào mô hình này. Bằng cách sử dụng các phương pháp lừa đảo tiên tiến, tin tặc có thể cố gắng đánh lừa hệ thống học máy để nó nhận diện sai. Việc bảo vệ thuật toán khỏi những kiểu tấn công này đòi hỏi sự đầu tư vào an ninh hệ thống và việc giám sát liên tục.



Hình 1.3. Email lừa đảo kèm link

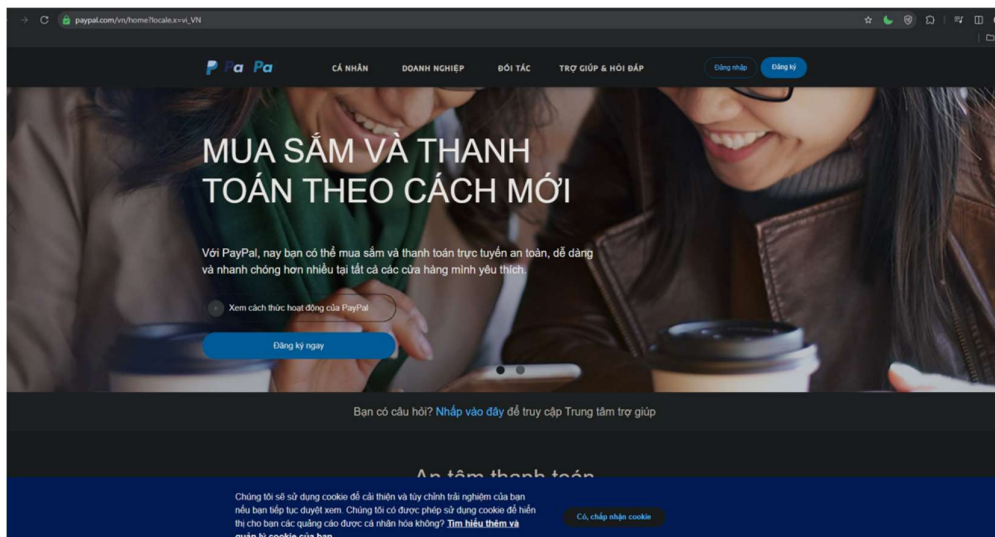


Hình 1.4. Email với các keyword lừa đảo



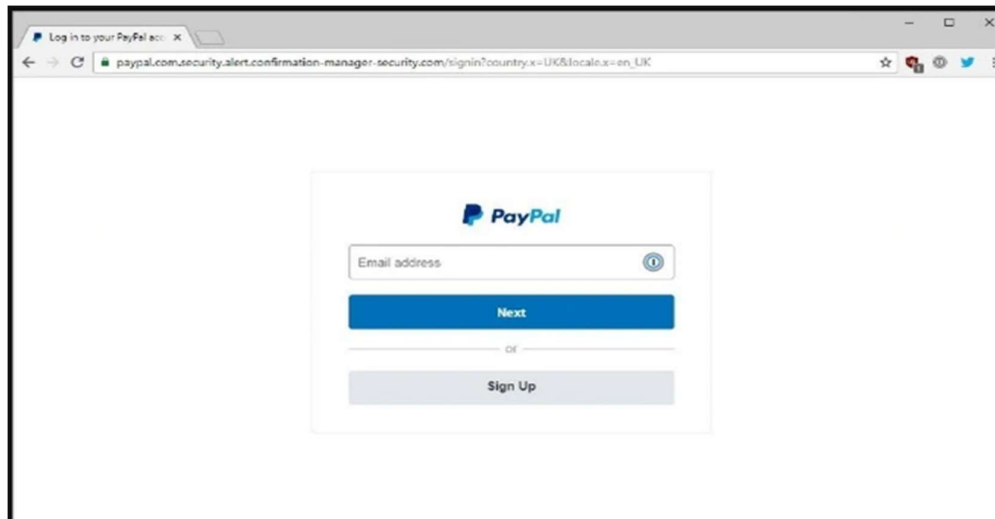
Hình 1.5. Tên miền không có chứng chỉ bảo mật SSL

Địa chỉ URL giả mạo: Trong trường hợp này, tin tặc đã tạo ra tên miền “paypal.com.confirm-manager-security.com” để đánh lừa thị giác của nạn nhân. Nếu không quan sát kỹ, nạn nhân có thể nghĩ đây là trang web của PayPal vì cụm từ “paypal.com” trong địa chỉ URL. Ví dụ về trang web giả mạo:



Hình 1.6. Trang thật nhìn rất chuyên nghiệp và đầy đủ thông tin

Còn dưới đây là một trang web fake dùng để ăn cắp thông tin cá nhân:

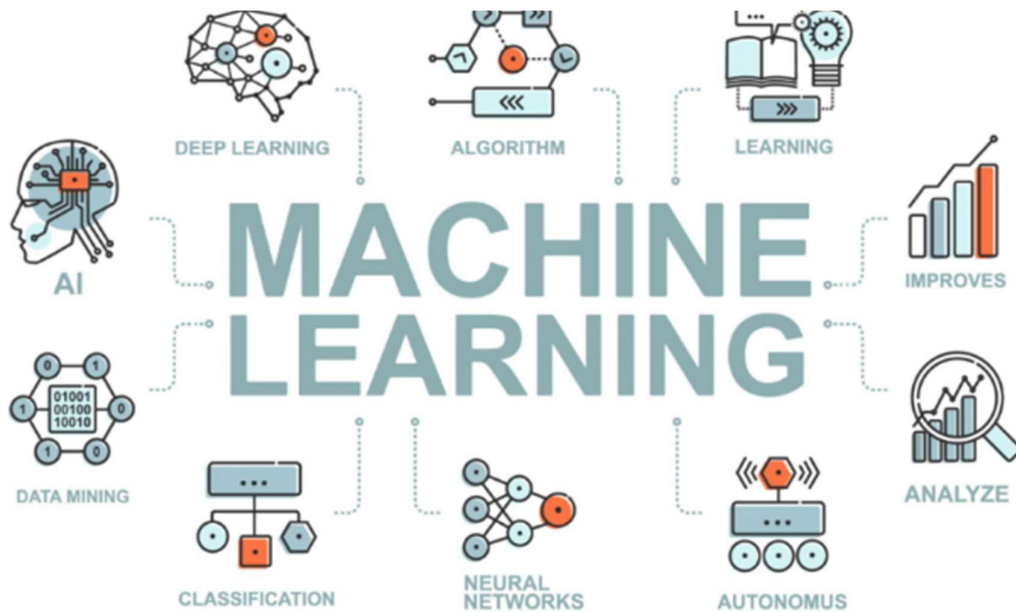


Hình 1.7. Trang giả mạo thì rất sơ sài

CHƯƠNG 2: THUẬT TOÁN RANDOM FOREST VÀ CÁC KHÁI NIỆM LIÊN QUAN

2.1 Machine Learning là gì?

Machine learning được hiểu đơn giản là công nghệ học máy, là một lĩnh vực con của công nghệ AI (Trí tuệ nhân tạo), liên quan đến việc nghiên cứu và xây dựng các kỹ thuật cho phép hệ thống “học” tự động thông qua các dữ liệu, từ đó có thể giải quyết những vấn đề được đưa ra. Các thuật toán trong phương pháp machine learning là các chương trình máy tính có khả năng học để hoàn thành các nhiệm vụ đưa ra và giúp cải thiện hiệu suất theo thời gian.



Hình 1.8. Các lĩnh vực sử dụng Học máy và Trí tuệ Nhân tạo

2.2. Phương pháp học trong machine learning

2.2.1. Phương pháp học có giám sát (Supervised Learning)

Đây là kỹ thuật học máy sử dụng cho các bài toán có sự phân lớp (Classification). Thuật toán được lựa chọn khi xây dựng bộ phân lớp này thường là: Máy vector hỗ trợ (Support Vector Machine – SVM); Cây quyết định (Decision Tree – DT); sử dụng mạng nơron (Neural Network – Net); dựa trên vector trọng tâm (Centroid– based vector); và tuyến tính bình phương nhỏ nhất (Linear Least Square Fit – LLSF).

2.2.2. Phương pháp học không giám sát (Unsupervised Learning)

Đây là kỹ thuật học máy dùng cho các bài toán phân cụm, gom cụm (Clustering). Có rất nhiều thuật toán học không giám sát đã xuất hiện và được phát triển nhằm phục vụ khai thác hiệu quả nguồn dữ liệu chưa được gán nhãn. Việc lựa chọn thuật toán tùy thuộc vào dữ liệu và mục đích sử dụng của từng dạng bài toán. Trong đó, các thuật toán được sử dụng thường xuyên là: k-means, HAC (Hierarchical Agglomerative Clustering), SOM (Self-Organizing Map), DBSCAN, FCM...

2.2.3. Phương pháp học bán giám sát (Semi-Supervised Learning)

Đây là một lớp của kỹ thuật học máy, sử dụng cả 2 dạng dữ liệu đã gán nhãn và chưa gán nhãn để hướng dẫn – điển hình như một lượng nhỏ dữ liệu có gán nhãn cùng với lượng lớn dữ liệu chưa gán nhãn. Các thuật toán thường xuyên được sử dụng có thể kể tới: thuật toán Cực đại kỳ vọng (EM – Expectation Maximization), SVM truyền dẫn (TSVM – Transductive Support Vector Machine), Self-training, Co-training và các phương pháp dựa trên đồ thị (graph-based).

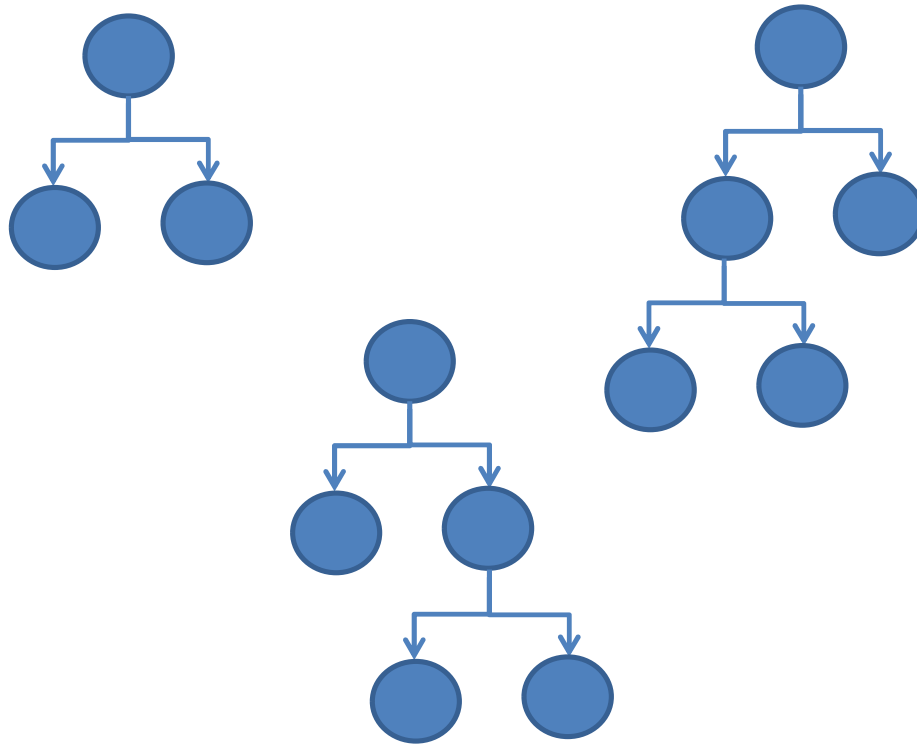
2.3. Random Forest là gì?

Thuật toán **Random Forest** là một phương pháp học máy mạnh mẽ dựa trên nguyên lý kết hợp nhiều cây quyết định để đưa ra dự đoán chính xác hơn. Tên gọi của thuật toán xuất phát từ hai yếu tố: "Random" (ngẫu nhiên) và "Forest" (rừng). Trong đó, "rừng" đại diện cho tập hợp của nhiều cây quyết định, mỗi cây quyết định này lại được xây dựng một cách ngẫu nhiên.

Quá trình xây dựng **Random Forest** bắt đầu từ việc tạo ra nhiều cây quyết định (Decision Tree) độc lập. Mỗi cây quyết định được huấn luyện với một tập con ngẫu nhiên của dữ liệu huấn luyện, có thể là các mẫu ngẫu nhiên hoặc lựa chọn ngẫu nhiên các đặc trưng (features) để phân chia tại mỗi nút trong cây. Điều này giúp giảm thiểu sự phụ thuộc giữa các cây quyết định, tạo ra sự đa dạng và cải thiện tính tổng quát của mô hình.

Kết quả dự đoán cuối cùng của **Random Forest** được xác định bằng cách tổng hợp các dự đoán từ tất cả các cây quyết định trong rừng. Đối với bài toán phân loại, kết quả dự đoán sẽ là sự "bỏ phiếu" của các cây quyết định, tức là lớp dự đoán được xác định dựa trên số lượng cây dự đoán cho mỗi lớp. Còn trong bài toán hồi quy, giá trị dự đoán cuối cùng là trung bình của tất cả các dự đoán từ các cây quyết định.

Nhờ vào tính ngẫu nhiên trong việc lựa chọn dữ liệu và đặc trưng, **Random Forest** có khả năng khắc phục được vấn đề quá khớp (overfitting) mà các cây quyết định đơn lẻ thường gặp phải, đồng thời nâng cao độ chính xác và độ ổn định trong dự đoán.

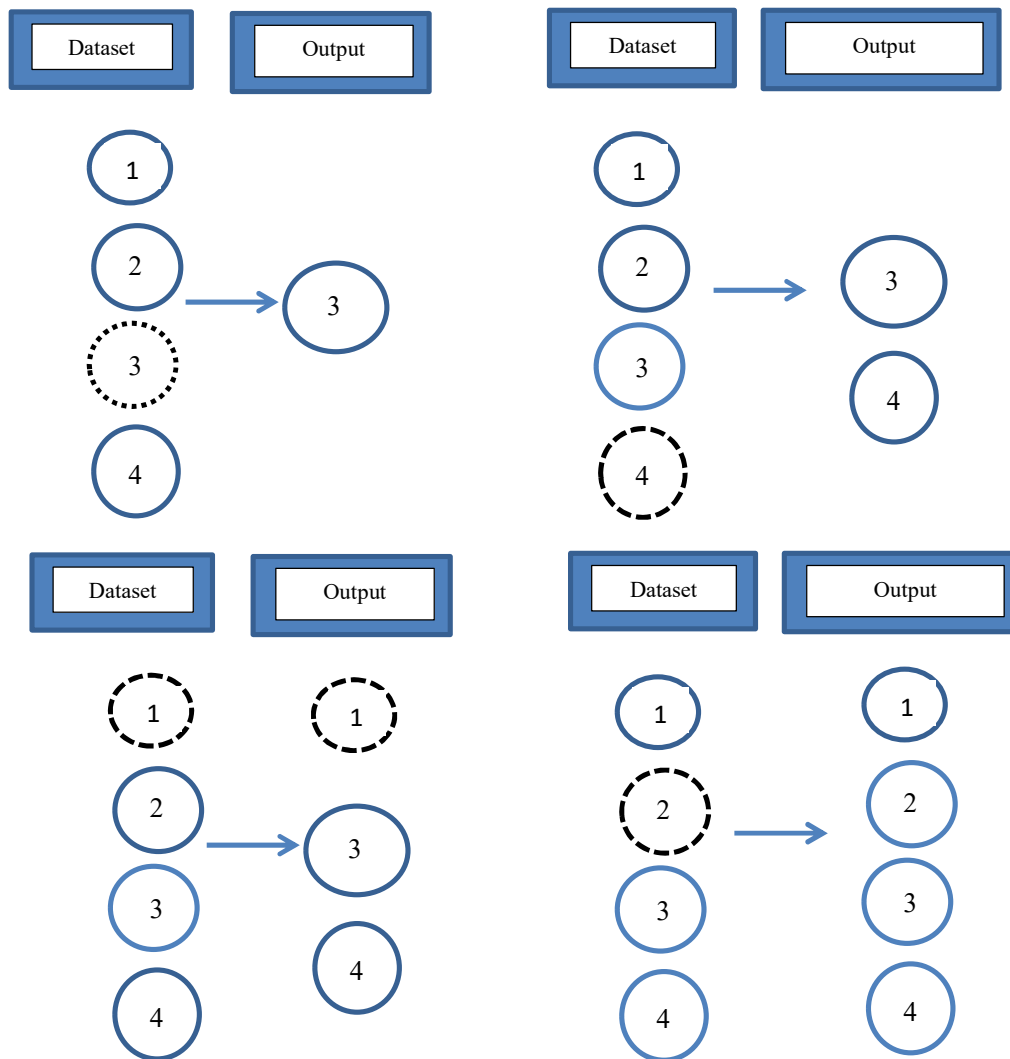


Hình 2.1. Random Forest được phát triển dựa trên Cây quyết định

Ví dụ thuật toán Random Forest có 4 cây quyết định, 3 cây dự đoán 1 và 1 cây dự đoán 0. Suy ra dự đoán cuối cùng sẽ là 1.

2.4. Xây dựng thuật toán Random Forest

Lấy ngẫu nhiên n dữ liệu từ bộ dữ liệu với kỹ thuật **Bootstrapping**, hay còn gọi là **random sampling with replacement**. Tức khi sample được 1 dữ liệu thì không bỏ dữ liệu đấy ra mà vẫn giữ lại trong tập dữ liệu ban đầu, rồi tiếp tục sample cho tới khi sample đủ n dữ liệu. Khi dùng kỹ thuật này thì tập n dữ liệu mới của mình có thể có những dữ liệu bị trùng nhau.



Hình 2.2. Cơ chế Random Sampling with Replacement

Sau khi sample được n dữ liệu từ bước 1 thì mình chọn ngẫu nhiên ở k thuộc tính ($k < n$). Giờ mình được bộ dữ liệu mới gồm n dữ liệu và mỗi dữ liệu có k thuộc tính.

Dùng thuật toán Decision Tree để xây dựng cây quyết định với bộ dữ liệu ở bước 2.

Do quá trình xây dựng mỗi cây quyết định đều có yếu tố ngẫu nhiên (random) nên kết quả là các cây quyết định trong thuật toán Random Forest có thể khác nhau.

Thuật toán Random Forest sẽ bao gồm nhiều cây quyết định, mỗi cây được xây dựng dùng thuật toán Decision Tree trên tập dữ liệu khác nhau và dùng tập thuộc tính khác nhau. Sau đó kết quả dự đoán của thuật toán Random Forest sẽ được tổng hợp từ các cây quyết định.

Khi dùng thuật toán Random Forest, mình hay để ý các thuộc tính như: số lượng cây quyết định sẽ xây dựng, số lượng thuộc tính dùng để xây dựng cây. Ngoài ra, vẫn có các thuộc tính của thuật

toán Decision Tree để xây dựng cây như độ sâu tối đa, số phần tử tối thiểu trong 1 node để có thể tách.

2.5. Nguyên lý hoạt động

Random Forest hoạt động bằng cách tạo ra một tập hợp lớn các cây quyết định. Dưới đây là các bước chính của quá trình này:

- Chọn ngẫu nhiên mẫu dữ liệu: Từ tập huấn luyện ban đầu, Random Forest chọn ra một số mẫu bằng phương pháp bootstrap (tạo mẫu có hoàn lại).
- Xây dựng cây quyết định: Mỗi mẫu ngẫu nhiên sẽ được sử dụng để xây dựng một cây quyết định. Tại mỗi nút của cây, Random Forest chỉ xem xét một tập con ngẫu nhiên của các thuộc tính (features) thay vì tất cả các thuộc tính. Điều này giúp giảm thiểu độ tương quan giữa các cây và tăng tính đa dạng của mô hình.
- Dự đoán: Đối với một mẫu mới, mỗi cây sẽ đưa ra một dự đoán. Random Forest sẽ tổng hợp tất cả các dự đoán từ các cây để đưa ra dự đoán cuối cùng (bằng cách phần lớn ý kiến cho phân loại hoặc tính trung bình cho hồi quy).

2.6. Ưu điểm của Random Forest

Khả năng kháng overfitting: Nhờ vào việc kết hợp nhiều cây quyết định, Random Forest có khả năng kháng overfitting cao hơn so với một cây quyết định đơn lẻ.

Độ chính xác cao: Random Forest thường mang đến độ chính xác cao nhờ vào việc kết hợp nhiều dự đoán.

Khả năng xử lý dữ liệu lớn: Với khả năng xử lý nhiều thuộc tính và mẫu đồng thời, Random Forest là sự lựa chọn tốt cho các bài toán lớn.

2.7. Nhược điểm của Random Forest

Khó giải thích: Mặc dù Random Forest có độ chính xác cao, nhưng việc giải thích mô hình cũng như cách mà nó đưa ra quyết định lại không phải là điều đơn giản.

Tốc độ chậm hơn: Khi làm việc với lượng lớn dữ liệu và số lượng cây quyết định nhiều, tốc độ tính toán có thể chậm hơn so với một số mô hình khác.

2.8. Ứng dụng của Random Forest

Random Forest được ứng dụng rộng rãi trong nhiều lĩnh vực khác nhau. Dưới đây là một số ví dụ tiêu biểu:

- **Phân loại email:** Random Forest có thể được sử dụng để phân loại email thành thư rác (spam) hoặc không phải thư rác (non-spam). Mô hình sẽ học từ các đặc điểm như từ trong tiêu đề, nội dung email, người gửi, v.v.

- **Dự đoán giá nhà:** Trong lĩnh vực bất động sản, Random Forest có thể được áp dụng để dự đoán giá nhà dựa trên các yếu tố như diện tích, số phòng ngủ, khu vực, và nhiều yếu tố khác.

- **Nhận diện cảm xúc:** The Random Forest can be used to predict the sentiment of the text (positive, negative, neutral) in various applications such as social media analysis or customer feedback.

2.9. Mã giả

```
import random # Nhập thư viện random để sử dụng các hàm ngẫu nhiên

# Hàm huấn luyện một cây quyết định (Decision Tree) - Giả lập
def train_decision_tree(data):
    # Chỉ là minh họa, cây quyết định sẽ được huấn luyện với một thuật toán cụ thể
    return "Cây huấn luyện từ dữ liệu"

# Hàm Random Forest
def random_forest(data, num_trees):
    # Khởi tạo một danh sách rừng (tập hợp các cây)
    forest = []

    for _ in range(num_trees):
        # Tạo mẫu ngẫu nhiên từ dữ liệu (có hoàn lại)
        sample = random.choices(data, k=len(data))

        # random.choices(data, k=len(data)) lấy ngẫu nhiên k phần tử từ dữ liệu `data` với khả năng chọn lại phần tử (có hoàn lại)

        # Huấn luyện một cây quyết định trên mẫu dữ liệu
        tree = train_decision_tree(sample)

        # Hàm `train_decision_tree` sẽ huấn luyện cây từ dữ liệu mẫu và trả về cây đã huấn luyện

        # Thêm cây vào rừng
        forest.append(tree)

    return forest

# Sau khi lặp qua hết số lượng cây, hàm trả về rừng (tập hợp các cây)

# Hàm dự đoán với Rừng (Random Forest)
def predict(forest, sample):
    predictions = []

    # Mỗi cây trong rừng sẽ dự đoán
    for tree in forest:
```



```

# Dự đoán từ cây (giả lập là cây trả về nhãn 'A')
prediction = "A"

predictions.append(prediction)

# Hàm `predict` sẽ lấy kết quả dự đoán từ cây (ở đây, giả lập rằng mỗi cây đều dự đoán 'A')
# Trả về nhãn xuất hiện nhiều nhất (Majority vote)
return max(set(predictions), key=predictions.count)

# `set(predictions)` loại bỏ các giá trị trùng lặp, còn `max(..., key=...)` trả về phần tử xuất hiện nhiều nhất trong predictions

# Ví dụ sử dụng
data = [1, 2, 3, 4, 5] # Dữ liệu huấn luyện giả lập
num_trees = 5 # Số lượng cây trong rừng

# Huấn luyện rừng
forest = random_forest(data, num_trees)

# Gọi hàm `random_forest` để huấn luyện 5 cây từ dữ liệu `[1, 2, 3, 4, 5]`

# Dự đoán mẫu mới
new_sample = 3 # Mẫu mới cần dự đoán
result = predict(forest, new_sample)

# Gọi hàm `predict` để dự đoán kết quả từ rừng đã huấn luyện cho mẫu mới (3)
print(f"Dự đoán cho mẫu {new_sample}: {result}")

# In kết quả dự đoán

```

import random:

Dòng này nhập thư viện random trong Python để sử dụng các hàm ngẫu nhiên. Chúng ta cần hàm này để chọn mẫu ngẫu nhiên từ tập dữ liệu.

def train_decision_tree(data):

Định nghĩa hàm train_decision_tree, một hàm giả lập cho việc huấn luyện cây quyết định. Trong thực tế, cây quyết định sẽ được huấn luyện từ dữ liệu data bằng cách phân chia và lựa chọn thuộc tính tốt nhất tại mỗi nút. Tuy nhiên, ở đây chúng ta chỉ giả lập việc huấn luyện và trả về một chuỗi đơn giản.

forest = []:

Tạo một danh sách rỗng có tên forest. Đây sẽ là nơi chứa các cây quyết định được huấn luyện từ dữ liệu mẫu.

for _ in range(num_trees):

Vòng lặp này sẽ chạy num_trees lần, tức là số lượng cây quyết định mà bạn muốn tạo trong rừng. Mỗi lần lặp, chúng ta sẽ tạo một cây mới và huấn luyện nó.

sample = random.choices(data, k=len(data)):

Sử dụng random.choices() để chọn ngẫu nhiên các mẫu từ dữ liệu data với kích thước là len(data). k=len(data) có nghĩa là chúng ta lấy một mẫu có kích thước bằng chính số lượng mẫu trong dữ liệu gốc. "Có hoàn lại" có nghĩa là mẫu có thể trùng nhau, tức là một phần tử có thể được chọn nhiều lần.

tree = train_decision_tree(sample):

Dữ liệu mẫu sample sẽ được đưa vào hàm train_decision_tree để huấn luyện một cây quyết định. Hàm này sẽ trả về một cây đã huấn luyện (ở đây chỉ là một chuỗi giả lập).

forest.append(tree):

Sau khi huấn luyện xong cây, chúng ta thêm cây vào danh sách forest để lưu lại.

return forest:

Sau khi huấn luyện xong tất cả các cây, hàm random_forest sẽ trả về rừng, tức là danh sách chứa tất cả các cây đã huấn luyện.

def predict(forest, sample):

Định nghĩa hàm predict để dự đoán đầu ra cho mẫu mới từ rừng cây. Hàm này nhận vào forest (rừng cây) và một mẫu sample cần dự đoán.

predictions = []:

Tạo danh sách rỗng predictions để lưu trữ các kết quả dự đoán từ mỗi cây trong rừng.

for tree in forest::

Lặp qua mỗi cây trong rừng để lấy kết quả dự đoán từ mỗi cây.

prediction = "A":

Mỗi cây dự đoán một nhãn (ở đây giả lập là nhãn "A"). Trong thực tế, mỗi cây sẽ dự đoán một giá trị khác nhau dựa trên cấu trúc cây và dữ liệu.

predictions.append(prediction):

Kết quả dự đoán từ cây được thêm vào danh sách predictions.

return max(set(predictions), key=predictions.count):

Dòng này thực hiện bước **majority vote**. Chúng ta chuyển predictions thành một tập hợp (set) để loại bỏ các giá trị trùng lặp, sau đó tìm nhãn xuất hiện nhiều nhất bằng cách sử dụng hàm max(). Tham số key=predictions.count đảm bảo rằng chúng ta chọn nhãn xuất hiện nhiều nhất trong danh sách predictions.

data = [1, 2, 3, 4, 5]:

Dữ liệu huấn luyện giả lập. Đây là dữ liệu mà rừng cây sẽ được huấn luyện trên đó.

num_trees = 5:

Số lượng cây trong rừng. Chúng ta tạo 5 cây quyết định trong rừng.

forest = random_forest(data, num_trees):

Gọi hàm random_forest với dữ liệu huấn luyện data và số lượng cây num_trees để huấn luyện một rừng.

new_sample = 3:

Mẫu mới cần dự đoán. Đây là mẫu mà chúng ta muốn dự đoán nhãn.

result = predict(forest, new_sample):

Gọi hàm predict để dự đoán kết quả từ rừng forest cho mẫu mới new_sample.

print(f'Dự đoán cho mẫu {new_sample}: {result}')

In kết quả dự đoán cho mẫu mới.

CHƯƠNG 3: TRIỂN KHAI & THỰC NGHIỆM

3.1. Chuẩn bị môi trường

3.1.1. Hệ điều hành và công cụ

- Hệ điều hành: Windows 11.
- RAM: 8GB.
- Dung lượng: 500GB.
- Công cụ: Jupyter Notebook của Anaconda.
- Ngôn ngữ: Python.

3.1.2. Thư viện

Cài đặt các thư viện cần thiết, bao gồm:

- pandas và numpy: giúp hỗ trợ xử lý và phân tích dữ liệu.
- matplotlib và seaborn: giúp trực quan hóa dữ liệu và vẽ biểu đồ (ma trận nhầm lẫn).
- nltk: giúp tiền xử lý văn bản, bao gồm stemming và loại bỏ stopwords.
- scikit-learn: giúp cung cấp các công cụ học máy như TfidfVectorizer để vector hóa văn bản, RandomForestClassifier để huấn luyện mô hình.
- re: giúp xử lý biểu thức.
- os: giúp quản lý file.
- email: giúp phân tích email.
- pickle và joblib: giúp lưu trữ dữ liệu và mô hình.
- urllib.parse: giúp phân tích URL.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import nltk
import re
import os
import email
import pickle
import joblib
from urllib.parse import urlparse
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, recall_score
from sklearn.metrics import roc_curve, auc
import warnings
import time
warnings.filterwarnings('ignore')

# Thiết lập hiển thị và tải stopwords
pd.set_option('display.max_colwidth', 200)
plt.style.use('ggplot')
nltk.download('stopwords')
```

Hình 3.1. Khởi tạo thư viện

3.1.3. Bộ dữ liệu dataset

Nhóm nghiên cứu sử dụng 2 bộ dữ liệu từ SpamAssassin và Kaggle, trong đó:

- Bộ dữ liệu SpamAssassin chứa 23516 emails, trong đó có 12324 email thường, 11192 email phishing được định dạng thành file text.
- Bộ dữ liệu Kaggle chứa 193752 emails, trong đó có 102153 email thường, 91599 email phishing được định dạng thành file csv.
- Cả 2 bộ dữ liệu đều được mô hình dùng 80% để train và 20% để test. Từ bộ dữ liệu, ta có thể trích xuất các đặc trưng của một email phishing gồm:

- + Tiêu đề.
- + Người gửi.
- + Nội dung.
- + URL đáng ngờ.

Bảng 1: Số lượng email của 2 bộ dữ liệu SpamAssassin và Kaggle

	SpamAssassin	Kaggle
Email Ham	12324	11192
Email Spam	102153	91599
Email Train	18812	155011
Email Test	4704	38751

3.2. Triển khai và thực nghiệm

- Bước 1: Tiền xử lý dữ liệu.

Quá trình tiền xử lý dữ liệu là bước quan trọng để chuẩn bị dữ liệu thô từ các tập email (bao gồm cả email phishing và email thường) thành dạng phù hợp cho huấn luyện mô hình. Các bước tiền xử lý bao gồm:

- + Loại bỏ stopwords: Các từ không mang ý nghĩa đặc trưng như "the", "is", "and" được loại bỏ để giảm nhiễu và tập trung vào từ khóa quan trọng. Thư viện NLTK được sử dụng để tải danh sách stop words tiếng Anh.

- + Stemming: Chuyển các từ về dạng gốc (ví dụ: "running" thành "run") bằng thuật toán Porter Stemmer để chuẩn hóa các biến thể từ. Điều này giúp tăng tính nhất quán trong phân tích văn bản.

+ Chuẩn hóa văn bản: Văn bản được chuyển thành chữ thường, loại bỏ ký tự đặc biệt (như dấu câu, số). Quy trình này đảm bảo dữ liệu đồng nhất trước khi vector hóa.

+ Vector hóa (TF-IDF): Dữ liệu văn bản được chuyển thành ma trận số bằng phương pháp TF-IDF (Term Frequency-Inverse Document Frequency) để thể hiện mức độ quan trọng của từ trong từng email. Các đặc trưng như nội dung, tiêu đề, và người gửi được vector hóa riêng biệt với giới hạn `max_features` (ví dụ: 3000 cho nội dung, 500 cho tiêu đề, 300 cho người gửi) để giảm kích thước ma trận và tránh lỗi bộ nhớ.

```
def preprocess_text(text):
    if not isinstance(text, str):
        return ""
    #1. Chuyển thành chữ thường
    text = text.lower()
    #2. Loại bỏ các ký tự đặc biệt và số
    text = re.sub(r'^a-zA-Z\s', '', text)
    #3. Stemming
    stemmer = PorterStemmer()
    words = text.split()
    words = [stemmer.stem(word) for word in words]
    #4. Loại bỏ stopwords
    stop_words = set(stopwords.words('english'))
    words = [stemmer.stem(word) for word in words]

    return ' '.join(words)
```

Hình 3.2. Tiến hành tiền xử lý dữ liệu

Hàm `preprocess_text(text)` được sử dụng để thực hiện tiền xử lý văn bản nhằm chuẩn bị dữ liệu đầu vào cho mô hình học máy. Trước tiên, hàm kiểm tra kiểu dữ liệu đầu vào, nếu không phải là chuỗi thì sẽ trả về `None` để tránh lỗi khi xử lý. Sau đó, toàn bộ văn bản được chuyển thành chữ thường để chuẩn hóa. Tiếp theo, các ký tự đặc biệt và chữ số sẽ bị loại bỏ thông qua biểu thức chính quy, chỉ giữ lại chữ cái và khoảng trắng. Văn bản sau đó được chia thành các từ riêng lẻ và áp dụng kỹ thuật stemming bằng công cụ `PorterStemmer` để rút gọn từ về gốc nhằm giảm số lượng từ cần xử lý. Kế đến, các từ dừng (stopwords) trong tiếng Anh như “the”, “is”, “in”,... sẽ bị loại bỏ vì chúng không mang nhiều ý nghĩa trong việc phân tích nội dung. Cuối cùng, các từ còn lại được nối lại thành một chuỗi và trả về.

```
# Chuyển đổi sang TF-IDF và kết hợp đặc trưng
print("\n⚡ Đang chuyển đổi dữ liệu...")
tfidf_vectorizer_content = TfidfVectorizer(max_features=3000, stop_words='english', ngram_range=(1, 2))
tfidf_vectorizer_subject = TfidfVectorizer(max_features=500, stop_words='english')
tfidf_vectorizer_from = TfidfVectorizer(max_features=300, stop_words='english')

contents = [f['content'] for f in feature_list]
subjects = [f['subject'] for f in feature_list]
froms = [f['from'] for f in feature_list]
urls = np.array([f['has_suspicious_url'] for f in feature_list]).reshape(-1, 1)

X_content = tfidf_vectorizer_content.fit_transform(contents)
X_subject = tfidf_vectorizer_subject.fit_transform(subjects)
X_from = tfidf_vectorizer_from.fit_transform(froms)
X = hstack([X_content, X_subject, X_from, urls.astype(np.float32)])
y = np.array(labels)
```

Hình 3.3. Tiến hành Vector hóa dữ liệu

Trong đoạn mã này, dữ liệu văn bản được chuyển đổi sang dạng vector bằng phương pháp TF-IDF (Term Frequency – Inverse Document Frequency), nhằm biểu diễn các đặc trưng của email dưới dạng số phức vụ cho việc huấn luyện mô hình. Ba trường văn bản quan trọng được xử lý riêng biệt là *content* (nội dung email), *subject* (tiêu đề), và *from* (địa chỉ người gửi). Mỗi trường được áp dụng một bộ biến đổi TF-IDF riêng với các tham số khác nhau: *content* sử dụng tối đa 3000 đặc trưng và hỗ trợ cả unigram và bigram (1-2 ngram), *subject* tối đa 500 đặc trưng, và *from* tối đa 300 đặc trưng. Ngoài ra, đặc trưng *has_suspicious_url* – biểu thị email có chứa URL đáng ngờ hay không – được lấy ra và định hình lại để có thể kết hợp với các đặc trưng văn bản. Sau khi xử lý, các đặc trưng này được kết hợp lại bằng hàm *hstack* để tạo ra ma trận đặc trưng *X*, là đầu vào chính cho mô hình học máy. Nhãn (label) tương ứng của từng email được lưu trong biến *y*.

- Bước 2: Gán nhãn và chia tập train, test.

Dữ liệu được gán nhãn dựa trên nguồn gốc của email:

- + Email trong thư mục spam được gán nhãn 1 (email phishing).
- + Email trong thư mục ham được gán nhãn 0 (email thường). Quá trình này được thực hiện tự động khi tải dữ liệu từ thư mục email.

Sau khi gán nhãn, dữ liệu được chia thành hai tập để huấn luyện và kiểm tra mô hình:

- + Tập huấn luyện (train): 80% dữ liệu, dùng để huấn luyện mô hình.
- + Tập kiểm tra (test): 20% dữ liệu, dùng để đánh giá hiệu suất mô hình.

Phương pháp chia tập sử dụng hàm `train_test_split` từ `scikit-learn` với tham số `stratify=y` để đảm bảo tỷ lệ email phishing và không phishing trong cả hai tập tương tự như dữ liệu gốc.

Bảng 2: Số lượng email SpamAssassin dùng để train, test

	Email Phishing	Email thường	Tổng
Training	8953	9859	18812
Testing	2239	2465	4704

Bảng 3: Số lượng email Kaggle dùng để train, test

	Email Phishing	Email thường	Tổng
Training	73279	81722	155001
Testing	18320	20431	38751

```
def load_emails_from_directory(directory):
    if not os.path.exists(directory):
        raise FileNotFoundError(f"Thư mục {directory} không tồn tại!")

    feature_list = []
    labels = []

    for label in ['spam', 'ham']:
        label_path = os.path.join(directory, label)
        if not os.path.exists(label_path):
            continue

        print(f"Đang xử lý {label}...")
        for filename in os.listdir(label_path):
            file_path = os.path.join(label_path, filename)

            try:
                with open(file_path, 'r', encoding='latin-1') as f:
                    msg = email.message_from_file(f)
                    features = extract_email_features(msg)

                    if features['content']: # Chỉ thêm nếu có nội dung
                        feature_list.append(features)
                        labels.append(1 if label == 'spam' else 0)
            except Exception as e:
                print(f"Lỗi khi xử lý {filename}: {str(e)}")

    return feature_list, labels
```

Hình 3.4. Tiến hành Gán nhãn cho các dữ liệu

Hàm `load_emails_from_directory(directory)` được sử dụng để đọc và phân loại các email từ một thư mục chứa dữ liệu, với hai loại nhãn là *spam* và *ham*. Trước tiên, hàm kiểm tra sự tồn tại của thư mục đầu vào; nếu không tồn tại, hàm sẽ báo lỗi và dừng thực thi. Sau đó, với mỗi nhãn (*spam* hoặc *ham*), chương trình xây dựng đường dẫn tương ứng và kiểm tra sự tồn tại của thư mục con đó. Nếu thư mục tồn tại, chương trình sẽ duyệt qua từng tệp tin trong thư mục, đọc nội dung email bằng

thư viện *email*, và trích xuất đặc trưng bằng hàm *extract_email_features(msg)*. Nếu email có nội dung (*content*) thì đặc trưng sẽ được thêm vào danh sách *feature_list*, đồng thời nhãn tương ứng cũng được thêm vào danh sách *labels* (với *spam* là 1 và *ham* là 0). Trong trường hợp xảy ra lỗi khi xử lý tệp tin, chương trình sẽ in ra thông báo lỗi nhưng không dừng toàn bộ quá trình. Cuối cùng, hàm trả về hai danh sách: một chứa đặc trưng của email và một chứa nhãn tương ứng.

```
# Chia tập train/test
X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42,
    stratify=y
)
```

Hình 3.5. Tiến hành chia dữ liệu để huấn luyện

Đoạn mã này sử dụng hàm *train_test_split* từ thư viện *sklearn.model_selection* để chia dữ liệu thành hai tập: tập huấn luyện (*X_train, y_train*) và tập kiểm tra (*X_test, y_test*). Tỷ lệ chia là 80% dữ liệu cho huấn luyện và 20% cho kiểm tra, được xác định thông qua tham số *test_size=0.2*. Tham số *random_state=42* được sử dụng để đảm bảo kết quả chia dữ liệu là nhất quán mỗi khi chạy lại chương trình. Ngoài ra, tham số *stratify=y* giúp đảm bảo rằng tỉ lệ giữa các nhãn (*spam* và *ham*) trong tập huấn luyện và kiểm tra vẫn được giữ nguyên như trong tập dữ liệu gốc, giúp mô hình học hiệu quả và đánh giá công bằng hơn.

Bước 3: Khởi tạo và huấn luyện mô hình Random Forest.

```
# Khởi tạo mô hình Random Forest
print("\n🚀 Khởi tạo mô hình Random Forest...")
rf_model = RandomForestClassifier(
    n_estimators=150,
    max_depth=30,
    min_samples_split=5,
    class_weight='balanced',
    random_state=42,
    n_jobs=-1
)

# Huấn Luyện mô hình
print("🔄 Đang huấn luyện mô hình Random Forest...")
start_time = time.time()
rf_model.fit(X_train, y_train)
training_time = time.time() - start_time
print(f"✅ Hoàn tất huấn luyện! Thời gian huấn luyện: {training_time:.2f} giây")
```

🚀 Khởi tạo mô hình Random Forest...
🔄 Đang huấn luyện mô hình Random Forest...
✅ Hoàn tất huấn luyện! Thời gian huấn luyện: 1.16 giây

Hình 3.6. Tiến hành khởi tạo và huấn luyện theo tập dữ liệu đã chia với bộ dữ liệu *SpamAssassin*

```
🌲 Khởi tạo mô hình Random Forest...
🚀 Đang huấn luyện mô hình Random Forest...
✅ Hoàn tất huấn luyện! Thời gian huấn luyện: 75.25 giây
```

Hình 3.7. Tiến hành khởi tạo và huấn luyện theo tập dữ liệu đã chia với bộ dữ liệu Kaggle

Mô hình Random Forest được khởi tạo bằng cách sử dụng lớp RandomForestClassifier từ thư viện sklearn.ensemble. Một số tham số quan trọng được thiết lập như sau: n_estimators=150 xác định số lượng cây trong rừng; max_depth=30 giới hạn độ sâu tối đa của mỗi cây nhằm kiểm soát độ phức tạp; min_samples_split=5 quy định số lượng mẫu tối thiểu để chia một nút; class_weight='balanced' được sử dụng để tự động cân bằng trọng số giữa các lớp, giúp mô hình hoạt động hiệu quả với dữ liệu mất cân bằng; random_state=42 đảm bảo kết quả có thể tái lập; và n_jobs=-1 cho phép sử dụng toàn bộ tài nguyên CPU để tăng tốc độ huấn luyện. Sau khi khởi tạo, mô hình được huấn luyện với tập dữ liệu X_train và y_train. Thời gian huấn luyện được đo bằng cách lấy hiệu giữa thời điểm bắt đầu và thời điểm kết thúc quá trình huấn luyện. Cuối cùng, chương trình in ra thông báo hoàn tất cùng với thời gian huấn luyện, ví dụ như: “Hoàn tất huấn luyện! Thời gian huấn luyện: 1.16 giây”.

- Bước 4: Triển khai và đánh giá mô hình Random Forest

```
# Đánh giá mô hình trên tập test
print("\n🌲 Đang đánh giá mô hình trên tập test...")
y_pred = rf_model.predict(X_test)

# Tính các chỉ số đánh giá
accuracy = accuracy_score(y_test, y_pred)
recall_spam = recall_score(y_test, y_pred, pos_label=1)
recall_ham = recall_score(y_test, y_pred, pos_label=0)
tn, fp, fn, tp = confusion_matrix(y_test, y_pred).ravel()
specificity = tn / (tn + fp)

print(f"📊 Độ chính xác: {accuracy*100:.2f}%")
print(f"📊 Độ nhạy (Spam): {recall_spam*100:.2f}%")
print(f"📊 Độ nhạy (Ham): {recall_ham*100:.2f}%")
print(f"📊 Độ đặc hiệu (Spam): {specificity*100:.2f}%")

# Báo cáo chi tiết
print("\n📄 Báo cáo phân loại:")
print(classification_report(y_test, y_pred, target_names=['Ham', 'Spam']))

# Vẽ ma trận nhầm lẫn
plt.figure(figsize=(8,6))
conf_mat = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_mat, annot=True, fmt='d', cmap='Blues',
            xticklabels=['Ham', 'Spam'],
            yticklabels=['Ham', 'Spam'])
plt.title('MA TRẬN NHẦM LẤN', pad=20)
plt.xlabel('Dự đoán')
plt.ylabel('Thực tế')
plt.show()
```

Hình 3.8. Đánh giá mô hình trên tập dữ liệu test

Mô hình Random Forest được đánh giá trên tập kiểm tra (X_{test}, y_{test}). Dự đoán của mô hình được lưu trong biến y_{pred} thông qua phương thức *predict*. Tiếp theo, các chỉ số đánh giá hiệu suất mô hình được tính toán bao gồm: độ chính xác (*accuracy*), độ nhạy đối với email spam (*recall_spam* với $pos_label=1$), độ nhạy đối với email hợp lệ - ham (*recall_ham* với $pos_label=0$), và độ đặc hiệu (*specificity*) – được tính bằng công thức $tn / (tn + fp)$ từ ma trận nhầm lẫn (*confusion_matrix*). Kết quả các chỉ số được in ra màn hình dưới dạng phần trăm. Đồng thời, một báo cáo chi tiết (*classification_report*) cũng được hiển thị, cho biết các thông số như Precision, Recall và F1-score cho từng lớp. Cuối cùng, ma trận nhầm lẫn được trực quan hóa bằng thư viện Seaborn dưới dạng biểu đồ nhiệt (heatmap), giúp người đọc dễ dàng nhận biết mô hình đã phân loại đúng – sai bao nhiêu email thuộc các lớp *Ham* và *Spam*.

- Độ chính xác: Tỷ lệ dự đoán đúng (ham và spam) so với tổng các email trong tập test.

$$DoChinhXac = \frac{TN + TP}{TongSoEmailMau}$$

Trong đó:

- + TP (True Positive): Số Spam được dự đoán đúng là Spam.
- + TN (True Negative): Số Ham được dự đoán đúng là Ham.
- Độ nhạy: Tỷ lệ các mẫu thực sự thuộc lớp (Spam hoặc Ham) mà mô hình dự đoán đúng

$$DoNhay = \frac{TP}{TP + FN}$$

Trong đó:

- + FN (False Negative): Số Spam bị dự đoán nhầm là Ham.
- Độ đặc hiệu (Specificity): Tỷ lệ các mẫu không thuộc lớp (ở đây là Spam) mà mô hình dự đoán đúng là không thuộc lớp đó (tức là dự đoán đúng Ham).

$$DoDacHieu = \frac{TN}{TN + FP}$$

Trong đó:

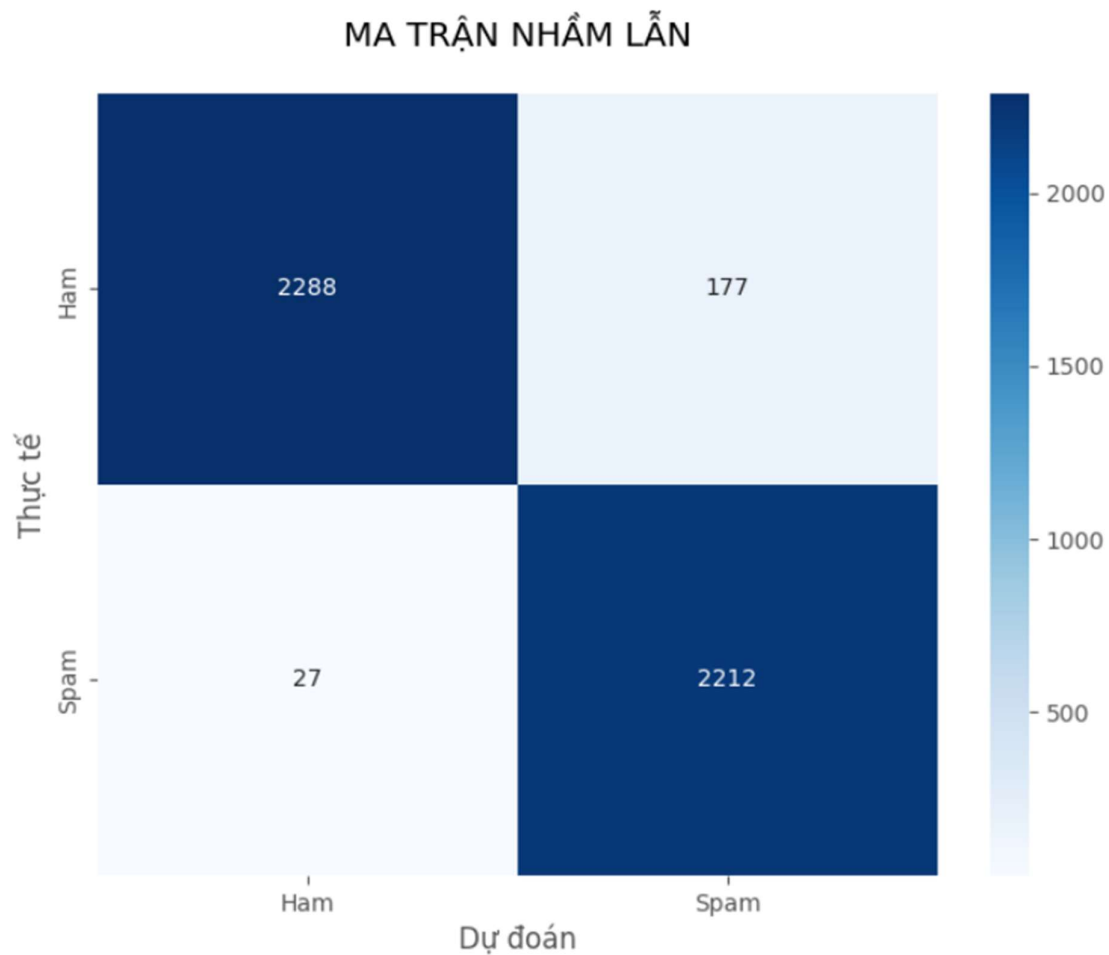
- + FP (False Positive): Số Ham bị dự đoán nhầm là Spam

Ma trận nhầm lẫn dùng để đánh giá hiệu suất của mô hình phân loại, đặc biệt trong bài toán phân loại nhị phân (như phân loại email thành Ham và Spam). Nó cho thấy số lượng dự đoán đúng và sai của mô hình trên từng lớp, giúp phân tích chi tiết các lỗi phân loại.

- Hàng: Đại diện cho giá trị thực tế.

- Cột: Đại diện cho giá trị dự đoán

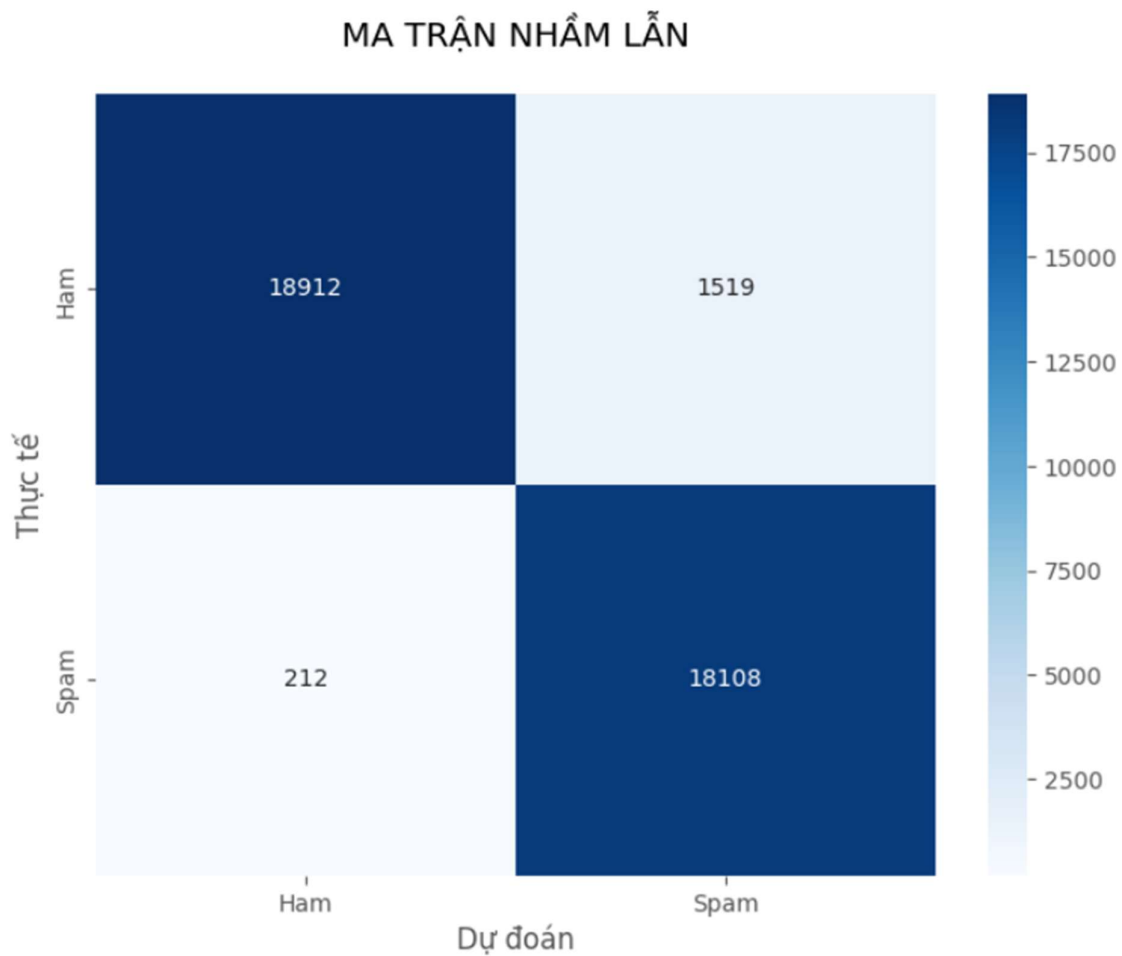
Ma trận nhầm lẫn của bộ dữ liệu SpamAssassin



Hình 3.9. Ma trận nhầm lẫn của bộ dữ liệu SpamAssassin

- Số lượng email thực tế là ham và được dự đoán đúng là ham: 2288 email.
- Số lượng email thực tế là ham nhưng được dự đoán là spam: 177 email.
- Số lượng email thực tế là spam nhưng được dự đoán là ham: 27 email.
- Số lượng email thực tế là spam và được dự đoán đúng là spam: 2212 email.

Ma trận nhầm lẫn của bộ dữ liệu Kaggle



Hình 3.10. Ma trận nhầm lẫn của bộ dữ liệu Kaggle

- Số lượng email thực tế là ham và được dự đoán đúng là ham: 18912 email.
- Số lượng email thực tế là ham nhưng được dự đoán là spam: 1519 email.
- Số lượng email thực tế là spam nhưng được dự đoán là ham: 212 email.
- Số lượng email thực tế là spam và được dự đoán đúng là spam: 18108 email.

📊 Đang đánh giá mô hình trên tập test...

- 🔍 Độ chính xác: 95.58%
- 🔍 Độ nhạy (Spam): 98.75%
- 🔍 Độ nhạy (Ham): 92.70%
- 🔍 Độ đặc hiệu (Spam): 92.70%

📄 Báo cáo phân loại:

	precision	recall	f1-score	support
Ham	0.99	0.93	0.96	2465
Spam	0.92	0.99	0.96	2239
accuracy			0.96	4704
macro avg	0.96	0.96	0.96	4704
weighted avg	0.96	0.96	0.96	4704

Hình 3.11. Đánh giá mô hình Random Forest cho bộ dữ liệu SpamAssassin

📊 Đang đánh giá mô hình trên tập test...

- 🔍 Độ chính xác: 95.53%
- 🔍 Độ nhạy (Spam): 98.84%
- 🔍 Độ nhạy (Ham): 92.57%
- 🔍 Độ đặc hiệu (Spam): 92.57%

📄 Báo cáo phân loại:

	precision	recall	f1-score	support
Ham	0.99	0.93	0.96	20431
Spam	0.92	0.99	0.95	18320
accuracy			0.96	38751
macro avg	0.96	0.96	0.96	38751
weighted avg	0.96	0.96	0.96	38751

Hình 3.12. Đánh giá mô hình Random Forest cho bộ dữ liệu Kaggle

Bảng 4: Bảng so sánh độ chính xác của 2 bộ dữ liệu SpamAssassin, Kaggle

	Độ chính xác	Độ nhạy (Ham)	Độ nhạy (Spam)	Độ đặc hiệu	Thời gian huấn luyện
SpamAssassin	95.58%	92.70%	98.75%	92.70%	1.16s
Kaggle	95.53%	92.57%	98.84%	92.57%	75.25s

3.3. Kiểm thử

```
# Hàm nhận diện
def detect_phishing(email_text):
    """Nhận diện email spam/phishing.

    Args:
        email_text (str): Nội dung email dạng văn bản.
    """
    msg = email.message_from_string(email_text)
    features = extract_email_features(msg)

    content_vec = tfidf_vectorizer_content.transform([features['content']])
    subject_vec = tfidf_vectorizer_subject.transform([features['subject']])
    from_vec = tfidf_vectorizer_from.transform([features['from']])
    url_vec = np.array([features['has_suspicious_url']]).reshape(-1, 1)
```

Hình 3.13. Hàm nhận diện email phishing (1)

```
text_vector = np.hstack((content_vec.toarray(), subject_vec.toarray(), from_vec.toarray(), url_vec))
prediction = rf_model.predict(text_vector)[0]
proba = rf_model.predict_proba(text_vector)[0][1]

result = "SPAM (lừa đảo)" if prediction == 1 else "HAM (hợp lệ)"
print(f"\n🎁 Kết quả: {result}")
print(f"📊 Độ tin cậy: {proba*100:.2f}%")
print("\n🔍 Các đặc trưng quan trọng ảnh hưởng đến quyết định:")

feature_names = (
    tfidf_vectorizer_content.get_feature_names_out().tolist() +
    tfidf_vectorizer_subject.get_feature_names_out().tolist() +
    tfidf_vectorizer_from.get_feature_names_out().tolist() +
    ['has_suspicious_url']
)
coef = rf_model.feature_importances_
top_indices = np.argsort(coef)[-1:][:10]
for i in top_indices:
    if text_vector[:, i].sum() > 0:
        print(f"- {feature_names[i]} (importance: {coef[i]:.4f})")
```

Hình 3.14. Hàm nhận diện email phishing (2)

3.3.1. Kiểm thử với bộ dữ liệu SpamAssassin

Trường hợp 1: Kiểm thử với email Spam cho bộ dữ liệu SpamAssassin

```
# Test demo
test_email = """
From: account-security@microsoft-support.com
Subject: Action Required: Unusual Sign-in Attempt Detected

Dear User,

We noticed an unusual sign-in attempt to your Microsoft account from a new device.

If this was you, please verify your account immediately:
https://microsoft-verify-login.com

If you do not verify within 24 hours, your account will be locked for security reasons.

Thanks,
Microsoft Security Team

"""
detect_phishing(test_email)
```

Hình 3.15. Kiểm thử với email spam cho bộ dữ liệu SpamAssassin

Kết quả

📧 Kết quả: SPAM (lừa đảo)
 📊 Độ tin cậy: 51.38%

🚩 Các đặc trưng quan trọng ảnh hưởng đến quyết định:
 - thank (importance: 0.0145)

Hình 3.16. Kết quả email spam với bộ dữ liệu SpamAssassin

Trường hợp 2: Kiểm thử với email thường (ham) cho bộ dữ liệu SpamAssassin

```
# Test demo
test_email = """
From: linh.tran@company.com
Subject: Meeting Reminder - Project Alpha

Hi team,

This is a reminder that we will have our next Project Alpha meeting tomorrow at 10:00 AM in Meeting Room B.

Agenda:
- Progress update
- Risk discussion
- Next steps

Please let me know if you have any additions.

Thanks,
Linh

"""
detect_phishing(test_email)
```

Hình 3.17. Kiểm thử với email ham cho bộ dữ liệu SpamAssassin

Kết quả

🏠 Kết quả: HAM (hợp lệ)
📊 Độ tin cậy: 8.88%

📌 Các đặc trưng quan trọng ảnh hưởng đến quyết định:
- thank (importance: 0.0145)

Hình 3.18. Kết quả email ham với bộ dữ liệu SpamAssassin

Trường hợp 3: Kiểm thử với email spam nhưng mô hình nhận định là email thường (ham) cho bộ dữ liệu SpamAssassin

```
# Test demo
test_email = """
Subject: Account Statement Available

From: notifications@yourbank.com

Dear Customer,

Your monthly account statement for April 2025 is now available.
Please log in to your account to view it: http://yourbank.com/login/statement.
If you have any questions, contact us at 1-800-555-1234.

Sincerely,

YourBank Team
"""
detect_phishing(test_email)
```

Hình 3.19. Kiểm thử với email spam nhưng mô hình nhận định là ham cho bộ dữ liệu SpamAssassin

Kết quả

🏠 Kết quả: HAM (hợp lệ)
📊 Độ tin cậy: 28.85%

📌 Các đặc trưng quan trọng ảnh hưởng đến quyết định:
- question (importance: 0.0121)

Hình 3.20. Kết quả với email spam nhưng mô hình nhận định là ham cho bộ dữ liệu SpamAssassin

Trường hợp 4: Kiểm thử với email ham nhưng mô hình nhận định là email spam cho bộ dữ liệu SpamAssassin

```
# Test demo
test_email = """
Subject: Please Update Your Employee Profile

From: hr@company.com

Hello [Employee],

We're updating our employee database. Please update your profile by clicking here: http://hr.company.com/update-profile.
This is required to ensure your information is current.

Best regards,

HR Team
"""
detect_phishing(test_email)
```

Hình 3.21. Kiểm thử với email ham nhưng mô hình nhận định là spam cho bộ dữ liệu SpamAssassin

Kết quả

🌐 Kết quả: SPAM (lừa đảo)
 📊 Độ tin cậy: 70.74%

🔴 Các đặc trưng quan trọng ảnh hưởng đến quyết định:
 - click (importance: 0.0135)

Hình 3.22. Kết quả với email ham nhưng mô hình nhận định là spam cho bộ dữ liệu SpamAssassin

3.3.2. Kiểm thử với bộ dữ liệu Kaggle

Trường hợp 1: Kiểm tra với email spam cho bộ dữ liệu Kaggle

```
# Test demo
test_email = """
From: account-security@microsoft-support.com
Subject: Action Required: Unusual Sign-in Attempt Detected
Dear User,
We noticed an unusual sign-in attempt to your Microsoft account from a new device.
If this was you, please verify your account immediately:
https://microsoft-verify-login.com
If you do not verify within 24 hours, your account will be locked for security reasons.
Thanks,
Microsoft Security Team
"""
detect_phishing(test_email)
```

Hình 3.23. Kiểm thử với email spam mẫu cho bộ dữ liệu Kaggle

Kết quả

📧 Kết quả: SPAM (lừa đảo)
📊 Độ tin cậy: 50.29%

🔴 Các đặc trưng quan trọng ảnh hưởng đến quyết định:

- thank (importance: 0.0188)
- subject (importance: 0.0177)

Hình 3.24. Kết quả email spam với bộ dữ liệu Kaggle

Trường hợp 2: Kiểm thử với email thường (ham) cho bộ dữ liệu Kaggle

```
# Test demo
test_email = """
From: linh.tran@company.com
Subject: Meeting Reminder Project Alpha
Hi team,
This is a reminder that we will have our next Project Alpha meeting tomorrow at 10:00 AM in Meeting Room B.
Agenda:
- Progress update
- Risk discussion
- Next steps
Please let me know if you have any additions.
Thanks,
Linh
"""
detect_phishing(test_email)
```

Hình 3.25. Kiểm thử với email ham mẫu cho bộ dữ liệu Kaggle

Kết quả

📧 Kết quả: HAM (hợp lệ)
📊 Độ tin cậy: 8.13%

🔴 Các đặc trưng quan trọng ảnh hưởng đến quyết định:

- thank (importance: 0.0188)
- subject (importance: 0.0177)



Hình 3.26. Kết quả email ham với bộ dữ liệu Kaggle


Trường hợp 3: Kiểm thử với email spam nhưng mô hình nhận định là email thường (ham) cho bộ dữ liệu Kaggle

```
# Test demo
test_email = """
Subject: Account Statement Available
From: notifications@yourbank.com
Dear Customer,
Your monthly account statement for April 2025 is now available.
Please log in to your account to view it: http://yourbank.com/login/statement.
If you have any questions, contact us at 1-800-555-1234.
Sincerely,
YourBank Team
"""
detect_phishing(test_email)
```

Hình 3.27. Kiểm thử với email spam nhưng mô hình nhận định là ham cho bộ dữ liệu Kaggle

Kết quả

 Kết quả: HAM (hợp lệ)
 Độ tin cậy: 37.19%

 Các đặc trưng quan trọng ảnh hưởng đến quyết định:
- subject (importance: 0.0177)



Hình 3.28. Kết quả với email spam nhưng mô hình nhận định là ham cho bộ dữ liệu Kaggle


Trường hợp 4: Kiểm thử với email thường (ham) nhưng mô hình nhận định là email spam cho bộ dữ liệu Kaggle

```
# Test demo
test_email = """
Subject: Please Update Your Employee Profile
From: hr@company.com
Hello [Employee],
We're updating our employee database. Please update your profile by clicking here: http://hr.company.com/update-profile.
This is required to ensure your information is current.
Best regards,
HR Team
"""
detect_phishing(test_email)
```

Hình 3.29. Kiểm thử với email ham nhưng mô hình nhận định là spam với bộ dữ liệu Kaggle

Kết quả

 Kết quả: SPAM (lừa đảo)
 Độ tin cậy: 56.00%

 Các đặc trưng quan trọng ảnh hưởng đến quyết định:
- subject (importance: 0.0177)

Hình 3.30. Kết quả với email ham nhưng mô hình nhận định là spam với bộ dữ liệu Kaggle

3.4. So sánh kết quả khi xử lý từ bộ dữ liệu SpamAssassin, Kaggle

Bảng 5: Bảng so sánh độ chính xác của 2 bộ dữ liệu SpamAssassin, Kaggle

	Độ chính xác	Độ nhạy (Ham)	Độ nhạy (Spam)	Độ đặc hiệu
SpamAssassin	95.58%	92.70%	98.75%	92.70%
Kaggle	95.53%	92.57%	98.84%	92.57%

Có thể thấy 2 bộ dữ liệu cho ra kết quả không có nhiều sự chênh lệch, kể cả việc đưa ra dự đoán khi kiểm tra với các email mới cũng có kết quả tương đương nhau.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Kết luận

Trong thời đại công nghệ số phát triển mạnh mẽ, các mối đe dọa về bảo mật thông tin, đặc biệt là các cuộc tấn công lừa đảo qua email (phishing), ngày càng gia tăng và tinh vi. Nhận thức được tầm quan trọng của việc bảo vệ người dùng trước những nguy cơ này, em đã thực hiện đồ án tốt nghiệp với đề tài “Xây dựng hệ thống phát hiện email lừa đảo bằng giải thuật Random Forest”.

Qua quá trình nghiên cứu và triển khai, em đã xây dựng thành công mô hình phát hiện email lừa đảo sử dụng thuật toán Random Forest. Mô hình này đã được huấn luyện và kiểm thử trên bộ dữ liệu đã được tiền xử lý, cho kết quả khả quan với độ chính xác cao. Việc sử dụng các chỉ số đánh giá như accuracy, precision, recall, F1-score và biểu đồ confusion matrix đã giúp em đánh giá hiệu quả của mô hình một cách toàn diện.

Tuy nhiên, do hạn chế về thời gian và kinh nghiệm, mô hình hiện tại vẫn còn một số điểm cần cải thiện, như khả năng xử lý các email lừa đảo mới với kỹ thuật tinh vi hơn. Trong tương lai, em dự định sẽ tiếp tục nghiên cứu và áp dụng các phương pháp học máy tiên tiến hơn, cũng như mở rộng bộ dữ liệu để nâng cao hiệu quả phát hiện.

Qua đồ án này, em đã tích lũy được nhiều kiến thức và kỹ năng quý báu trong lĩnh vực học máy và bảo mật thông tin. Em xin chân thành cảm ơn Quý Thầy Cô trong khoa Công nghệ Thông tin, đặc biệt là giảng viên hướng dẫn, đã tận tình hỗ trợ và góp ý để em hoàn thành tốt đồ án này.

Hướng phát triển

Trong tương lai, nhóm dự định tiếp tục mở rộng và nâng cao chất lượng mô hình bằng cách thu thập thêm dữ liệu email từ nhiều nguồn khác nhau. Việc tăng số lượng và sự đa dạng của dữ liệu huấn luyện sẽ giúp mô hình học được nhiều đặc điểm phức tạp hơn và nâng cao khả năng phát hiện email lừa đảo mới. Bên cạnh đó, nhóm cũng sẽ thử nghiệm với các thuật toán học máy tiên tiến hơn như XGBoost, LightGBM hoặc các mô hình học sâu (Deep Learning) như LSTM hoặc BERT – vốn rất mạnh trong phân tích văn bản và ngôn ngữ tự nhiên.

Để ứng dụng mô hình vào thực tế, nhóm có thể phát triển một phần mềm kiểm tra và cảnh báo email lừa đảo với giao diện người dùng thân thiện. Phần mềm này có thể được tích hợp vào hệ thống email của cá nhân hoặc doanh nghiệp để tự động quét và phát hiện các email nghi ngờ trước khi người dùng mở xem. Ngoài ra, mô hình cũng có thể được tích hợp vào tiện ích mở rộng trình duyệt (browser extension) nhằm cung cấp cảnh báo trực tiếp khi người dùng truy cập vào các liên kết đáng ngờ trong email.

Một hướng phát triển quan trọng khác là kết hợp mô hình phát hiện phishing với các công nghệ bảo mật khác như kiểm tra mã độc trong tệp đính kèm, phân tích URL độc hại hay xác minh chứng chỉ SSL của địa chỉ gửi. Việc tích hợp đa lớp bảo vệ như vậy sẽ tăng tính chính xác và toàn diện trong việc phát hiện các cuộc tấn công lừa đảo ngày càng tinh vi.

Cuối cùng, nhóm cũng định hướng triển khai mô hình lên các nền tảng điện toán đám mây như AWS, Google Cloud hoặc Azure để mở rộng quy mô xử lý và phục vụ nhiều người dùng hơn. Việc xây dựng một API dịch vụ phát hiện email phishing trên nền tảng web sẽ giúp các tổ chức, doanh nghiệp dễ dàng tích hợp vào hệ thống bảo mật của họ. Những bước phát triển này không chỉ nâng

cao hiệu quả mô hình mà còn đưa giải pháp đến gần hơn với ứng dụng thực tiễn trong lĩnh vực an toàn thông tin.

TÀI LIỆU THAM KHẢO

- [1]. 190K+ Spam | Ham Email Dataset for Classification - Meruvu Likith - 2024, from:<<https://www.kaggle.com/datasets/meruvulikith/190k-spam-ham-email-dataset-for-classification?resource=download>>
- [2]. SpamAssassin Public Corpus, from:<<https://spamassassin.apache.org/old/publiccorpus/>>
- [3]. Project Jupyter Documentation, from:<<https://docs.jupyter.org/en/latest/>>
- [4]. Random Forest là gì trong Machine Learning? Giải thích các thuật ngữ, cách hoạt động, và ví dụ thực tế - Công Duy - 29/11-2024, from:<<https://statio.vn/blog/random-forest-la-gi-trong-machine-learning-giai-thich-cac-thuat-ngu-cach-hoat-ong-va-vi-du-thuc-te>>
- [5]. Máy học là gì? Machine Learning là gì? Các dạng máy học và khái niệm liên quan đến Machine Learning - Digital FPT - 25/9/2024, from:<https://digital.fpt.com/chuyen-doi-so/tri-tue-nhan-cao-ai-vi-may-hoc-la-gi-machine-learning-la-gi-cac-dang-may-hoc-va-khai-niem-lien-quan-den-machine-learning.html?trk=article-ssr-frontend-pulse_little-text-block>