

BỘ THÔNG TIN VÀ TRUYỀN THÔNG
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
CƠ SỞ TẠI THÀNH PHỐ HỒ CHÍ MINH



BÁO CÁO TỔNG KẾT

**ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN
NĂM 2023**

**PHÁT HIỆN WEB PHISHING THÔNG QUA PHÂN
TÍCH BỐ CỤC TRANG WEB
26-SV-2023-TH2**

Thuộc nhóm ngành khoa học: An toàn thông tin

TPHCM – 10/2023

BỘ THÔNG TIN VÀ TRUYỀN THÔNG

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

CƠ SỞ TẠI THÀNH PHỐ HỒ CHÍ MINH

BÁO CÁO TỔNG KẾT

ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NĂM 2023

PHÁT HIỆN WEB PHISHING THÔNG QUA PHÂN TÍCH BỐ CỤC TRANG WEB

26-SV-2023-TH2

Thuộc nhóm ngành khoa học: An toàn thông tin

- Sinh viên thực hiện: Trần Thiện Nhân
Nam, Nữ: Nam
Dân tộc: Kinh

Lớp, khoa: Lớp D20CQAT01-N, Khoa Công nghệ thông tin 02
Năm thứ: 4 / Số năm đào tạo: 4.5

Ngành học: An toàn thông tin

- Sinh viên hỗ trợ: Phạm Hoàng Hiếu
Dân tộc: Kinh

Lớp, khoa: Lớp D20CQAT01-N, Khoa Công nghệ thông tin 02
Năm thứ: 4 / Số năm đào tạo: 4.5

Ngành học: An toàn thông tin

- Sinh viên hỗ trợ: Bùi Đức Phú
Dân tộc: Kinh

Lớp, khoa: Lớp D20CQAT01-N, Khoa Công nghệ thông tin 02
Năm thứ: 4 / Số năm đào tạo: 4.5

Ngành học: An toàn thông tin

Người hướng dẫn: Thạc sĩ Nguyễn Hoàng Thành

TP.HCM – 10/2023

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

LỜI CẢM ƠN

Để có thể hoàn thiện được đề tài này, trước tiên chúng em xin bày tỏ lòng biết ơn sâu sắc nhất tới Thầy Nguyễn Hoàng Thành (Ngành An toàn thông tin – Học Viện Công Nghệ Bưu Chính Viễn Thông Cơ Sở Thành Phố Hồ Chí Minh). Sự gần gũi và nhiệt tình hướng dẫn của Thầy là nguồn động lực rất lớn đối với chúng em trong suốt quá trình hoàn thiện đề tài này.

Chúng em cũng xin gửi lời cảm ơn chân thành nhất tới các Thầy, các Cô trong Khoa Công nghệ thông tin 02 – Học Viện Công Nghệ Bưu Chính Viễn Thông Cơ Sở Thành Phố Hồ Chí Minh đã nhiệt tình giảng dạy, cung cấp cho chúng em những kiến thức, những kinh nghiệm không chỉ trong học tập mà còn trong cuộc sống thường ngày.

Đồng thời chúng em cũng xin gửi lời cảm ơn đặc biệt nhất tới cha mẹ, anh chị em, người thân trong gia đình, bạn bè đã luôn động viên, giúp đỡ, tạo điều kiện tốt nhất cho chúng em trong suốt quá trình học tập tại Học Viện Công Nghệ Bưu Chính Viễn Thông Cơ Sở Thành Phố Hồ Chí Minh để chúng em có thể hoàn thiện tốt đề tài này.

TP. HCM, ngày 16 tháng 04 năm 2023

SINH VIÊN

TRẦN THIỆN NHÂN

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

MỤC LỤC

DANH MỤC NHỮNG HÌNH ẢNH	7
DANH MỤC NHỮNG TỪ VIẾT TẮT.....	8
LỜI MỞ ĐẦU	9
1. Tổng quan tình hình nghiên cứu.....	9
2. Lý do chọn đề tài	9
3. Mục tiêu đề tài	9
4. Phương pháp nghiên cứu	9
5. Đối tượng nghiên cứu.....	10
6. Phạm vi nghiên cứu	10
CHƯƠNG 1. NGHIÊN CỨU TỔNG QUAN WEB PHISHING.....	11
1.1 GIỚI THIỆU VỀ WEB PHISHING	11
1.1.1 Định nghĩa và mô tả Web Phishing	11
1.1.2 Các phương thức phổ biến của Web Phishing.....	11
1.1.3 Tác hại của Web Phishing đối với người dùng và tổ chức	12
1.2 CÁC PHƯƠNG PHÁP PHÁT HIỆN WEB PHISHING	13
1.2.1 Phương pháp dựa trên nội dung trang web.....	13
1.2.2 Phương pháp dựa trên hành vi người dùng	14
1.2.3 Phương pháp dựa trên bộ cục trang web	14
1.3 KỸ THUẬT MÁY HỌC TRONG PHÁT HIỆN WEB PHISHING	15
1.3.1 Học máy và trí tuệ nhân tạo.....	15
1.3.2 Ứng dụng kỹ thuật máy học trong phát hiện Web Phishing thông qua phân tích bộ cục trang web.....	16
1.4 CÁC CÔNG TRÌNH PHÁT HIỆN WEB PHISHING TRONG NUỐC VÀ TRÊN THẾ GIỚI	16
1.4.1 Các công trình trong nước	16
1.4.2 Các công trình trên thế giới	17

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

CHƯƠNG 2. NGUYÊN CỨU THUẬT TOÁN HỌC MÁY SUPPORT VECTOR MACHINES VÀ DECISION TREES.....	20
2.1 HỌC MÁY SUPPORT VECTOR MACHINES	20
2.1.1 Giới thiệu	20
2.1.2 Lịch sử phát triển	20
2.1.3 Tổng quan	20
2.1.4 Cách Support Vector Machines hoạt động	21
2.1.5 Support Vector Machines Kernels	23
2.1.5.1 Linear Kernel.....	23
2.1.5.2 Polynomial Kernel.....	24
2.1.5.3 Radial Basis Function Kernel.....	25
2.1.5.4 Sigmoid Kernel.....	26
2.1.6 Xây dựng bộ phân loại Support Vector Machines bằng Scikit-learn	27
2.1.6.1 Nạp dữ liệu	27
2.1.6.2 Khám phá dữ liệu	27
2.1.6.3 Chia dữ liệu	27
2.1.6.4 Tạo mô hình.....	27
2.1.6.5 Đánh giá mô hình	28
2.1.6.6 Tinh chỉnh mô hình.....	28
2.1.6.7 Hiển thị kết quả	28
2.1.6.8 Lưu và tải mô hình	29
2.1.6.9 Sử dụng mô hình	29
2.1.7 Điều chỉnh tham số Hyperparameter trong Support Vector Machines.....	29
2.1.7.1 Kernel	29
2.1.7.2 Chính Quy Hóa (Regularization)	30
2.1.7.3 Gamma	30
2.1.8 Ưu Điểm của Support Vector Machines	30

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

2.1.9 Nhược Điểm của Support Vector Machines	31
2.2 HỌC MÁY DECISION TREES.....	31
2.2.1 Giới thiệu	31
2.2.2 Lịch sử phát triển.....	32
2.2.3 Tổng quan	32
2.2.4 Cách Decision Trees hoạt động	33
2.2.5 Các biện pháp chọn lựa thuộc tính	35
2.2.5.1 Information Gain	35
2.2.5.2 Gain Ratio.....	36
2.2.5.3 Gini Index	37
2.2.6 Xây dựng bộ phân loại Decision Trees bằng Scikit-learn	37
2.2.6.1 Nhập thư viện cần thiết.....	37
2.2.6.2 Nạp dữ liệu	38
2.2.6.3 Lựa chọn đặc trưng.....	38
2.2.6.4 Chia dữ liệu	38
2.2.6.5 Xây dựng mô hình	38
2.2.6.6 Đánh giá mô hình	38
2.2.6.7 Hiển thị cây quyết định.....	39
2.2.6.8 Điều chỉnh siêu tham số	39
2.2.6.9 Kiểm định chéo.....	39
2.2.7 Tối ưu hóa hiệu suất Decision Trees	40
2.2.8 Ưu Điểm của Decision Trees.....	41
2.2.9 Nhược Điểm của Decision Trees.....	42
CHƯƠNG 3. XÂY DỰNG CHƯƠNG TRÌNH PHÁT HIỆN WEB PHISHING THÔNG QUA PHÂN TÍCH BỐ CỤC TRANG WEB	43
3.1 Giới thiệu	43
3.2 Thiết lập môi trường.....	44

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

3.2.1 Môi trường phát triển	44
3.2.2 Phiên bản Python.....	44
3.2.3 Bộ dữ liệu	44
3.2.4 Thư viện và công cụ	45
3.3 Cấu trúc chương trình.....	47
3.3.1 Hằng số và cấu hình	47
3.3.2 Hàm chính	47
3.4 Phân tích dữ liệu HTML.....	48
3.4.1 Đọc nội dung HTML	48
3.4.2 Rút trích đặc trưng HTML	48
3.5 Thuật toán học máy	50
3.5.1 Huấn luyện và lưu mô hình học máy.....	50
3.5.2 Đánh giá hiệu suất của mô hình học máy.....	51
3.6 Triển khai và ứng dụng.....	53
KẾT LUẬN	71
1. Hiệu quả.....	71
2. Hạn chế	71
3. Hướng phát triển trong tương lai.....	71
TÀI LIỆU THAM KHẢO	72

DANH MỤC NHỮNG HÌNH ẢNH

Hình 1 Quy trình phát hiện Web Phishing [1]	16
Hình 2 Sơ đồ khái [3]	18
Hình 3 Bảng so sánh độ chính xác của các thuật toán [3]	18
Hình 4 Mô tả thuật toán Support Vector Machines [7]	21
Hình 5 Bộ dữ liệu gốc [8]	22
Hình 6 Dữ liệu với bộ tách là đường cong đã được thêm vào [8]	23
Hình 7 Dữ liệu đã được phân chia [8]	23
Hình 8 Cây quyết định trong việc nhận thức phòng chống bệnh đau tim [10]	33
Hình 9 Các dataset huấn luyện được gắn nhãn từ cơ sở dữ liệu khách hàng [11]	34
Hình 10 Cây quyết định từ các dataset [11]	35
Hình 11 Bộ dữ liệu "Phishing Website HTML Classification" [13]	45
Hình 12 File logging wp.log ghi lại quá trình hoạt động của chương trình	65
Hình 13 Thông tin về độ chính xác và ma trận nhầm lẫn	65
Hình 14 Trang Báo Điện tử Chính phủ	66
Hình 15 Chương trình hoạt động với trang web không phishing	67
Hình 16 Trang OpenPhish	68
Hình 17 Chương trình hoạt động với trang web phishing	69
Hình 18 Lịch sử kiểm tra từ file results.csv	70

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

DANH MỤC NHỮNG TỪ VIẾT TẮT

AI	Artificial Intelligence
CART	Classification And Regression Tree
CPU	Central Processing Unit
CSS	Cascading Style Sheet
CSV	Comma-Separated Values
DOM	Document Object Model
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HTML	Hyper Text Markup Language
ID3	Iterative Dichotomiser 3
IDE	Integrated Development Environment
IP	Internet Protocol
ML	Machine Learning
RBF	Radial Basis Function
SEO	Search Engine Optimization
SMS	Short Message Service
SSL	Secure Socket Layer
SVM	Support Vector Machines
URL	Uniform Resource Locator

LỜI MỞ ĐẦU

1. Tổng quan tình hình nghiên cứu

Trong những năm gần đây, tội phạm mạng đang ngày càng gia tăng, trong đó tấn công Web Phishing là một hình thức phổ biến và nguy hiểm. Tội phạm mạng thường giả mạo các trang web chính thức để lừa đảo người dùng và đánh cắp thông tin cá nhân, tài khoản và các dữ liệu quan trọng khác.

Việc phát hiện các tấn công Web Phishing là một vấn đề quan trọng trong lĩnh vực an toàn thông tin. Tuy nhiên, phát hiện tấn công Web Phishing là công việc khó khăn, và dù đã có nhiều giải pháp được công bố, nhưng vẫn cần cải thiện độ chính xác. Phân tích bối cảnh trang web có thể được xem là một phương pháp hiệu quả trong việc phát hiện Web Phishing.

2. Lý do chọn đề tài

Từ thực tế trên, chúng em chọn đề tài "Phát hiện Web Phishing thông qua phân tích Bối cảnh trang web" để nghiên cứu.

3. Mục tiêu đề tài

Mục tiêu của đề tài là xây dựng chương trình dùng học máy để phân tích bối cảnh trang web nhằm phát hiện các trang Web Phishing.

4. Phương pháp nghiên cứu

Phương pháp nghiên cứu định tính được sử dụng để nghiên cứu các đặc điểm của Web Phishing, bao gồm:

- Nghiên cứu tổng quan về Web Phishing: Sử dụng phương pháp tổng quan tài liệu để tìm hiểu các khái niệm, đặc điểm, nguyên nhân và tác hại của Web Phishing.
- Nghiên cứu hai thuật toán máy học: Sử dụng phương pháp phân tích tài liệu để tìm hiểu hai thuật toán máy học là Support Vector Machines và Decision Trees.

Phương pháp nghiên cứu định lượng được sử dụng để xây dựng bộ dữ liệu và đánh giá hiệu quả của chương trình phát hiện tấn công Web Phishing.

- Xây dựng bộ dữ liệu: Sử dụng phương pháp thu thập dữ liệu và phân loại dữ liệu để xây dựng bộ dữ liệu gồm hai tập dữ liệu Phishing và Not Phishing.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

- Xây dựng chương trình phát hiện tấn công Web Phishing: Sử dụng phương pháp huấn luyện mô hình máy học để xây dựng chương trình phát hiện tấn công Web Phishing.

5. Đối tượng nghiên cứu

Đối tượng nghiên cứu chính của đề tài là Web Phishing và phương pháp phân tích bộ cục trang web gồm hai thuật toán học máy.

6. Phạm vi nghiên cứu

Phạm vi nghiên cứu thời gian: Nghiên cứu sẽ được thực hiện trong 10 tháng kể từ tháng 12 năm 2022 đến tháng 09 năm 2023.

Phạm vi nghiên cứu không gian: Nghiên cứu sẽ được thực hiện trên các trang web Phishing và không Phishing.

Phạm vi nghiên cứu lĩnh vực: Nghiên cứu sẽ tập trung vào việc phát triển chương trình phát hiện tấn công Web Phishing thông qua phân tích bộ cục trang web sử dụng hai thuật toán học máy Support Vector Machines và Decision Trees.

Với điều kiện thời gian cũng như kinh nghiệm còn hạn chế của sinh viên, nghiên cứu này không thể tránh được những thiếu sót. Chúng em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của các thầy, các cô để chúng em có điều kiện bổ sung, nâng cao và phục vụ tốt hơn trong công việc sau này.

Chúng em xin chân thành cảm ơn.

CHƯƠNG 1. NGHIÊN CỨU TỔNG QUAN WEB PHISHING

1.1 GIỚI THIỆU VỀ WEB PHISHING

1.1.1 Định nghĩa và mô tả Web Phishing

Web Phishing là một kỹ thuật lừa đảo trực tuyến, trong đó tin tặc tạo ra một trang web giả mạo giống như trang web của một tổ chức nào đó, ví dụ như ngân hàng, dịch vụ email hoặc mạng xã hội, với mục đích lừa đảo người dùng cung cấp thông tin cá nhân nhạy cảm như tên đăng nhập, mật khẩu, thông tin thẻ tín dụng, v.v. Thông thường, tin tặc sẽ gửi cho người dùng một email giả mạo, yêu cầu họ truy cập vào trang web giả mạo và cập nhật lại thông tin cá nhân của mình. Trang web giả mạo này sẽ được thiết kế giống hệt trang web chính thức của tổ chức đó, bao gồm các hình ảnh, biểu tượng và giao diện người dùng tương tự. Khi người dùng truy cập vào trang web giả mạo và cung cấp thông tin cá nhân, tin tặc sẽ sử dụng thông tin này để thực hiện các hoạt động lừa đảo như trộm tiền, đánh cắp danh tính hoặc tấn công mạng. Để tránh bị mắc kẹt trong kỹ thuật lừa đảo này, người dùng cần phải cẩn trọng khi truy cập vào các liên kết từ email hoặc trang web không rõ nguồn gốc. Đồng thời, họ nên kiểm tra xem địa chỉ URL trên trình duyệt có phù hợp với trang web chính thức của tổ chức hay không và cung cấp thông tin cá nhân chỉ trên các trang web đáng tin cậy. Ngoài ra, các công cụ bảo mật như phần mềm chống virus, phần mềm chặn spam cũng là cách giúp người dùng giảm thiểu rủi ro bị lừa đảo trực tuyến.

1.1.2 Các phương thức phổ biến của Web Phishing

- Phishing qua email: Phương thức này bao gồm việc gửi email giả mạo từ một tổ chức hoặc công ty uy tín. Email này sẽ yêu cầu người dùng cung cấp thông tin cá nhân hoặc đăng nhập vào tài khoản của họ bằng cách nhấp vào một liên kết được cung cấp trong email. Liên kết sẽ dẫn đến một trang web giả mạo, nơi kẻ tấn công có thể lấy thông tin người dùng.
- Phishing qua trang web giả mạo: Kẻ tấn công có thể tạo ra một trang web giả mạo giống hệt trang web của một tổ chức hoặc công ty nào đó. Khi người dùng truy cập vào trang web giả mạo này, họ sẽ được yêu cầu cung cấp thông tin cá nhân hoặc đăng nhập vào tài khoản của họ. Khi thông tin này được cung cấp, nó sẽ được gửi đến kẻ tấn công.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

- Smishing (SMS Phishing): Kẻ tấn công sử dụng tin nhắn văn bản giả mạo để lừa người dùng cung cấp thông tin cá nhân hoặc đăng nhập vào tài khoản của họ. Tin nhắn này có thể yêu cầu người dùng nhấp vào một liên kết hoặc gọi số điện thoại cung cấp để tiếp tục thực hiện quá trình xác thực.
- Vishing (Voice Phishing): Phương thức này liên quan đến việc sử dụng điện thoại để lừa đảo người dùng. Kẻ tấn công sẽ gọi điện cho người dùng và giả vờ là một tổ chức hoặc công ty nào đó. Họ sẽ yêu cầu người dùng cung cấp thông tin cá nhân hoặc đăng nhập vào tài khoản của họ. Khi thông tin này được cung cấp, nó sẽ được gửi đến kẻ tấn công.
- Tabnabbing: Phương thức này liên quan đến việc thay đổi nội dung trang web mà người dùng đang truy cập để lừa đảo người dùng. Khi người dùng đang truy cập một trang web, kẻ tấn công sẽ sử dụng mã độc để thay đổi nội dung trang web đó thành một trang web giả mạo. Khi người dùng chuyển đến trang web giả mạo này, họ sẽ được yêu cầu cung cấp thông tin cá nhân hoặc đăng nhập vào tài khoản của mình. Khi thông tin này được cung cấp, nó sẽ được gửi đến kẻ tấn công.
- Search Engine Phishing: Phương thức này liên quan đến việc lừa đảo người dùng bằng cách sử dụng các kết quả tìm kiếm giả mạo trên các công cụ tìm kiếm. Khi người dùng tìm kiếm thông tin trên internet, kẻ tấn công sẽ sử dụng các kỹ thuật SEO để đẩy các trang web giả mạo của họ lên đầu danh sách kết quả tìm kiếm. Khi người dùng nhấp vào một liên kết từ các kết quả tìm kiếm này, họ sẽ được chuyển hướng đến một trang web giả mạo và bị yêu cầu cung cấp thông tin cá nhân hoặc đăng nhập vào tài khoản của họ.
- Malware-Based Phishing: Kẻ tấn công có thể sử dụng phần mềm độc hại để lừa đảo người dùng. Phần mềm độc hại này có thể được phát tán qua email, trang web giả mạo, tin nhắn văn bản hoặc các tài liệu tải xuống. Khi người dùng bị nhiễm phần mềm độc hại này, kẻ tấn công có thể thu thập thông tin cá nhân hoặc đăng nhập vào tài khoản của họ.

1.1.3 Tác hại của Web Phishing đối với người dùng và tổ chức

Trong khi các website giả mạo được tạo ra để lừa đảo người dùng, tấn công phishing thường được tiến hành bằng cách gửi email hoặc tin nhắn SMS với nội dung giả mạo để lừa người dùng cung cấp thông tin cá nhân hay đăng nhập vào các trang web giả mạo.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

Những thông tin này thường bao gồm tài khoản và mật khẩu, số thẻ tín dụng hoặc số chứng minh nhân dân, dẫn đến các hậu quả nghiêm trọng cho cá nhân và tổ chức.

Đối với người dùng, tác hại của Web Phishing có thể bao gồm việc bị mất tiền, bị truy cập trái phép vào tài khoản ngân hàng và thông tin cá nhân, bị lừa tiền hoặc tài sản, bị đánh cắp thông tin đăng nhập của các tài khoản cá nhân và kinh doanh quan trọng, và thậm chí bị sử dụng cho các hoạt động trái pháp luật.

Đối với tổ chức, tác hại của Web Phishing cũng rất đáng lo ngại. Nếu các kẻ tấn công có thể lấy được thông tin đăng nhập của các nhân viên, họ có thể truy cập và đánh cắp dữ liệu nhạy cảm, thậm chí kiểm soát các tài khoản quản trị. Ngoài ra, các tổ chức cũng có thể bị đánh cắp thông tin cá nhân của khách hàng, dẫn đến mất mát uy tín và đòi hỏi phải chi tiêu để bù đắp thiệt hại.

Vì vậy, để bảo vệ bản thân và tổ chức khỏi các tác hại của Web Phishing, người dùng và tổ chức cần phải học cách nhận biết các website giả mạo và tin tặc qua các cách thức giả mạo email hoặc tin nhắn SMS, đồng thời sử dụng các công cụ bảo mật và cập nhật phần mềm thường xuyên để giảm thiểu rủi ro. Ngoài ra, các tổ chức cần đào tạo nhân viên của mình về các phương pháp tấn công mạng để có thể phát hiện sớm những tấn công độc hại và thực hiện các biện pháp bảo mật hiệu quả hơn. Các tổ chức cũng có thể sử dụng các giải pháp bảo vệ dữ liệu như mã hóa và giám sát các hoạt động trên mạng để phát hiện các hoạt động bất thường và ngăn chặn các tấn công từ các hacker.

Ngoài ra, các nhà cung cấp dịch vụ internet cũng có trách nhiệm giám sát và bảo vệ khách hàng của mình khỏi các tấn công mạng. Các nhà cung cấp có thể sử dụng các giải pháp chặn các website giả mạo và đưa ra cảnh báo đối với các người dùng khi họ truy cập vào các website có nguy cơ.

1.2 CÁC PHƯƠNG PHÁP PHÁT HIỆN WEB PHISHING

1.2.1 Phương pháp dựa trên nội dung trang web

Để thực hiện phương pháp này, các nhà nghiên cứu đã phát triển các thuật toán máy học và mô hình xử lý ngôn ngữ tự nhiên để phân tích nội dung của trang web. Các thuật toán này sử dụng các kỹ thuật phân tích cú pháp và từ vựng để xác định các phần tử trang web độc hại, chẳng hạn như các liên kết giả mạo và các trường nhập liệu giả mạo.

Một trong những kỹ thuật được sử dụng trong phương pháp này là kỹ thuật tìm kiếm chuỗi đặc biệt. Kỹ thuật này cho phép các nhà nghiên cứu xác định các chuỗi ký tự đặc

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

bíệt trong nội dung của trang web, chẳng hạn như các địa chỉ URL giả mạo và các từ khóa độc hại. Các chuỗi này có thể được sử dụng để xác định các trang web lừa đảo.

Ngoài ra, phương pháp này cũng sử dụng kỹ thuật phân tích nội dung trang web để xác định các phần tử độc hại trong trang web. Các phần tử này có thể bao gồm các tài liệu giả mạo, các tài khoản giả mạo và các liên kết đến các trang web lừa đảo khác.

1.2.2 Phương pháp dựa trên hành vi người dùng

Phương pháp này sử dụng các kỹ thuật máy học và học sâu để phân tích hành vi của người dùng và xác định các mẫu và đặc điểm của các trang web bị tấn công bởi Web Phishing. Nó đánh giá các thông tin liên quan đến hành vi của người dùng, bao gồm lịch sử truy cập web, các hoạt động trên trang web và các hành động khác liên quan đến trang web.

Phương pháp này cũng đánh giá các yếu tố khác nhau như tính chân thực của trang web, quy trình xác thực người dùng và cơ chế bảo vệ thông tin. Điều này giúp người dùng đánh giá và phát hiện các trang web bị tấn công bởi Web Phishing một cách chính xác và hiệu quả.

1.2.3 Phương pháp dựa trên bộ cục trang web

Phương pháp này bao gồm hai bước chính. Đầu tiên, bộ cục của trang web được phân tích và các đặc trưng của trang web được trích xuất. Các đặc trưng này có thể bao gồm các thông tin về cấu trúc của trang web, vị trí của các phần tử trong trang web, thông tin về kích thước và màu sắc của các phần tử, và các đặc trưng khác.

Sau đó, các mô hình máy học được sử dụng để phân loại trang web là phishing hay không. Các mô hình này được đào tạo trên các tập dữ liệu được gắn nhãn và bao gồm các kỹ thuật như học máy, mạng nơ-ron và các phương pháp khác.

Phương pháp phát hiện Web Phishing dựa trên bộ cục trang web có nhiều ưu điểm. Đầu tiên, phương pháp này có thể phát hiện các trang web phishing mới và chưa được biết đến trước đó. Thứ hai, phương pháp này có thể hoạt động hiệu quả trên các trang web được tạo ra bằng các công cụ tự động và được xây dựng theo cách không đồng nhất. Cuối cùng, phương pháp này có thể đưa ra kết quả phân loại chính xác và nhanh chóng. Tuy nhiên, cũng có những hạn chế của phương pháp này. Đối với các trang web phishing được thiết kế đặc biệt để tránh phát hiện, phương pháp này có thể không hiệu quả. Ngoài

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

ra, phương pháp này cần một lượng lớn dữ liệu huấn luyện và các kỹ thuật xử lý dữ liệu để đạt được hiệu quả tối đa.

Ngoài ra, để nâng cao hiệu quả của phương pháp này, các nghiên cứu đang tiếp tục phát triển và cải tiến các mô hình máy học và kỹ thuật phân tích dữ liệu để có thể phát hiện các trang web phishing với độ chính xác và tốc độ cao hơn.

Các nhà nghiên cứu cũng đang tập trung vào việc kết hợp các phương pháp phát hiện phishing khác nhau để tạo ra các hệ thống phát hiện phishing toàn diện hơn. Ví dụ như kết hợp phương pháp phát hiện dựa trên bộ cục trang web với phương pháp phát hiện dựa trên nội dung trang web, hoặc sử dụng phương pháp phát hiện phishing dựa trên các thông tin về địa chỉ IP và tên miền để đánh giá mức độ rủi ro của trang web.

1.3 KỸ THUẬT MÁY HỌC TRONG PHÁT HIỆN WEB PHISHING

1.3.1 Học máy và trí tuệ nhân tạo

Trí tuệ nhân tạo (AI) là lĩnh vực của khoa học máy tính liên quan đến việc phát triển các hệ thống có khả năng tự học và tự cải tiến dựa trên dữ liệu. Một trong những ứng dụng tiêu biểu của AI là học máy (Machine Learning), một kỹ thuật cho phép máy tính học hỏi từ dữ liệu một cách tự động mà không cần phải được lập trình rõ ràng.

Trong kỹ thuật phát hiện web phishing, học máy và AI đóng vai trò quan trọng trong việc xây dựng các mô hình phân loại để nhận diện các trang web phishing. Các mô hình này được huấn luyện trên dữ liệu có sẵn về các trang web đáng tin cậy và các trang web phishing, và sau đó được sử dụng để phân loại các trang web mới.

Các thuật toán học máy thường được sử dụng trong kỹ thuật phát hiện web phishing bao gồm cây quyết định (Decision Trees) và hỗ trợ vectơ (Support Vector Machines). Mỗi thuật toán có các ưu điểm và hạn chế riêng, và việc lựa chọn thuật toán phù hợp là một bước quan trọng trong quá trình xây dựng các mô hình phân loại.

Các kỹ thuật học máy cũng được sử dụng trong việc phát hiện các kỹ thuật tấn công phishing mới, bằng cách phân tích các đặc trưng của các trang web độc hại và tự động tìm kiếm các mẫu mới.

1.3.2 *Ứng dụng kỹ thuật máy học trong phát hiện Web Phishing thông qua phân tích bối cảnh trang web*

Máy học có thể được sử dụng để phát hiện các trang web phishing dựa trên phân tích bối cảnh. Các thuật toán máy học có thể được huấn luyện để phân tích các bối cảnh trang web và xác định xem một trang web có phải là trang web phishing hay không. Các kỹ thuật máy học có thể được sử dụng để tìm ra các đặc trưng quan trọng của bối cảnh, như đường viền hay vị trí của các nút nhấn, và sử dụng chúng để phân loại các trang web.

1.4 CÁC CÔNG TRÌNH PHÁT HIỆN WEB PHISHING TRONG NƯỚC VÀ TRÊN THẾ GIỚI

1.4.1 *Các công trình trong nước*

“**Detecting Phishing Web Pages based on DOM-Tree Structure and Graph Matching Algorithm.**”

Năm 2014, tác giả Le Dang Nguyen, Đại học Hải Phòng, cùng các thành viên trong nhóm của mình nghiên cứu và đề xuất các giải pháp để phát hiện các trang web lừa đảo, giả mạo dựa trên cấu trúc của cây DOM và thuật toán Graph Matching. Thuật toán đề xuất gồm 4 bước để phát hiện các trang web lừa đảo lần lượt là trích xuất thông tin từ cây DOM, xác định độ tương đồng giữa web thật và web lừa đảo, phát hiện và báo cáo Phishing. [1]



Hình 1 Quy trình phát hiện Web Phishing [1]

“**Phishing Attacks Detection Using Genetic Programming**”

Năm 2014, tác giả Tuan Anh Pham cùng các thành viên trong nhóm đã đề xuất phương pháp phòng chống Phishing bằng Genetic Programming. Sau các quá trình thử nghiệm trên các trang web giả mạo và bắt hợp pháp thu thập được từ trên internet. Kết quả cho thấy Genetic Programming có khả năng tạo ra các mô hình dự đoán phân loại chính xác hơn các kỹ thuật học máy khác. Các giai đoạn để triển khai phòng chống tấn công Phishing bằng Genetic Programming gồm hai phần là khai thác tính năng và mô tả hệ thống. Giai đoạn đầu tiên của việc sử dụng Genetic Programming để giải quyết vấn đề

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

phát hiện lừa đảo là trích xuất tính năng. Đây là một bước rất quan trọng vì nó có thể ảnh hưởng mạnh mẽ đến hiệu quả của thuật toán học tập Genetic Programming. Các tính năng được trích xuất phải chứa thông tin giúp phân biệt giữa các trang web lừa đảo và hợp pháp. Giai đoạn tiếp theo là mô tả hệ thống được chia làm hai phần là đào tạo và thử nghiệm. Mục tiêu của phần đào tạo là phát triển mô hình bộ phân loại có thể xác định một trang web là lừa đảo hay hợp pháp dựa trên các giá trị tính năng của nó. Còn trong phần thử nghiệm, mô hình đã học được sử dụng để đưa ra dự đoán về dữ liệu chưa xem. Độ chính xác của những dự đoán này được sử dụng như một chỉ báo về chất lượng hiệu quả của mô hình. [2]

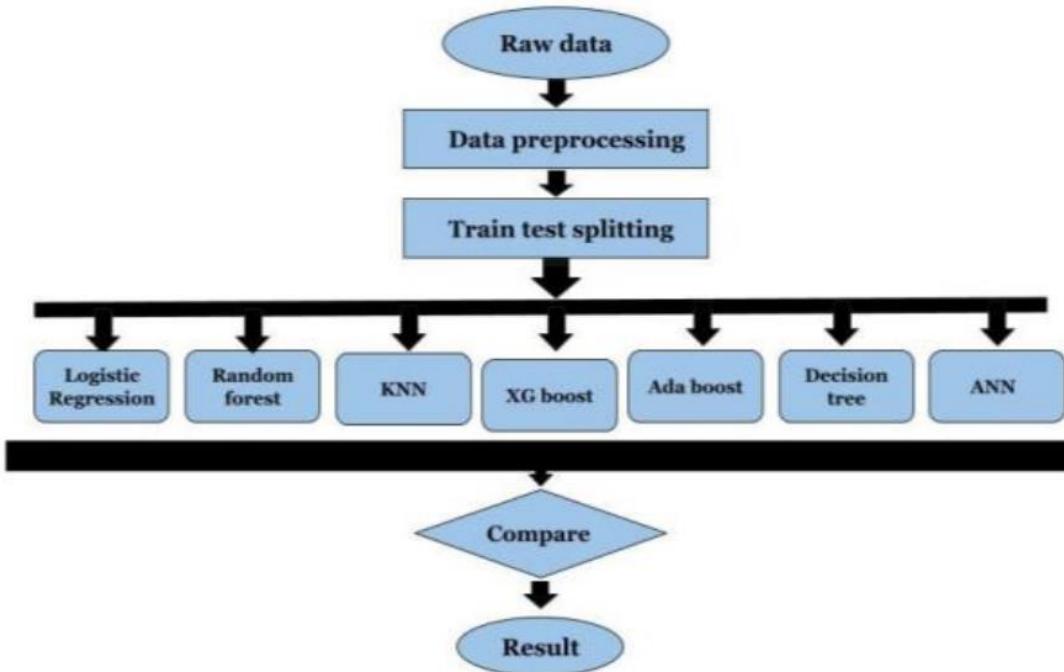
1.4.2 Các công trình trên thế giới

“Phishing website detection using machine learning and deep learning techniques.”

Đi cùng với sự phát triển nhanh chóng của internet là số lượng website ngày càng tăng. Do đó phát sinh nên các cuộc tấn công lừa đảo trên mạng. Trong các cuộc tấn công này, kẻ tấn công mạo danh một thực thể đáng tin cậy có ý định đánh cắp thông tin nhạy cảm hoặc danh tính kỹ thuật số của người dùng, ví dụ: thông tin đăng nhập tài khoản, số thẻ tín dụng và các chi tiết khác của người dùng. Nhận thấy được sự nguy hiểm của Phishing nhóm các nhà khoa học gồm Selvakumari M, Sowjanya M, Sneha Das và Padmavathi S đã thực hiện nghiên cứu nhằm so sánh các thuật toán Machine Learning, kỹ thuật Deep Learning và thuật toán tốt nhất có độ chính xác và độ chính xác cao nhất được chọn để phát hiện trang web lừa đảo.

Nhóm nghiên cứu đã tiến hành lấy bộ dữ liệu được sử dụng rộng rãi trên mạng như Kaggle và kết hợp với bộ dữ liệu mà nhóm tự xây dựng. 20% bộ dữ liệu lừa đảo từ Kaggle được sử dụng để kiểm tra mô hình và sau đó 80% còn lại của tập dữ liệu được sử dụng để đào tạo mô hình. Sau đó, bộ dữ liệu sẽ trải qua quá trình tiền xử lý bao gồm nhiều kỹ thuật trích xuất đặc trưng như Feature Extraction, Instance Selection, Normalization... Đây là quá trình có thể ảnh hưởng đến độ chính xác và kết quả cuối cùng, vì thế quá trình này cần phải xử lý và loại bỏ các đặc trưng cũng như dữ liệu không cần thiết cho quá trình thử nghiệm. [3]

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26



Hình 2 Sơ đồ khái [3]

Quá trình thử nghiệm sẽ áp dụng các thuật toán Machine Learning như Logistic Regression, K Nearest Neighbor, Decision Tree, Random Forest, XG Boost, AdaBoost. Để so sánh và chọn ra được thuật toán tốt nhất cho mô hình.

S.No	Algorithm	Training set accuracy	Testing set accuracy	Precision Score
1	Logistic Regression	79.00	79.00	82.30
2	K Nearest Neighbor	96.70	93.10	93.84
3	Decision Tree	100	95.50	96.13
4	Random Forest	95.00	94.40	94.81
5	XG Boost	93.80	93.40	93.92
6	Ada Boost	87.00	86.90	84.71

Hình 3 Bảng so sánh độ chính xác của các thuật toán [3]

“Phishing Website Detection Based on Deep Convolutional Neural Network and Random Forest Ensemble Learning”

Các cuộc tấn công Phishing đã trở thành một mối quan tâm đáng kể do sự gia tăng về số lượng của chúng. Gây ra thiệt hại hàng trăm triệu đô la và hàng triệu vụ vi phạm dữ liệu mỗi năm. Hiện tại, các kỹ thuật chống lừa đảo yêu cầu các chuyên gia trích xuất các

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

tính năng của các trang lừa đảo và sử dụng các dịch vụ của bên thứ ba để phát hiện các trang lừa đảo. Nhưng kỹ thuật này vẫn còn nhiều hạn chế như việc phát hiện tồn tại nhiều thời gian và độ chính xác không phải là ngoại lệ do những khó khăn liên quan đến trích xuất tính năng dịch vụ của bên thứ ba. Phát hiện trang web lừa đảo dựa trên máy học sử dụng các phương pháp máy học để phát hiện các tính năng URL của trang web lừa đảo được trích xuất thủ công. Hiệu quả của việc phát hiện có thể được cải thiện bằng cách sử dụng phương pháp này. Do đó, bài báo này đề xuất một phương pháp phát hiện trang web lừa đảo tích hợp dựa trên Convolutional Neural Networks và Random Forest. Phương pháp này có thể dự đoán tính hợp pháp của các URL mà không cần truy cập nội dung web hoặc sử dụng dịch vụ của bên thứ ba. Kỹ thuật được đề xuất sử dụng kỹ thuật nhúng ký tự để chuyển đổi URL thành ma trận kích thước cố định, trích xuất các tính năng ở các cấp độ khác nhau bằng mô hình Convolutional Neural Networks, phân loại các đối tượng địa lý nhiều cấp độ bằng cách sử dụng nhiều bộ phân loại Random Forest và cuối cùng, xuất ra kết quả dự đoán bằng cách sử dụng phương pháp lấy tất cả. Trên tập dữ liệu, tỷ lệ chính xác 99,35% đã đạt được bằng cách sử dụng mô hình được đề xuất. Tỷ lệ chính xác đạt được là 99,26% trên dữ liệu điểm chuẩn, cao hơn nhiều so với tỷ lệ của mô hình cực đoan hiện có. [4]

“PhishTrim: Fast and adaptive phishing detection based on deep representation learning.”

Năm 2020, nhóm nghiên cứu phòng chống tấn công Phishing đến từ Trung Quốc gồm Lei Zhang và Peng Zhang đã đề xuất phương pháp PhishTrim - một phương pháp phát hiện URL lừa đảo có mức độ nguy hiểm nhẹ dựa trên học sâu các đặc trưng, phương pháp được nhóm tác giả nhận định là nhanh và thích ứng tốt. Nhóm tác giả lấy các đặc trưng nhúng ban đầu từ các URL thông qua mô hình được huấn luyện trước đó là Skip-gram. Sau đó, bộ nhớ ngắn hạn hai chiều được sử dụng để trích xuất sự phụ thuộc vào ngữ cảnh để tìm hiểu thêm về đặc trưng sâu của URL. Các đặc tính n-gram cục bộ được trích xuất bằng cách sử dụng Convolutional Neural Networks. Các thử nghiệm cho thấy PhishTrim hoạt động tốt hơn trên các bộ dữ liệu quy mô lớn với độ chính xác 99,797% và chỉ ra rằng phương pháp của nhóm tác giả có khả năng nhất định để phát hiện các cuộc tấn công lừa đảo zero-day. [5]

CHƯƠNG 2. NGUYÊN CỨU THUẬT TOÁN HỌC MÁY

SUPPORT VECTOR MACHINES VÀ DECISION TREES

2.1 HỌC MÁY SUPPORT VECTOR MACHINES

2.1.1 Giới thiệu

Support Vector Machines là một trong những thuật toán học máy phổ biến và hiệu quả, được sử dụng trong nhiều lĩnh vực như phân loại dữ liệu, nhận dạng khuôn mẫu, hồi quy, và phân tích đa biến...

2.1.2 Lịch sử phát triển

Support Vector Machines được giới thiệu lần đầu tiên vào năm 1963 bởi hai nhà khoa học máy tính Vladimir Vapnik và Alexey Chervonenkis. Tuy nhiên, nó không được chú ý nhiều cho đến đầu những năm 1990, khi Vapnik và các đồng nghiệp của ông tại AT&T Bell Labs tìm ra cách áp dụng Kernel Trick cho Support Vector Machines, giúp thuật toán này có thể giải quyết các bài toán phức tạp và không tuyến tính. Kể từ đó, Support Vector Machines đã trở thành một trong những thuật toán học máy phổ biến nhất và được nghiên cứu nhiều nhất. [6]

2.1.3 Tổng quan

Support Vector Machines không chỉ được biết đến như một kỹ thuật phân loại mạnh mẽ, mà còn có khả năng thực hiện cả bài toán hồi quy một cách hiệu quả. Được trang bị khả năng linh động trong việc xử lý các biến liên tục và phân loại, SVM tạo ra một mặt phẳng siêu không gian ở không gian đa chiều để phân biệt rõ ràng các lớp dữ liệu.

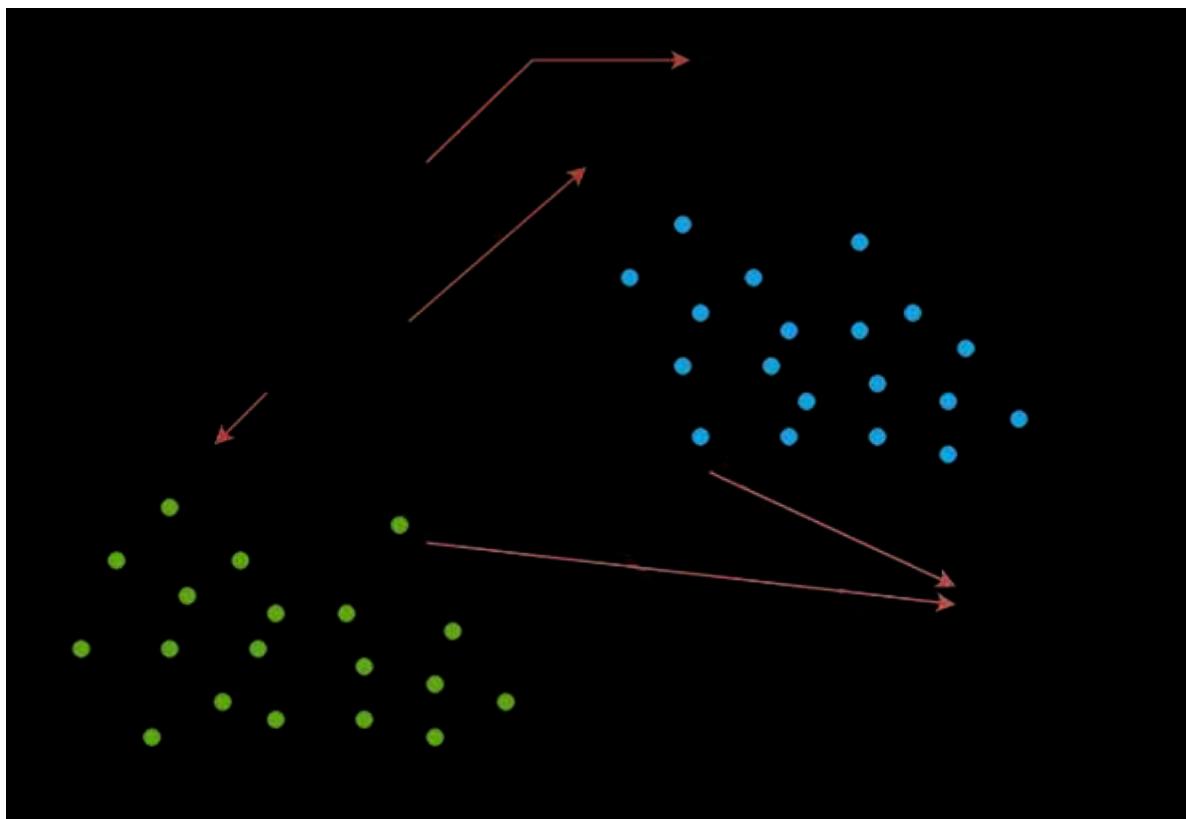
Tại sao SVM lại hiệu quả đến vậy? Bí mật nằm ở việc tối ưu hóa mặt phẳng siêu không gian. SVM không ngừng điều chỉnh để giảm thiểu lỗi, chắc chắn rằng mặt phẳng siêu tối ưu nhất được tạo ra. Ý tưởng trung tâm của SVM là xác định một siêu phẳng có lề tối đa (Maximum Marginal Hyperplane), đảm bảo phân chia dữ liệu một cách tốt nhất. Những điểm dữ liệu gần nhất với siêu phẳng, được gọi là Support Vectors, đóng một vai trò quan trọng trong việc xây dựng bộ phân loại. Chính những điểm này định rõ đường phân chia tốt nhất bằng cách tính toán các khoảng cách lề.

Khái niệm khác quan trọng bao gồm Hyperplane, một mặt phẳng quyết định phân loại dữ liệu, và Margin, khoảng cách giữa hai đường trên các điểm dữ liệu gần nhất của các

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

lớp khác nhau. Margin lớn cho thấy sự phân biệt tốt giữa các lớp, trong khi một Margin nhỏ có thể chỉ ra một phân loại kém hiệu quả.

Với cách tiếp cận thông minh này, SVM đảm bảo dữ liệu được phân loại một cách chính xác nhất, tối ưu hóa hiệu suất và độ chính xác cho cả bài toán phân loại và hồi quy.



Hình 4 Mô tả thuật toán Support Vector Machines [7]

2.1.4 Cách Support Vector Machines hoạt động

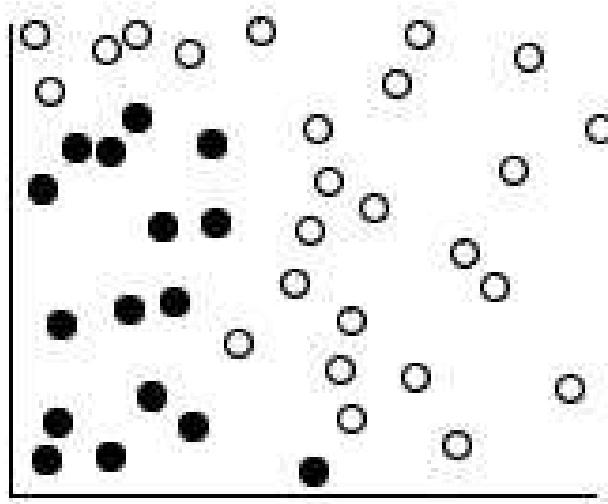
SVM hoạt động bằng cách ánh xạ dữ liệu lên một không gian đặc trưng nhiều chiều sao cho các điểm dữ liệu có thể được phân loại, kể cả khi dữ liệu không thể được tách biệt một cách tuyến tính. Một biên phân chia giữa các nhóm được xác định, sau đó dữ liệu được biến đổi theo cách mà biên phân chia có thể được vẽ dưới dạng một siêu mặt phẳng. Theo sau đó, đặc điểm của dữ liệu mới có thể được sử dụng để dự đoán nhóm mà bản ghi mới nêu thuộc về.

Lý thuyết là thế, giả sử như ta đang chơi một trò chơi trực tuyến nơi người chơi cần phân loại các bức ảnh là của chó hoặc mèo dựa trên một số đặc trưng như màu sắc, kích thước, vị trí của tai, v.v. Ta thấy rằng một số bức ảnh khá khó để phân loại chỉ dựa vào những đặc trưng này vì một số chó có nhiều đặc trưng giống mèo và ngược lại. Bây giờ, hãy tưởng tượng rằng SVM chính là một kính hiển vi kỳ diệu. Khi ta nhìn qua kính hiển vi này, ta không chỉ thấy màu sắc và kích thước của chó và mèo, mà còn thấy cả "hình

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

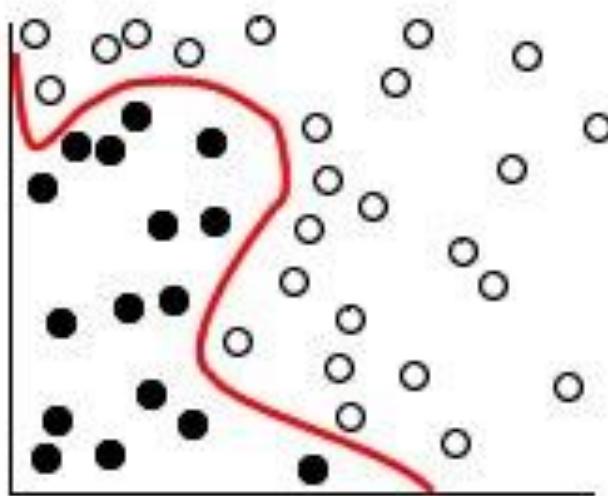
dáng" của tiếng gọi hoặc "cảm xúc" trên khuôn mặt của chúng, điều mà mắt thường không thể nhận biết. Với kính hiển vi này (SVM), ta có thể dễ dàng vẽ một đường phân chia giữa chó và mèo mà trước đó ta không thể làm được. Và sau này, mỗi khi ta gặp một bức ảnh mới, ta chỉ cần nhìn qua "kính hiển vi" (SVM) để xác định nó là hình ảnh của chó hay mèo. Ở đây, "kính hiển vi kỳ diệu" chính là việc SVM ánh xạ dữ liệu lên một không gian đặc trưng cao chiều, giúp ta nhìn thấy các đặc trưng khác nhau giữa chó và mèo mà trước đó ta không thể nhận biết.

Xem xét hình dưới đây, dữ liệu được chia thành hai loại chấm tròn.



Hình 5 Bộ dữ liệu gốc [8]

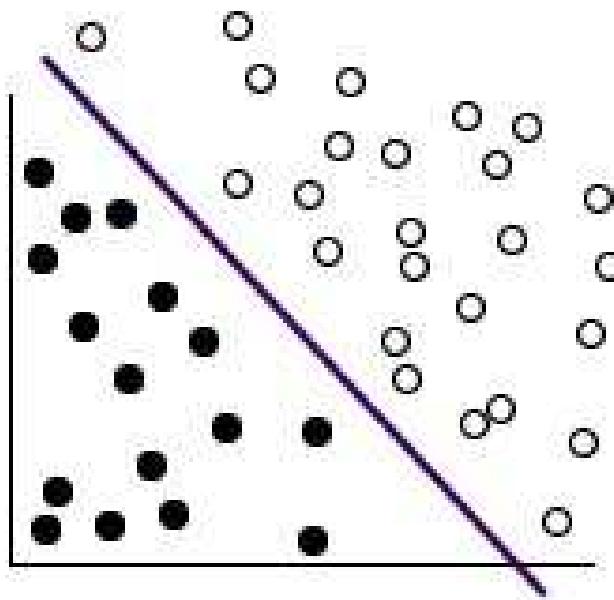
Hai loại chấm tròn có thể được tách biệt bằng một đường cong.



NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

Hình 6 Dữ liệu với bộ tách là đường cong đã được thêm vào [8]

Sau quá trình chuyển đổi, biên giữa hai loại chấm tròn có thể được xác định bởi một siêu phẳng.



Hình 7 Dữ liệu đã được phân chia [8]

Hàm toán học được sử dụng để phân chia được biết đến với tên là hàm kernel.

2.1.5 Support Vector Machines Kernels

Thuật toán SVM được thực hiện trong thực tế bằng cách sử dụng một bộ lõi (kernel). Một kernel biến đổi không gian dữ liệu đầu vào thành dạng cần thiết. SVM sử dụng một kỹ thuật gọi là "kernel trick". Ở đây, kernel nhận một không gian đầu vào có số chiều thấp và biến đổi nó thành một không gian có số chiều cao hơn. Nói cách khác, ta có thể nói rằng nó chuyển đổi vấn đề không tách biệt thành vấn đề có thể tách biệt bằng cách thêm nhiều chiều vào đó. Nó rất hữu ích trong vấn đề tách biệt không tuyến tính. Mẹo kernel giúp ta xây dựng một trình phân loại chính xác hơn.

2.1.5.1 Linear Kernel

Linear Kernel có thể được sử dụng như một tích vô hướng thông thường giữa hai điểm bất kỳ được cho trước. Tích giữa hai véc-tor là tổng của phép nhân của mỗi cặp giá trị đầu vào. Ta có thể hình dung nó giống như việc vẽ một đường thẳng giữa hai điểm trên một mặt phẳng và đo độ dài của đường thẳng đó. Trong không gian hai chiều, công thức tính toán sự tương tự này chính là tích vô hướng (dot product) của hai véc-tor đại diện

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

cho hai điểm dữ liệu. Nếu hai điểm dữ liệu giống nhau hoặc nằm trên cùng một đường thẳng, Linear Kernel sẽ trả về một giá trị lớn, còn nếu chúng khác biệt nhiều, giá trị sẽ nhỏ hơn.

Ví dụ về Linear Kernel bằng Python:

```
import numpy as np
```

```
def linear_kernel(x1, x2):
```

```
    return np.dot(x1, x2)
```

```
vector1 = np.array([2, 3])
```

```
vector2 = np.array([4, 5])
```

```
result = linear_kernel(vector1, vector2)
```

```
print(result)
```

Kết quả sẽ là 23 ($2 \cdot 4 + 3 \cdot 5$).

2.1.5.2 Polynomial Kernel

Polynomial Kernel là một dạng tổng quát hơn của Linear Kernel. Thay vì chỉ tính tích vô hướng giữa hai vec-tor, Polynomial Kernel sẽ nâng giá trị lên một số lượng cần thiết, tạo ra một độ phức tạp đa thức. Polynomial Kernel giống như việc ta đo sự tương tự giữa hai điểm dữ liệu không chỉ dựa trên khoảng cách thẳng giữa chúng, mà còn dựa trên cách chúng tương tác với nhau ở các cấp độ phức tạp khác nhau. Ta có thể hình dung nó như việc đo sự tương tự giữa hai điểm dữ liệu qua một lăng kính phóng đại nhiều lớp.

Ví dụ về Polynomial Kernel bằng Python:

```
import numpy as np
```

```
def polynomial_kernel(x1, x2, d=2, c=1):
```

```
    """
```

Tính toán Polynomial Kernel giữa hai vec-tor x1 và x2.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
:param x1: véc-tor thứ nhất
:param x2: véc-tor thứ hai
:param d: độ của đa thức (mặc định là 2)
:param c: hệ số tự do (mặc định là 1)
:return: giá trị của Polynomial Kernel
"""
return (np.dot(x1, x2) + c) ** d
```

```
vector1 = np.array([2, 3])
vector2 = np.array([4, 5])
result = polynomial_kernel(vector1, vector2, d=3)
print(result)
```

Kết quả sẽ dựa trên giá trị của $(2*4 + 3*5 + 1)^3$.

2.1.5.3 Radial Basis Function Kernel

Radial Basis Function là một kernel có thể ánh xạ một không gian đầu vào đi vào không gian với số chiều vô hạn.

Ví dụ về Radial Basis Function bằng Python:

```
import numpy as np
```

```
def rbf_kernel(x1, x2, sigma=1):
"""
Tính toán RBF Kernel giữa hai véc-tor x1 và x2.
```

```
:param x1: véc-tor thứ nhất
:param x2: véc-tor thứ hai
:param sigma: tham số sigma (mặc định là 1)
:return: giá trị của RBF Kernel
"""
distance = np.linalg.norm(x1 - x2) ** 2
return np.exp(-distance / (2 * sigma ** 2))
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
vector1 = np.array([2, 3])
vector2 = np.array([4, 5])
result = rbf_kernel(vector1, vector2)
print(result)
```

2.1.5.4 Sigmoid Kernel

Sigmoid Kernel là một kernel mô phỏng hàm Sigmoid, giống như trong mạng nơ-ron nhân tạo nhằm để biến đổi dữ liệu.

Ví dụ về Sigmoid Kernel bằng Python:

```
import numpy as np
```

```
def sigmoid_kernel(x1, x2, alpha=1, c=0):
```

```
    """
```

Tính toán Sigmoid Kernel giữa hai véc-tor x_1 và x_2 .

:param x_1 : véc-tor thứ nhất

:param x_2 : véc-tor thứ hai

:param α : hệ số α (mặc định là 1)

:param c : hệ số tự do c (mặc định là 0)

:return: giá trị của Sigmoid Kernel

```
    """
```

```
    return np.tanh(alpha * np.dot(x1, x2) + c)
```

```
vector1 = np.array([2, 3])
vector2 = np.array([4, 5])
result = sigmoid_kernel(vector1, vector2)
print(result)
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

2.1.6 Xây dựng bộ phân loại Support Vector Machines bằng Scikit-learn

2.1.6.1 Nạp dữ liệu

Đầu tiên, ta cần nạp dữ liệu. Để làm việc này, Scikit-learn có một số tập dữ liệu có sẵn, hoặc ta cũng có thể nạp dữ liệu từ nguồn khác như CSV.

Ví dụ, nạp dữ liệu từ tập dữ liệu ‘iris’:

```
from sklearn import datasets
```

```
iris = datasets.load_iris()
```

```
X = iris.data
```

```
y = iris.target
```

2.1.6.2 Khám phá dữ liệu

Để hiểu rõ hơn về dữ liệu.

```
import pandas as pd
```

```
df = pd.DataFrame(data=iris.data, columns=iris.feature_names)
```

```
df['target'] = iris.target
```

```
print(df.head())
```

2.1.6.3 Chia dữ liệu

Trước khi xây dựng mô hình, ta cần chia dữ liệu thành hai phần: tập huấn luyện và tập kiểm tra.

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,  
random_state=42)
```

2.1.6.4 Tạo mô hình

Ta sẽ sử dụng Support Vector Machines từ Scikit-learn.

```
from sklearn.svm import SVC
```

```
model = SVC(kernel='linear', C=1)
```

```
model.fit(X_train, y_train)
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

2.1.6.5 Đánh giá mô hình

Để đánh giá mô hình, ta sẽ dự đoán dữ liệu từ tập kiểm tra và so sánh kết quả với nhãn thực tế.

```
y_pred = model.predict(X_test)
```

```
from sklearn.metrics import classification_report, accuracy_score
```

```
print(classification_report(y_test, y_pred))
```

```
print("Độ chính xác:", accuracy_score(y_test, y_pred))
```

2.1.6.6 Tinh chỉnh mô hình

Đôi khi, ta cần tinh chỉnh các tham số để cải thiện hiệu suất. Ta có thể sử dụng GridSearchCV từ Scikit-learn để làm điều này.

```
from sklearn.model_selection import GridSearchCV
```

```
param_grid = {
```

```
    'C': [0.1, 1, 10, 100],
```

```
    'kernel': ['linear', 'rbf', 'poly'],
```

```
    'degree': [1, 2, 3, 4],
```

```
    'gamma': ['scale', 'auto']
```

```
}
```

```
grid_search = GridSearchCV(SVC(), param_grid, cv=5)
```

```
grid_search.fit(X_train, y_train)
```

```
print("Tham số tốt nhất:", grid_search.best_params_)
```

2.1.6.7 Hiển thị kết quả

Sau khi tinh chỉnh mô hình, ta có thể trực quan hóa kết quả để dễ dàng đánh giá hiệu suất của mô hình. Đoạn mã dưới đây sẽ hiển thị ma trận nhầm lẫn, giúp ta nhanh chóng thấy số lượng dự đoán đúng và sai của mô hình.

```
import matplotlib.pyplot as plt
```

```
from sklearn.metrics import plot_confusion_matrix
```

```
plot_confusion_matrix(grid_search, X_test, y_test, cmap=plt.cm.Blues)
plt.show()
```

2.1.6.8 Lưu và tải mô hình

Một khi ta đã huấn luyện xong mô hình, ta có thể muốn lưu mô hình đó để sử dụng lại sau này mà không cần phải huấn luyện lại từ đầu.

```
import joblib
```

```
joblib.dump(grid_search.best_estimator_, 'svm_model.pkl')
```

```
loaded_model = joblib.load('svm_model.pkl')
```

2.1.6.9 Sử dụng mô hình

Cuối cùng, ta cũng có thể muốn sử dụng mô hình đã huấn luyện để dự đoán dữ liệu mới.

```
new_data = [[5.1, 3.5, 1.4, 0.2]]
prediction = loaded_model.predict(new_data)
print(f"Dự đoán cho dữ liệu mới: {prediction}")
```

2.1.7 Điều chỉnh tham số Hyperparameter trong Support Vector Machines

2.1.7.1 Kernel

Kernel là thành phần mấu chốt trong SVM, chịu trách nhiệm biến đổi dữ liệu đầu vào sang một không gian mới, trong đó dữ liệu có thể được phân loại dễ dàng hơn. Có nhiều lựa chọn về kernel:

- Tuyến Tính (Linear): Đây là lựa chọn đơn giản nhất, không yêu cầu bất kỳ sự biến đổi nào phức tạp.
- Đa Thức (Polynomial): Cho phép mô hình xử lý các mối quan hệ phức tạp và không tuyến tính.
- Radial Basis Function (RBF): Rất mạnh mẽ, cho phép mô hình tạo ra các biên phân loại phức tạp.

Lựa chọn kernel phụ thuộc vào đặc điểm của tác vụ phân loại, và việc sử dụng các kernel phức tạp hơn thường yêu cầu thêm thời gian và tài nguyên tính toán.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

2.1.7.2 Chính Quy Hóa (Regularization)

Chính quy hóa là một kỹ thuật quan trọng trong SVM. Tham số C trong Scikit-learn giúp ngăn chặn tình trạng overfitting bằng cách áp đặt hình phạt cho sai số. Tham số này cần được điều chỉnh cẩn thận:

- Giá trị C thấp: Tạo ra một mô hình có biên phân loại lớn, nhưng có thể chấp nhận một số sai phạm.
- Giá trị C cao: Tạo ra một mô hình phân loại chính xác hơn nhưng biên phân loại có thể trở nên hẹp.

2.1.7.3 Gamma

Tham số gamma quy định mức độ phụ thuộc vào các điểm dữ liệu cá nhân khi tính toán biên phân loại:

- Gamma thấp: Mô hình không quá phụ thuộc vào các điểm dữ liệu cá nhân, có thể bỏ sót một số thông tin nhưng tránh được overfitting.
- Gamma cao: Mô hình trở nên cực kỳ nhạy cảm với từng điểm dữ liệu, có thể dẫn đến overfitting.

2.1.8 Ưu Điểm của Support Vector Machines

- Hiệu Quả Trong Không Gian Đa Chiều:

SVM nổi bật trong việc xử lý dữ liệu đa chiều. Khả năng này cho phép nó dễ dàng làm việc với dữ liệu có số lượng chiều lớn mà không gặp nhiều khó khăn.

- Khả Năng Tống Quát Hóa Xuất Sắc:

SVM giảm thiểu nguy cơ quá mức (overfitting) bằng cách tối ưu hóa lề giữa các lớp. Điều này giúp cải thiện khả năng tổng quát hóa, giúp mô hình hoạt động tốt trên dữ liệu chưa từng thấy.

- Thích Ứng Với Biên Non-linear:

Đối với các trường hợp biên lớp không tuyến tính, SVM vẫn hoạt động hiệu quả bằng cách sử dụng kỹ thuật kernel. Điều này cho phép nó làm việc trong không gian đặc trưng được biến đổi mà không yêu cầu tính toán trực tiếp tọa độ dữ liệu.

- Hiệu Quả Về Bộ Nhớ:

SVM chỉ sử dụng một tập con của dữ liệu huấn luyện (các vector hỗ trợ) trong quá trình ra quyết định, do đó giúp tiết kiệm bộ nhớ đáng kể.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

- Kháng Lại Overfitting:

SVM hiệu quả trong việc ngăn chặn overfitting, đặc biệt khi làm việc với dữ liệu đa chiều.

2.1.9 Nhược Điểm của Support Vector Machines

- Độ Phức Tạp Tính Toán Cao:

Thời gian huấn luyện tăng lên đáng kể với các tập dữ liệu lớn, khiến SVM không phù hợp cho các tác vụ học máy quy mô lớn.

- Nhạy Cảm Với Nhiễu:

SVM có thể bị ảnh hưởng nghiêm trọng bởi nhiễu, nhất là khi các mẫu dữ liệu được gán nhãn sai.

- Khó Khăn Trong Lựa Chọn Kernel:

Việc tìm ra kernel phù hợp là một thách thức lớn, và hiệu suất của SVM phụ thuộc nhiều vào việc lựa chọn kernel đó.

- Khả Năng Mở Rộng Hạn Chế:

Khả năng mở rộng của SVM bị giới hạn, khiến cho việc xử lý dữ liệu lớn trở nên khó khăn hơn, tiêu tốn nhiều thời gian và nguồn lực.

- Khó Khăn Trong Việc Giải Thích:

Các mô hình SVM, đặc biệt khi sử dụng các kernel không tuyến tính, thường khó hiểu và giải thích, gây rắc rối khi sử dụng trong các lĩnh vực không chuyên môn.

2.2 HỌC MÁY DECISION TREES

2.2.1 Giới thiệu

Decision Trees (Cây quyết định) là một phương pháp học máy phổ biến và hiệu quả, được ứng dụng rộng rãi trong nhiều lĩnh vực khác nhau. Cây quyết định thuộc nhóm thuật toán học có giám sát và có thể giải quyết cả bài toán phân lớp và hồi quy. Cây quyết định được xây dựng dựa trên việc chia nhỏ tập dữ liệu ban đầu thành các nhóm con, sao cho mỗi nhóm con có tính chất đồng nhất cao nhất. Mỗi quá trình chia này được thể hiện dưới dạng một cấu trúc cây.

2.2.2 *Lịch sử phát triển*

Nhiều người dẫn nguồn gốc từ bài báo của Ronald Fisher vào năm 1936. Dự án AID của Morgan và Sonquist và công bố năm 1966 của Hunt cũng đã đóng vai trò quan trọng. Trong tâm lý học, phương pháp cây quyết định được sử dụng để mô phỏng khái niệm học của con người. Dọc theo hướng này, các nhà nghiên cứu phát hiện ra thuật toán này hữu ích cho lập trình. Vào năm 1972, cây phân loại đầu tiên xuất hiện trong dự án THAID do Messenger và Mandell thực hiện. Giáo sư Thống kê Berkeley, Leo Breiman và Charles Joel Stone, cùng với Jerome H. Friedman và Richard Olshen từ Đại học Stanford, sau đó bắt đầu phát triển thuật toán cây phân loại và hồi quy (CART). Thuật toán này được công bố vào năm 1977. [9]

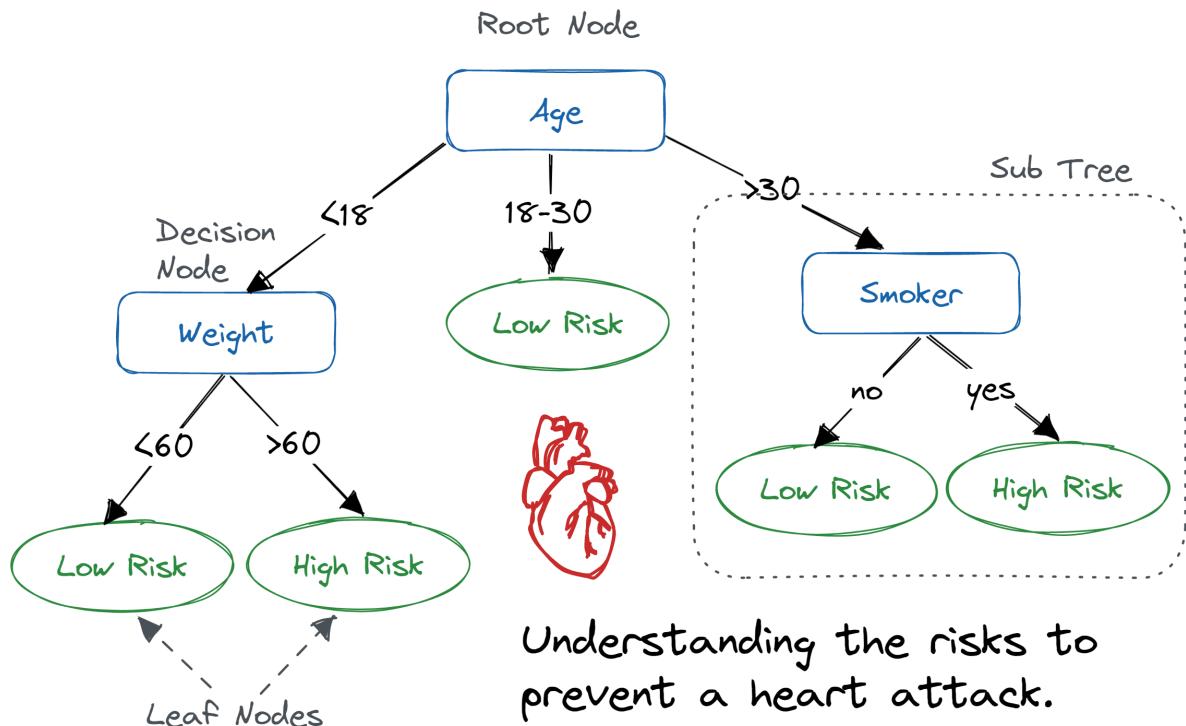
2.2.3 *Tổng quan*

Cây quyết định là một cấu trúc tương tự như sơ đồ luồng, nơi mỗi nút nội bộ tượng trưng cho một đặc điểm, mỗi nhánh biểu diễn một quy tắc quyết định và mỗi nút lá thể hiện một kết quả. Tại vị trí đầu tiên của cây là nút gốc, nơi bắt đầu quá trình phân loại dựa trên các giá trị của thuộc tính. Cây quyết định hoạt động một cách đệ quy, tức là quá trình phân loại được thực hiện lặp đi lặp lại cho đến khi đạt được kết quả cuối cùng.

Đặc biệt, cây quyết định mô phỏng cách suy nghĩ của con người trong quá trình ra quyết định, làm cho nó trở nên dễ hiểu và dễ giải thích. Điều này tạo nên sự phân biệt của nó so với các thuật toán học máy hộp đen khác, như mạng nơ-ron, nơi logic nội bộ không dễ dàng được minh bạch.

Bên cạnh đó, thời gian huấn luyện của cây quyết định thường nhanh hơn so với các mô hình học sâu, và độ phức tạp của nó là hàm của số lượng dữ liệu và thuộc tính. Cây quyết định không yêu cầu giả định về phân phối xác suất của dữ liệu và có khả năng xử lý dữ liệu đa chiều mà vẫn duy trì độ chính xác cao.

Như vậy, cây quyết định không chỉ là một thuật toán mạnh mẽ cho việc phân loại và học máy, mà còn là một phương tiện hữu ích để minh họa và giải thích quá trình ra quyết định.



Hình 8 Cây quyết định trong việc nhận thức phòng chống bệnh đau tim [10]

2.2.4 Cách Decision Trees hoạt động

Cây quyết định là một cấu trúc cây mà mỗi nút đại diện cho một thuộc tính (hoặc đặc điểm) của dữ liệu, mỗi nhánh đại diện cho một quyết định dựa trên giá trị của thuộc tính đó, và mỗi nút lá đại diện cho một dự đoán hoặc kết quả.

- Khởi tạo cây: Thuật toán bắt đầu bằng cách xác định nút gốc, chứa toàn bộ tập dữ liệu. Mục tiêu ở đây là xác định thuộc tính nào sẽ được sử dụng để phân chia dữ liệu tại nút này.
- Lựa chọn thuộc tính: Dùng Tiêu chí Lựa chọn Thuộc tính (Attribute Selection Measure) như Entropy, Gain, Gini Index,... để tìm thuộc tính tốt nhất để phân chia tập dữ liệu. Thuộc tính được chọn sẽ là thuộc tính giúp tối ưu hóa việc phân loại dữ liệu.
- Phân chia dữ liệu: Dựa vào giá trị của thuộc tính đã chọn, tập dữ liệu sẽ được chia thành các tập con. Mỗi tập con tương ứng với một giá trị cụ thể của thuộc tính.
- Xây dựng nút và nhánh: Với mỗi giá trị của thuộc tính, một nhánh sẽ được tạo ra từ nút hiện tại. Nút tiếp theo trên nhánh đó sẽ chứa tập con dữ liệu tương ứng.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

- Lặp lại cho đến nút lá: Quá trình trên sẽ được lặp lại cho mỗi nút mới cho đến khi đạt đến một điều kiện dừng nhất định - chẳng hạn như tập dữ liệu tại nút đó thuộc về một lớp duy nhất, hoặc đã đạt đến độ sâu tối đa của cây. Khi đó, nút sẽ trở thành nút lá, và chứa một dự đoán hoặc kết quả.
- Đánh giá và cắt tỉa cây: Để tránh hiện tượng quá khớp, sau khi xây dựng cây, một số kỹ thuật cắt tỉa có thể được áp dụng để loại bỏ những nhánh không cần thiết, giúp mô hình tổng quát hóa tốt hơn trên dữ liệu chưa được nhìn thấy.

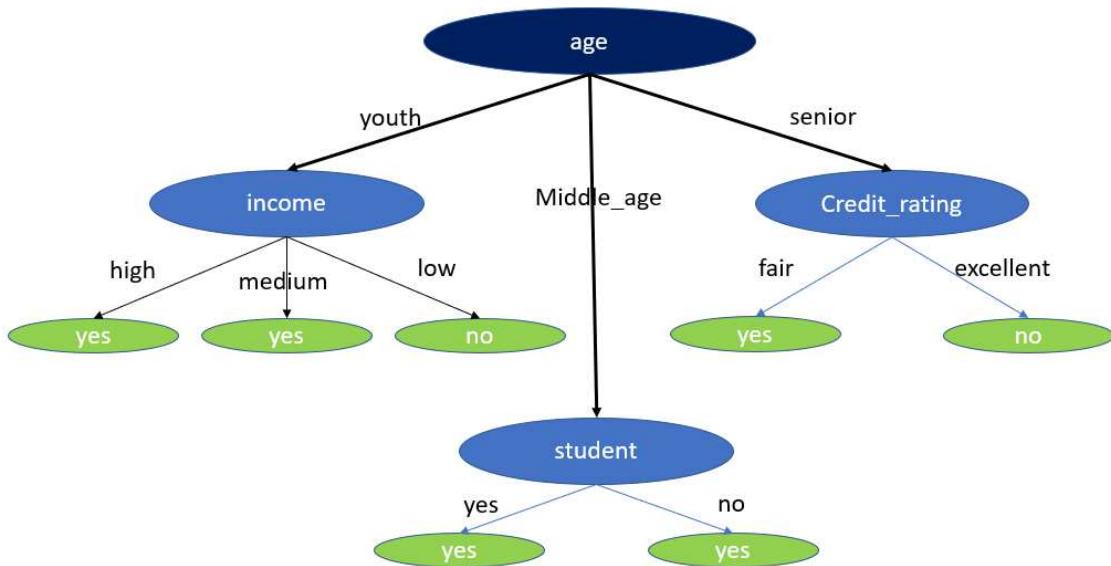
Ví dụ ta có các dataset như sau:

R_ID	Age	Income	Student	Credit_rating	Buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Table: Class-labeled training tuples from AllElectronics customer database.

Hình 9 Các dataset huấn luyện được gắn nhãn từ cơ sở dữ liệu khách hàng [11]

Cây quyết định từ các dataset đó sẽ được vẽ như sau:



Hình 10 Cây quyết định từ các dataset [11]

2.2.5 Các biện pháp chọn lựa thuộc tính

2.2.5.1 Information Gain

Claude Shannon là người sáng tạo ra khái niệm về entropy, một đại lượng đo lường mức độ không thuần khiết của một tập hợp dữ liệu đầu vào. Trong lý thuyết vật lý và toán học, entropy thường được hiểu là đại lượng biểu hiện sự ngẫu nhiên hoặc sự không thuần khiết trong một hệ thống. Nhưng trong lý thuyết thông tin, entropy nói lên sự không thuần khiết trong một nhóm ví dụ hoặc dữ liệu. [12]

Information Gain là sự giảm entropy trong dữ liệu. Một cách cụ thể hơn, nó tính toán sự chênh lệch giữa entropy trước khi tách và entropy trung bình sau khi tách dữ liệu dựa trên các giá trị thuộc tính đã cho.

Có thể hình dung, nếu chúng ta đang cố gắng phân loại một tập dữ liệu và chọn ra một thuộc tính nào đó để tách dữ liệu thành các nhóm con, Information Gain sẽ giúp chúng ta đánh giá xem việc tách dữ liệu dựa trên thuộc tính đó có làm giảm sự không chắc chắn (hay sự không thuần khiết) của dữ liệu không.

Công thức tính Information Gain cho thuộc tính A là:

$$\text{Information Gain}(D, A) = \text{Entropy}(D) - \sum \left(\frac{|D_v|}{|D|} \times \text{Entropy}(D_v) \right)$$

Trong đó:

- D là tập dữ liệu hiện tại.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

- D_v là tập con của D tương ứng với giá trị v của thuộc tính A .
- $|D|$ là số lượng mẫu trong D .
- $|D_v|$ là số lượng mẫu trong D_v .

Công thức tính Entropy cho tập dữ liệu D là:

$$\text{Entropy}(D) = - \sum_{i=1}^m p_i \log_2 p_i$$

Trong đó:

- p_i là xác suất của một mẫu thuộc lớp thứ i trong tập dữ liệu D .
- m là số lớp khác nhau trong tập dữ liệu.

2.2.5.2 Gain Ratio

Gain Ratio cũng dựa trên nền tảng của entropy, nhưng nó được thiết kế để giảm bớt sự thiên vị của Information Gain đối với các thuộc tính có nhiều giá trị. Điều này được thực hiện bằng cách chia Information Gain cho một giá trị gọi là Split Information, một đại lượng đo lường mức độ "phân tán" khi sử dụng một thuộc tính cụ thể để phân chia dữ liệu.

Một cách hình dung, Gain Ratio không chỉ xem xét việc giảm sự không thuần khiết khi tách dữ liệu theo một thuộc tính, mà còn cân nhắc việc tách dữ liệu theo thuộc tính đó có mang lại thông tin hữu ích không.

Công thức tính Gain Ratio cho thuộc tính A là:

$$\text{Gain Ratio}(D, A) = \frac{\text{Information Gain}(D, A)}{\text{Split Information}(A)}$$

Split Information cho một thuộc tính cung cấp một độ đo về cách tập dữ liệu được chia dựa trên thuộc tính đó, giống như cách Entropy đo độ không chắc chắn của tập dữ liệu.

Công thức tính Split Information cho thuộc tính A với tập dữ liệu D là:

$$\text{Split Information}(A) = - \sum_{v \in \text{Values}(A)} \left(\frac{|D_v|}{|D|} \right) \log_2 \left(\frac{|D_v|}{|D|} \right)$$

Trong đó:

- $|D|$ là số lượng mẫu trong D .
- $|D_v|$ là số lượng mẫu trong tập con D_v của D tương ứng với giá trị v của thuộc tính A .
- $\text{Values}(A)$ là tập hợp tất cả các giá trị có thể của thuộc tính A .

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

2.2.5.3 Gini Index

Khác với entropy, Gini Index là một đại lượng đo lường mức độ không thuần khiết của dữ liệu dựa trên một cách tiếp cận khác. Trong lý thuyết tài chính và kinh tế, Gini Index thường được sử dụng để đo mức độ bất bình đẳng. Nhưng trong lý thuyết thông tin, Gini Index đo lường mức độ không thuần khiết của một tập hợp dữ liệu. Gini Index được tính toán dựa trên xác suất của mỗi lớp và sự không thuần khiết tối đa xảy ra khi tất cả các lớp có cùng xác suất xuất hiện.

Để phân loại một tập dữ liệu, nếu chúng ta sử dụng một thuộc tính để tách dữ liệu thành các nhóm con, Gini Index sẽ giúp chúng ta đánh giá xem việc tách dữ liệu theo thuộc tính đó có làm giảm sự không thuần khiết của dữ liệu không. Thuật toán cây quyết định như CART (Classification And Regression Trees) thường sử dụng Gini Index như một tiêu chí để chọn thuộc tính tách.

Công thức tính Gini Index với tập dữ liệu D :

$$Gini(D) = 1 - \sum_{i=1}^m p_i^2$$

Trong đó:

- p_i là xác suất của một mẫu thuộc lớp thứ i trong tập dữ liệu D .
- m là số lớp khác nhau trong tập dữ liệu.

Gini Index có giá trị từ 0 (tập dữ liệu hoàn toàn thuần túy, tất cả mẫu thuộc cùng một lớp) đến 0.5 (trong trường hợp phân loại nhị phân và mỗi lớp có xác suất xuất hiện là 0.5) hoặc $1-1/m$ (trong trường hợp phân loại đa lớp và mỗi lớp có xác suất xuất hiện bằng nhau).

2.2.6 Xây dựng bộ phân loại Decision Trees bằng Scikit-learn

2.2.6.1 Nhập thư viện cần thiết

Trước tiên, ta cần nhập các thư viện cần thiết.

```
import pandas as pd  
from sklearn.model_selection import train_test_split  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix  
from sklearn import tree  
from sklearn.model_selection import GridSearchCV
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
from sklearn.model_selection import cross_val_score  
import matplotlib.pyplot as plt
```

2.2.6.2 Nạp dữ liệu

Tiếp theo, ta cần nạp dữ liệu để xây dựng mô hình Decision Tree. Đảm bảo ta đã có tệp dữ liệu CSV hoặc DataFrame để làm việc với nó.

```
data = pd.read_csv('decision_trees.csv')
```

2.2.6.3 Lựa chọn đặc trưng

Nếu cần, ta có thể lựa chọn các đặc trưng quan trọng từ dữ liệu.

```
X = data[['feature1', 'feature2', 'feature3']]
```

```
y = data['target']
```

2.2.6.4 Chia dữ liệu

Chia dữ liệu thành hai phần: một phần để huấn luyện mô hình và một phần để kiểm tra mô hình.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,  
random_state=42)
```

2.2.6.5 Xây dựng mô hình

Bây giờ, ta có thể xây dựng mô hình Decision Tree bằng cách sử dụng dữ liệu huấn luyện.

```
clf = DecisionTreeClassifier()
```

```
clf.fit(X_train, y_train)
```

2.2.6.6 Đánh giá mô hình

Sau khi xây dựng mô hình, ta cần đánh giá hiệu suất của nó bằng các độ đo như độ chính xác, báo cáo phân loại và ma trận nhầm lẫn.

```
y_pred = clf.predict(X_test)
```

```
accuracy = accuracy_score(y_test, y_pred)
```

```
classification_rep = classification_report(y_test, y_pred)
```

```
confusion_mat = confusion_matrix(y_test, y_pred)
```

```
print(f'Dộ chính xác: {accuracy}')
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
print(f'Báo cáo phân loại:\n{classification_rep}')
print(f'Ma trận nhầm lẫn:\n{confusion_mat}'')
```

2.2.6.7 Hiển thị cây quyết định

Ta có thể muốn hiển thị cây quyết định để hiểu cách mô hình hoạt động.

```
fig, ax = plt.subplots(figsize=(12, 12))
tree.plot_tree(clf, filled=True, feature_names=X.columns, class_names=True)
plt.show()
```

2.2.6.8 Điều chỉnh siêu tham số

Điều chỉnh siêu tham số của mô hình Decision Trees có thể cải thiện hiệu suất của nó.

Ta có thể sử dụng Grid Search hoặc Random Search để tìm các giá trị tối ưu cho các tham số như độ sâu của cây (max_depth), số lượng mẫu tối thiểu để chia một nút (min_samples_split), và số lượng mẫu tối thiểu trong mỗi lá (min_samples_leaf).

```
# Định nghĩa danh sách các siêu tham số cần điều chỉnh và các giá trị thử nghiệm
param_grid = {
```

```
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}
```

```
# Tạo Grid Search với mô hình Decision Tree
```

```
grid_search = GridSearchCV(DecisionTreeClassifier(), param_grid, cv=5)
grid_search.fit(X_train, y_train)
```

```
# Lấy ra các giá trị tối ưu của siêu tham số
```

```
best_params = grid_search.best_params_
best_estimator = grid_search.best_estimator_
```

2.2.6.9 Kiểm định chéo

Kiểm định chéo (cross-validation) là một phương pháp quan trọng để đánh giá mô hình và đảm bảo tính tổng quát của nó. Ta có thể sử dụng kiểm định chéo để đánh giá hiệu suất của mô hình trên nhiều tập dữ liệu con khác nhau từ tập huấn luyện.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

Tạo một mô hình Decision Tree

clf = `DecisionTreeClassifier(max_depth=3)`

Sử dụng kiểm định chéo để đánh giá mô hình

scores = `cross_val_score(clf, X_train, y_train, cv=5)`

mean_accuracy = scores.mean()

std_deviation = scores.std()

`print(f'Điểm độ chính xác trung bình qua kiểm định chéo: {mean_accuracy}')`

`print(f'Dộ lệch chuẩn qua kiểm định chéo: {std_deviation}')`

2.2.7 Tối ưu hóa hiệu suất Decision Trees

Để tối ưu hóa hiệu suất của Decision Trees, ta cần xem xét một số tham số quan trọng:

- Tiêu chí chọn lựa thuộc tính (criterion): Mặc định là "gini". Tham số này cho phép chúng ta sử dụng các phép đo lựa chọn thuộc tính khác nhau. Các tiêu chí được hỗ trợ là "gini" cho chỉ số Gini và "entropy" cho lượng thông tin thu được.
- Chiến lược chia (splitter): Mặc định là "best". Tham số này cho phép ta lựa chọn chiến lược chia. Các chiến lược được hỗ trợ là "best" để chọn phép chia tốt nhất và "random" để chọn phép chia ngẫu nhiên tốt nhất.
- Độ sâu tối đa của cây (max_depth): Mặc định là None. Đây là độ sâu tối đa của cây. Nếu là None, thì các nút sẽ được mở rộng cho đến khi tất cả các lá chứa ít hơn min_samples_split mẫu. Giá trị cao của độ sâu tối đa gây ra hiện tượng quá khớp, và giá trị thấp gây ra hiện tượng không đủ khớp.

Trong Scikit-learn, việc tối ưu hóa bộ phân loại Decision Trees chỉ được thực hiện bằng cách cắt tỉa trước. Độ sâu tối đa của cây có thể được sử dụng như một biến kiểm soát cho việc cắt tỉa trước.

Ví dụ:

Tạo đối tượng Cây Quyết Định

clf = `DecisionTreeClassifier(criterion="entropy", max_depth=3)`

Huấn luyện Cây Quyết Định

clf = clf.fit(X_train,y_train)

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
# Dự đoán kết quả cho bộ dữ liệu kiểm tra
```

```
y_pred = clf.predict(X_test)
```

```
# Độ chính xác của mô hình
```

```
print("Độ chính xác:", metrics.accuracy_score(y_test, y_pred))
```

Ta có thể biểu diễn cây quyết định một cách trực quan với mã sau:

```
from six import StringIO
```

```
from IPython.display import Image
```

```
from sklearn.tree import export_graphviz
```

```
import pydotplus
```

```
dot_data = StringIO()
```

```
export_graphviz(clf, out_file=dot_data,
```

```
    filled=True, rounded=True,
```

```
    special_characters=True, feature_names = feature_cols, class_names=['0','1'])
```

```
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
```

```
graph.write_png('diabetes.png')
```

```
Image(graph.create_png())
```

Ở đây, chúng ta đã thực hiện các bước sau:

- Nhập các thư viện cần thiết.
- Tạo một đối tượng StringIO tên là dot_data để lưu trữ biểu diễn văn bản của cây quyết định.
- Xuất cây quyết định dưới dạng định dạng dot bằng cách sử dụng hàm export_graphviz và ghi kết quả ra dot_data.
- Tạo một đối tượng đồ thị pydotplus từ biểu diễn định dạng dot của cây quyết định được lưu trữ trong dot_data.
- Ghi đồ thị đã tạo ra một tệp PNG tên là "diabetes.png".
- Hiển thị hình ảnh PNG đã tạo của cây quyết định sử dụng đối tượng Image từ mô-đun IPython.display.

2.2.8 Ưu Điểm của Decision Trees

- Dễ Dàng Hiểu và Biểu Diễn:

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

Cây quyết định có thể được biểu diễn trực quan, giúp cho việc diễn giải mô hình trở nên dễ dàng, ngay cả đối với những người không chuyên môn.

- Không Cần Chuẩn Hóa Dữ Liệu:

Ngược lại với nhiều phương pháp học máy khác, cây quyết định không yêu cầu chuẩn hóa dữ liệu trước khi huấn luyện.

- Khả Năng Xử Lý Dữ Liệu Cả Phân Loại và Liên Tục:

Cây quyết định có thể xử lý cả dữ liệu dạng phân loại và dạng liên tục mà không cần thay đổi cơ cấu.

- Xử Lý Dữ Liệu Thiếu:

Cây quyết định có thể xử lý dữ liệu khi một số thuộc tính bị thiếu mà không cần bổ sung hoặc xóa bỏ.

- Tự Động Lựa Chọn Thuộc Tính:

Quá trình xây dựng cây sẽ tự động chọn những thuộc tính quan trọng nhất để phân chia.

- Cắt Tỉa Tự Động:

Có các thuật toán giúp cắt tỉa cây để tránh overfitting và tối ưu hóa kết quả.

2.2.9 Nhược Điểm của Decision Trees

- Dễ Gặp Overfitting:

Cây quyết định, đặc biệt là những cây sâu, dễ gặp tình trạng overfitting, làm giảm khả năng tổng quát hóa trên dữ liệu chưa từng thấy.

- Khả Năng Tổng Quát Hóa Có Thể Kém Hơn Các Mô Hình Khác:

Trong một số trường hợp, mô hình cây quyết định có thể không hoạt động tốt như các mô hình học máy khác như Random Forests hay Gradient Boosted Trees.

- Các Biên Phân Loại Là Orthogonal:

Cây quyết định chỉ thực hiện các phân chia dựa trên giá trị thuộc tính, dẫn đến việc tạo ra các biên phân loại vuông góc, có thể không hiệu quả trong một số trường hợp.

- Khả Năng Mở Rộng:

Với tập dữ liệu lớn, việc xây dựng và đào tạo cây quyết định có thể trở nên khá chậm.

- Độ Nhạy:

Một thay đổi nhỏ trong dữ liệu có thể dẫn đến việc xây dựng một cây hoàn toàn khác, làm giảm tính ổn định của mô hình.

CHƯƠNG 3. XÂY DỰNG CHƯƠNG TRÌNH PHÁT HIỆN WEB PHISHING THÔNG QUA PHÂN TÍCH BỐ CỤC TRANG WEB

3.1 Giới thiệu

"Phát hiện Web Phishing thông qua phân tích bố cục trang web" không chỉ là một đề tài nghiên cứu khoa học mang tính ứng dụng cao, mà còn đáp ứng nhu cầu thực tế trong việc phòng chống các cuộc tấn công từ web phishing. Đặc biệt, việc phân tích bố cục trang web giúp ta nắm bắt được những đặc điểm tiêu biểu của các trang web phishing, từ đó xây dựng các mô hình phát hiện chính xác và kịp thời.

Chương trình WebPhishing.py mà chúng em cung cấp được xây dựng dựa trên hai thuật toán học máy phổ biến: Support Vector Machines và Decision Trees. Cả hai thuật toán này đều đã được chứng minh là hiệu quả trong nhiều lĩnh vực ứng dụng học máy, bao gồm cả bảo mật thông tin. Chương trình được thiết kế để tự động học và phát hiện các trang web phishing dựa trên dữ liệu đầu vào là bố cục trang web, giúp người dùng có thể phát hiện và ngăn chặn kịp thời trước những mối đe dọa tiềm tàng.

Đây là một số ưu điểm chung của Scikit-learn khi xử lý Support Vector Machines và Decision Trees, cùng với lý do tại sao việc không sử dụng GPU có thể là một lựa chọn hợp lý:

- **Độc lập với phần cứng:** Scikit-learn được thiết kế để hoạt động hiệu quả trên CPU mà không cần đến sự hỗ trợ từ GPU. Đối với nhiều bài toán học máy có kích thước dữ liệu nhỏ đến trung bình, việc huấn luyện trên CPU đã đủ nhanh và hiệu quả.
- **Chỉ trọng tâm vào thuật toán cơ bản:** Nhiều thuật toán trong Scikit-learn, bao gồm Support Vector Machines và Decision Trees, không được thiết kế cho việc xử lý song song trên GPU. Trong khi một số thuật toán sâu có lợi ích lớn từ việc sử dụng GPU, Support Vector Machines và Decision Trees thường không cần đến khả năng tính toán song song mạnh mẽ của GPU.
- **Tối ưu hóa cho CPU:** Do không cần phải chú trọng đến tính toán song song của GPU, nhiều thuật toán trong Scikit-learn được tối ưu hóa để hoạt động hiệu quả trên CPU.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

Dù GPU có thể mang lại lợi ích trong việc tăng tốc độ huấn luyện cho một số thuật toán, đặc biệt là trong lĩnh vực học sâu, nhưng với Support Vector Machines và Decision Trees, việc sử dụng CPU đã phù hợp và hiệu quả cho nhiều ứng dụng.

3.2 Thiết lập môi trường

3.2.1 Môi trường phát triển

Để phát triển và triển khai chương trình WebPhishing.py, chúng em đã chọn PyCharm Professional - một môi trường phát triển tích hợp (IDE) chuyên nghiệp dành cho Python. PyCharm Professional không chỉ mang lại một trải nghiệm viết mã trực quan và thuận tiện, mà còn tích hợp sẵn nhiều công cụ giúp quá trình phát triển trở nên linh hoạt và hiệu quả.

3.2.2 Phiên bản Python

Chương trình được xây dựng dưới nền tảng Python 3.11. Đây là phiên bản Python mới nhất và ổn định tại thời điểm chúng em thực hiện dự án, mang lại nhiều cải tiến và tính năng mới. Để chương trình hoạt động mượt mà và hiệu quả, ta nên sử dụng Python 3.11 hoặc các phiên bản sau này.

3.2.3 Bộ dữ liệu

Chúng em sử dụng bộ dữ liệu “Phishing Website HTML Classification” của tác giả Hunter Kempf, tập dữ liệu 800MB này là một bộ các tệp HTML chứa các tệp HTML trang web phishing và trang web không phải phishing. Tác giả tạo ra tập dữ liệu này như một phần của dự án thực tập cho chương trình Thạc sĩ về An ninh mạng của tác giả tại Georgia Tech. [13]

Bộ dữ liệu gồm hai tập dữ liệu là training và validation. Tập training gồm 6139 tệp HTML các trang web không phải phishing và 4216 tệp HTML các trang web là phishing, tập này dùng để huấn luyện trong quá trình máy học. Tập validation gồm 1535 tệp HTML các trang web không phải phishing và 1055 tệp HTML các trang web là phishing, tập này dùng để kiểm thử độ chính xác của mô hình máy học trong quá trình huấn luyện.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

The screenshot shows a Kaggle dataset page. The left sidebar has a 'Datasets' section selected. The main content area is titled 'Phishing Website HTML Classification' with a subtitle 'Full HTML files showing example Phishing and Non-Phishing Websites'. It includes a cover photo of a person fishing at dawn. Below the title are tabs for 'Data Card', 'Code (0)', and 'Discussion (0)'. The 'About Dataset' section contains a brief description of the dataset, its source, and usage statistics (Usability: 6.88). The 'License' section indicates CC BY-SA 3.0. The 'Expected update frequency' is listed as 'Never'. The 'Tags' section includes 'Text', 'Classification', 'Websites', and 'Transformers'. At the bottom, there's a link to 'See what others are saying about this dataset'.

Hình 11 Bộ dữ liệu "Phishing Website HTML Classification" [13]

3.2.4 Thư viện và công cụ

1. csv

Mục đích: Thư viện **csv** giúp ta đọc và ghi dữ liệu dưới định dạng CSV một cách dễ dàng và hiệu quả.

Ứng dụng trong chương trình:

```
import csv
```

2. datetime

Mục đích: Thư viện **datetime** cung cấp các công cụ để làm việc với ngày và thời gian.

Ứng dụng trong chương trình:

```
import datetime
```

3. logging

Mục đích: **logging** hỗ trợ ghi nhật ký, ghi lại các thông báo, cảnh báo và lỗi xuất hiện trong quá trình chạy chương trình.

Ứng dụng trong chương trình:

```
import logging
```

4. os

Mục đích: Thư viện **os** cung cấp các hàm để tương tác với hệ thống hệ điều hành, bao gồm xử lý tệp và thư mục.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

Ứng dụng trong chương trình:

`import os`

5. pandas

Mục đích: **pandas** là một thư viện mạnh mẽ giúp ta xử lý và phân tích dữ liệu một cách dễ dàng và hiệu quả.

Ứng dụng trong chương trình:

`import pandas as pd`

6. requests

Mục đích: **requests** là một thư viện cho phép ta gửi các yêu cầu HTTP một cách dễ dàng.

Ứng dụng trong chương trình:

`import requests`

7. BeautifulSoup

Mục đích: **BeautifulSoup** hỗ trợ phân tích cú pháp HTML và XML, giúp trích xuất dữ liệu từ web một cách dễ dàng.

Ứng dụng trong chương trình:

`from bs4 import BeautifulSoup`

8. sklearn

Mục đích: **sklearn** cung cấp các công cụ mạnh mẽ cho việc xây dựng và đánh giá mô hình học máy.

Ứng dụng trong chương trình:

`from sklearn.metrics import accuracy_score, confusion_matrix`

`from sklearn.svm import SVC`

`from sklearn.tree import DecisionTreeClassifier`

`from sklearn.ensemble import VotingClassifier`

9. joblib

Mục đích: **joblib** giúp lưu và tải mô hình học máy một cách dễ dàng.

Ứng dụng trong chương trình:

`from joblib import dump, load`

10. Các thư viện và công cụ khác

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

Mục đích: Các thư viện như **time**, **tkinter**, **warnings**, và **ThreadPoolExecutor** từ **concurrent.futures** đều đóng vai trò quan trọng trong chương trình.

Ứng dụng trong chương trình:

```
import time  
import tkinter as tk  
import warnings  
from concurrent.futures import ThreadPoolExecutor
```

Giải thích:

- **time** được sử dụng để theo dõi và điều chỉnh thời gian thực thi của chương trình.
- **tkinter** giúp ta xây dựng giao diện người dùng đồ họa.
- **warnings** được sử dụng để cảnh báo các vấn đề có thể phát sinh mà không làm dừng chương trình.
- **ThreadPoolExecutor** từ **concurrent.futures** giúp ta thực hiện các tác vụ đa luồng, tăng hiệu suất xử lý của chương trình.

3.3 Cấu trúc chương trình

3.3.1 Hằng số và cấu hình

Trong chương trình, chúng em đã sử dụng một số hằng số để quản lý đường dẫn và cấu hình chung, cụ thể như sau:

- **TRAINING_DIR**: Đường dẫn đến thư mục chứa dữ liệu huấn luyện.
- **VALIDATION_DIR**: Đường dẫn đến thư mục chứa dữ liệu kiểm định.
- **PHISHING_DIR**: Đường dẫn đến thư mục chứa dữ liệu phishing.
- **NOT_PHISHING_DIR**: Đường dẫn đến thư mục chứa dữ liệu không phải phishing.
- **SVM_MODEL_PATH**: Đường dẫn đến file lưu mô hình SVM.
- **DT_MODEL_PATH**: Đường dẫn đến file lưu mô hình Decision Tree.
- **LOG_FILE**: Đường dẫn đến file log.

3.3.2 Hàm chính

- **read_html_from_file(filepath)**: Đọc nội dung HTML từ một tệp.
- **read_data()**: Đọc dữ liệu từ các thư mục và trả về dữ liệu đã được xử lý.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

- **extract_features_from_html(html)**: Trích xuất các đặc trưng từ nội dung HTML.
- **process_data()**: Xử lý dữ liệu và trích xuất đặc trưng cho việc huấn luyện.
- **save_model(model, filename)**: Lưu mô hình học máy vào một tệp.
- **load_model(filename)**: Tải mô hình học máy từ một tệp.
- **calculate_confusion_matrix(y_true, y_pred)**: Tính toán ma trận nhầm lẫn.
- **read_html_from_url(url)**: Lấy nội dung HTML từ một URL cụ thể.
- **process_url(url)**: Xử lý URL và trả về kết quả dự đoán về việc URL có phải là trang phishing hay không.
- **check_url()**: Kiểm tra URL từ clipboard và hiển thị kết quả dự đoán.
- **get_clipboard_url()**: Lấy URL từ clipboard của máy tính.

3.4 Phân tích dữ liệu HTML

3.4.1 Đọc nội dung HTML

Hàm **read_html_from_file(filepath)**:

```
def read_html_from_file(filepath):
    try:
        with open(filepath, 'r', encoding='utf-8') as file:
            html = file.read()
        return html
    except Exception as e:
        logging.warning(f'Không thể đọc file HTML {filepath}. Lỗi: {e}')
        return None
```

Đầu tiên, hàm sẽ cố gắng mở tệp HTML với đường dẫn được cung cấp thông qua biến **filepath** sử dụng mã hóa 'utf-8'.

Nếu việc mở tệp thành công, nội dung của tệp sẽ được đọc và trả về.

Trong trường hợp xảy ra lỗi (ví dụ: tệp không tồn tại), chúng ta sẽ ghi lại lỗi và trả về **None**.

3.4.2 Rút trích đặc trưng HTML

```
# Hàm rút trích các đặc trưng về bộ cục trang web từ tệp HTML
```

```
def extract_features_from_html(html):
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

try:

```
soup = BeautifulSoup(html, 'html.parser')
```

```
except Exception as e:
```

```
    logging.warning(f'Lỗi khi phân tích tệp HTML: {e}')
```

```
return None
```

```
tags_to_count = ['a', 'b', 'body', 'div', 'embed', 'form', 'head', 'html', 'i', 'iframe', 'img',  
'input',
```

```
        'link', 'meta', 'p', 'script', 'strong', 'style', 'title', 'u']
```

```
element_counts = {tag: 0 for tag in tags_to_count}
```

```
for tag in soup.find_all(tags_to_count):
```

```
    element_counts[tag.name] += 1
```

```
internal_links = sum(1 for a in soup.find_all('a') if 'href' in a.attrs and  
a['href'].startswith('/'))
```

```
external_links = sum(1 for a in soup.find_all('a') if 'href' in a.attrs and  
a['href'].startswith('http://'))
```

```
insecure_links = sum(1 for a in soup.find_all('a') if 'href' in a.attrs and  
a['href'].startswith('http://'))
```

```
external_images = sum(1 for img in soup.find_all('img') if 'src' in img.attrs and  
img['src'].startswith('http://'))
```

```
insecure_forms = sum(  
    1 for form in soup.find_all('form') if 'action' in form.attrs and not  
    form['action'].startswith('/'))
```

```
dom_complexity = len(soup.contents)
```

```
duplicated_content = len([tag for tag in soup.stripped_strings if  
list(soup.stripped_strings).count(tag) > 1])
```

```
return list(element_counts.values()) + [internal_links, external_links, insecure_links,  
external_images, insecure_forms, dom_complexity, duplicated_content]
```

Hàm extract_features_from_html để rút trích đặc trưng từ một đoạn mã HTML. Cụ thể:

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

- Sử dụng BeautifulSoup để phân tích HTML.
- Đếm số lượng các thẻ HTML quan trọng như 'a', 'div', 'img', v.v.
- Tính số liên kết nội bộ, liên kết bên ngoài, liên kết không an toàn và hình ảnh bên ngoài.
- Tính độ phức tạp của DOM.
- Tính số lượng nội dung trùng lặp.

Kết quả trả về là một danh sách chứa tất cả các đặc trưng đã được rút trích.

3.5 Thuật toán học máy

3.5.1 Huấn luyện và lưu mô hình học máy

```
# Tải mô hình SVM
if os.path.exists(SVM_MODEL_PATH):
    clf_svm = load_model(SVM_MODEL_PATH)
    svm_accuracy = accuracy_score(y_test, clf_svm.predict(X_test)) * 100
    logging.info('Đã tải xong mô hình SVM')

else:
    start_time = time.time()
    svm = SVC(kernel='linear', probability=True)
    clf_svm = svm.fit(X_train, y_train)
    svm_pred = clf_svm.predict(X_test)
    svm_accuracy = accuracy_score(y_test, svm_pred) * 100

# Tải mô hình Decision Tree
if os.path.exists(DT_MODEL_PATH):
    clf_dt = load_model(DT_MODEL_PATH)
    dt_accuracy = accuracy_score(y_test, clf_dt.predict(X_test)) * 100
    logging.info('Đã tải xong mô hình Decision Tree')

else:
    dt = DecisionTreeClassifier()
    clf_dt = dt.fit(X_train, y_train)
    dt_pred = clf_dt.predict(X_test)
    dt_accuracy = accuracy_score(y_test, dt_pred) * 100
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
# Lưu mô hình Decision Tree  
save_model(clf_dt, DT_MODEL_PATH)  
logging.info('Đã lưu xong mô hình Decision Tree')
```

Giải thích:

Đầu tiên, chương trình kiểm tra xem một mô hình SVM đã được lưu trước đó hay chưa.

Nếu mô hình đã tồn tại, chương trình sẽ tải mô hình đó thay vì huấn luyện từ đầu.

Nếu không, chương trình sẽ tạo ra một SVM với một "kernel" tuyến tính. "Kernel" là một phương thức giúp SVM phân loại dữ liệu. Trong trường hợp này, ta sử dụng "tuyến tính", nghĩa là ta đang tìm một đường thẳng (trong không gian 2D) hoặc một mặt phẳng (trong không gian 3D hoặc hơn) để phân loại dữ liệu.

svm.fit(X_train, y_train): Lệnh này sẽ bắt đầu quá trình huấn luyện, nơi SVM sẽ học từ dữ liệu huấn luyện **X_train** và nhãn tương ứng **y_train**.

Tương tự với Decision Trees, chương trình kiểm tra xem mô hình Decision Trees đã được lưu trước đó hay chưa. Nếu chưa có, một mô hình Decision Trees mới sẽ được tạo và huấn luyện.

3.5.2 Đánh giá hiệu suất của mô hình học máy

a. Độ chính xác (Accuracy):

```
svm_accuracy = accuracy_score(y_test, clf_svm.predict(X_test)) * 100  
svm_accuracy = accuracy_score(y_test, svm_pred) * 100  
dt_accuracy = accuracy_score(y_test, clf_dt.predict(X_test)) * 100  
dt_accuracy = accuracy_score(y_test, dt_pred) * 100
```

Giải thích:

Độ chính xác là một chỉ số đơn giản giúp ta biết mô hình dự đoán đúng bao nhiêu phần trăm trong tổng số dự đoán.

Để tính độ chính xác, ta so sánh kết quả dự đoán của mô hình (**clf_svm.predict(X_test)**) với những nhãn thực sự (**y_test**).

b. Ma trận nhầm lẫn (Confusion Matrix):

```
def calculate_confusion_matrix(model, X, y):  
    y_pred = model.predict(X)  
    return confusion_matrix(y, y_pred)
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
svm_confusion_matrix = calculate_confusion_matrix(clf_svm, X_test, y_test)
```

```
dt_confusion_matrix = calculate_confusion_matrix(clf_dt, X_test, y_test)
```

Giải thích:

Ma trận nhầm lẫn là một bảng được sử dụng để đánh giá hiệu suất của một mô hình phân loại. Nó hiển thị số lượng dự đoán chính xác và không chính xác được tạo ra bởi mô hình, cụ thể hơn, nó cung cấp thông tin về:

- True Positives (TP): Số lượng dự đoán chính xác về lớp positive.
- False Positives (FP): Số lượng dự đoán sai, mô hình dự đoán chúng là positive nhưng thực tế là negative.
- True Negatives (TN): Số lượng dự đoán chính xác về lớp negative.
- False Negatives (FN): Số lượng dự đoán sai, mô hình dự đoán chúng là negative nhưng thực tế là positive.

c. Voting Classifier:

```
voting_clf = VotingClassifier(  
    estimators=[('svm', clf_svm), ('dt', clf_dt)],  
    voting='soft'  
)  
voting_clf.fit(X_train, y_train)  
pred = voting_clf.predict([features])[0]
```

```
if pred == 0:  
    return "Tệp HTML từ URL cho thấy trang web KHÔNG CÓ dấu hiệu lừa đảo"
```

```
elif pred == 1:  
    return "Tệp HTML từ URL cho thấy trang web CÓ dấu hiệu lừa đảo"
```

Giải thích:

Chương trình tạo một "voting classifier" mới. Điều này giống như việc hỏi ý kiến từ hai người bạn: một người bạn được huấn luyện để nhận biết các trang web phishing thông qua mô hình SVM, và người kia thông qua mô hình Decision Tree.

Chương trình sau đó huấn luyện voting classifier này bằng cách sử dụng một tập dữ liệu đã biết, bao gồm các trang web đã được gắn nhãn là phishing hoặc không phải phishing.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

Khi có một trang web mới cần kiểm tra, chương trình sẽ lấy HTML của trang web đó, rút trích các đặc trưng của nó, và sau đó hỏi voting classifier để dự đoán xem trang web có phải là phishing hay không.

Voting classifier sẽ hỏi cả hai "người bạn" của mình (tức là mô hình SVM và Decision Tree) để xem họ nghĩ gì. Mỗi "người bạn" sẽ đưa ra dự đoán của mình.

Trong trường hợp của "soft voting", không chỉ là việc đếm phiếu bầu, mà còn xem xét mức độ "tự tin" của mỗi mô hình trong dự đoán của mình. Nếu một mô hình rất tự tin rằng một trang web là phishing, nó có thể truyền sự tự tin đó vào kết quả tổng thể.

3.6 Triển khai và ứng dụng

Đây là toàn bộ mã nguồn chương trình Phát hiện Web Phishing thông qua phân tích Bố cục trang web của nhóm chúng em, được lưu dưới tệp WebPhishing.py:

```
# Chương trình phát hiện Web Phishing thông qua phân tích Bố cục trang web dùng  
hai thuật toán học máy SVM và DecisionTree.
```

```
import csv  
import datetime  
import logging  
import os  
import time  
import tkinter as tk  
import warnings  
from concurrent.futures import ThreadPoolExecutor  
import pandas as pd  
import requests  
from bs4 import BeautifulSoup  
from joblib import dump, load  
from sklearn.metrics import accuracy_score, confusion_matrix  
from sklearn.svm import SVC  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.ensemble import VotingClassifier
```

```
# Các hằng số
```

```
TRAINING_DIR = 'training'
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
VALIDATION_DIR = 'validation'  
PHISHING_DIR = 'Phishing'  
NOT_PHISHING_DIR = 'NotPhishing'  
SVM_MODEL_PATH = 'svm_model.pkl'  
DT_MODEL_PATH = 'dt_model.pkl'  
LOG_FILE = 'wp.log'
```

```
# Thiết lập logging  
logging.basicConfig(  
    format='%(asctime)s - %(levelname)s - %(message)s',  
    level=logging.INFO,  
    filename=LOG_FILE  
)
```

```
console = logging.StreamHandler()  
console.setLevel(logging.INFO)  
logging.getLogger("").addHandler(console)  
warnings.filterwarnings("ignore", category=UserWarning, module="sklearn")
```

```
# Hàm đọc HTML từ file  
def read_html_from_file(filepath):  
    try:  
        with open(filepath, 'r', encoding='utf-8') as f:  
            content = f.read()  
            logging.info(f"Đọc xong file HTML {filepath}")  
            return content  
    except Exception as e:  
        logging.warning(f"Không thể đọc file HTML {filepath}. Lỗi: {e}")  
        return None
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

Hàm đọc dữ liệu từ các thư mục training và validation

```
def read_data(directory):
```

```
    categories = ['Phishing', 'NotPhishing']
```

```
    data = []
```

```
    for category in categories:
```

```
        category_dir = os.path.join(directory, category)
```

```
        for filename in os.listdir(category_dir):
```

```
            filepath = os.path.join(category_dir, filename)
```

```
            data.append({'path': filepath, 'category': category})
```

```
    return pd.DataFrame(data)
```

Đọc dữ liệu từ thư mục training và validation

```
training_data = read_data(TRAINING_DIR)
```

```
validation_data = read_data(VALIDATION_DIR)
```

```
logging.info('Đã đọc xong dữ liệu training và validation')
```

Đọc HTML từ các file

```
with ThreadPoolExecutor(max_workers=10) as executor:
```

```
    training_data['html'] = list(executor.map(read_html_from_file,  
    training_data['path']))
```

```
    validation_data['html'] = list(executor.map(read_html_from_file,  
    validation_data['path']))
```

Hàm rút trích các đặc trưng về bộ lọc trang web từ tệp HTML

```
def extract_features_from_html(html):
```

```
    try:
```

```
        soup = BeautifulSoup(html, 'html.parser')
```

```
    except Exception as e:
```

```
        logging.warning(f'Lỗi khi phân tích tệp HTML: {e}')
```

```
    return None
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
tags_to_count = ['a', 'b', 'body', 'div', 'embed', 'form', 'head', 'html', 'i', 'iframe', 'img',
'input',
    'link', 'meta', 'p', 'script', 'strong', 'style', 'title', 'u']

element_counts = {tag: 0 for tag in tags_to_count}

for tag in soup.find_all(tags_to_count):
    element_counts[tag.name] += 1

internal_links = sum(1 for a in soup.find_all('a') if 'href' in a.attrs and
a['href'].startswith('/'))

external_links = sum(1 for a in soup.find_all('a') if 'href' in a.attrs and
a['href'].startswith('http://'))

insecure_links = sum(1 for a in soup.find_all('a') if 'href' in a.attrs and
a['href'].startswith('http://'))

external_images = sum(1 for img in soup.find_all('img') if 'src' in img.attrs and
img['src'].startswith('http://'))

insecure_forms = sum(
    1 for form in soup.find_all('form') if 'action' in form.attrs and not
form['action'].startswith('/'))

dom_complexity = len(soup.contents)

duplicated_content = len([tag for tag in soup.stripped_strings if
list(soup.stripped_strings).count(tag) > 1])

return list(element_counts.values()) + [internal_links, external_links, insecure_links,
external_images, insecure_forms, dom_complexity, duplicated_content]
```

Hàm xử lý dữ liệu

```
def process_data(df):
    df['html'] = df['path'].apply(read_html_from_file)
    df['features'] = df['html'].apply(extract_features_from_html)
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
df = df.dropna(subset=['features'])

features = pd.DataFrame(df['features'].to_list())
features.columns = features.columns.astype(str)

df = pd.concat([df, features], axis=1)

df = df.drop(columns=['html', 'features', 'path'])

return df
```

```
# Xử lý dữ liệu training và validation

training_data = process_data(training_data)
training_data = training_data.dropna()
validation_data = process_data(validation_data)
validation_data = validation_data.dropna()

logging.info('Đã xử lý xong dữ liệu training và validation')
```

```
# Hàm để lưu mô hình ML vào tệp

def save_model(model, filename):
    try:
        dump(model, filename)
    except Exception as e:
        logging.warning(f"Không thể lưu mô hình. Lỗi: {e}")
```

```
# Hàm để tải mô hình ML từ tệp

def load_model(filename):
    if not os.path.exists(filename):
        logging.warning("File mô hình không tồn tại: " + filename)
        return None
    try:
        return load(filename)
    except Exception as e:
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
logging.warning(f'Không thể tải mô hình từ file {filename}. Lỗi: {e}')
```

```
# Tạo DataFrame từ danh sách các file HTML phishing và không phishing trong thư mục training
```

```
training_phishing_data = training_data[training_data['category'] == 'Phishing'].copy()  
training_not_phishing_data = training_data[training_data['category'] ==  
'NotPhishing'].copy()
```

```
# Gán nhãn cho dữ liệu training
```

```
training_phishing_data['label'] = 1  
training_not_phishing_data['label'] = 0
```

```
# Kết hợp dữ liệu training
```

```
training_data = pd.concat([training_phishing_data, training_not_phishing_data])
```

```
# Tạo DataFrame từ danh sách các file HTML phishing và không phishing trong thư mục validation
```

```
validation_phishing_data = validation_data[validation_data['category'] ==  
'Phishing'].copy()  
validation_not_phishing_data = validation_data[validation_data['category'] ==  
'NotPhishing'].copy()
```

```
# Gán nhãn cho dữ liệu validation
```

```
validation_phishing_data['label'] = 1  
validation_not_phishing_data['label'] = 0
```

```
# Kết hợp dữ liệu validation
```

```
validation_data = pd.concat([validation_phishing_data,  
validation_not_phishing_data])
```

```
# Chia dữ liệu thành tập huấn luyện và tập kiểm tra
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
X_train = training_data.drop(columns=['label', 'category'])
y_train = training_data['label']
X_test = validation_data.drop(columns=['label', 'category'])
y_test = validation_data['label']

# Tải mô hình SVM
if os.path.exists(SVM_MODEL_PATH):
    clf_svm = load_model(SVM_MODEL_PATH)
    svm_accuracy = accuracy_score(y_test, clf_svm.predict(X_test)) * 100
    logging.info('Đã tải xong mô hình SVM')

else:
    start_time = time.time()
    svm = SVC(kernel='linear', probability=True)
    clf_svm = svm.fit(X_train, y_train)
    svm_pred = clf_svm.predict(X_test)
    svm_accuracy = accuracy_score(y_test, svm_pred) * 100

# Lưu mô hình SVM
save_model(clf_svm, SVM_MODEL_PATH)
logging.info('Đã lưu xong mô hình SVM')

# Tải mô hình Decision Tree
if os.path.exists(DT_MODEL_PATH):
    clf_dt = load_model(DT_MODEL_PATH)
    dt_accuracy = accuracy_score(y_test, clf_dt.predict(X_test)) * 100
    logging.info('Đã tải xong mô hình Decision Tree')

else:
    dt = DecisionTreeClassifier()
    clf_dt = dt.fit(X_train, y_train)
    dt_pred = clf_dt.predict(X_test)
    dt_accuracy = accuracy_score(y_test, dt_pred) * 100
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
# Lưu mô hình Decision Tree
save_model(clf_dt, DT_MODEL_PATH)
logging.info('Đã lưu xong mô hình Decision Tree')

# Hiển thị độ chính xác của hai mô hình
logging.info(f'Dộ chính xác của mô hình SVM: {svm_accuracy}%')
logging.info(f'Dộ chính xác của mô hình Decision Tree: {dt_accuracy}%')

# Hàm để tính ma trận nhầm lẫn
def calculate_confusion_matrix(model, X, y):
    y_pred = model.predict(X)
    return confusion_matrix(y, y_pred)

# Tính ma trận nhầm lẫn cho hai thuật toán
svm_confusion_matrix = calculate_confusion_matrix(clf_svm, X_test, y_test)
dt_confusion_matrix = calculate_confusion_matrix(clf_dt, X_test, y_test)

# Hiển thị ma trận nhầm lẫn cho hai thuật toán
logging.info("Ma trận nhầm lẫn cho SVM:")
logging.info(svm_confusion_matrix)
logging.info("\nMa trận nhầm lẫn cho Decision Tree:")
logging.info(dt_confusion_matrix)

# Tạo một voting classifier
voting_clf = VotingClassifier(
    estimators=[('svm', clf_svm), ('dt', clf_dt)],
    voting='soft'
)

# Huấn luyện voting classifier trên tập huấn luyện
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
voting_clf.fit(X_train, y_train)
```

```
# Hàm đọc HTML từ URL
```

```
def read_html_from_url(url):
```

```
    try:
```

```
        response = requests.get(url)
```

```
        if response.status_code == 200:
```

```
            content = response.text
```

```
            logging.info(f"Đọc xong tệp HTML từ URL: {url}")
```

```
            return content
```

```
        else:
```

```
            logging.warning(f'Lỗi khi lấy tệp HTML từ URL: {url}. Mã trạng thái:
```

```
{response.status_code}")
```

```
    return None
```

```
except Exception as e:
```

```
    logging.warning(f'Lỗi khi lấy tệp HTML từ URL: {url}. Lỗi: {e}')
```

```
    return None
```

```
# Hàm lấy tệp HTML từ URL và thực hiện phân tích
```

```
def process_url(url):
```

```
    html_content = read_html_from_url(url)
```

```
    if html_content is None:
```

```
        return "Không thể lấy tệp HTML từ URL nên không thể phân tích"
```

```
    features = extract_features_from_html(html_content)
```

```
    if features is None:
```

```
        return "Không thể rút trích đặc trưng từ tệp HTML"
```

```
    pred = voting_clf.predict([features])[0]
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
if pred == 0:  
    return "Tệp HTML từ URL cho thấy trang web KHÔNG CÓ dấu hiệu lừa đảo"  
elif pred == 1:  
    return "Tệp HTML từ URL cho thấy trang web CÓ dấu hiệu lừa đảo"  
  
# Hàm xử lý sự kiện khi muốn kiểm tra tệp HTML từ URL  
def check_url():  
    url = url_entry.get()  
    if url:  
        result = process_url(url)  
        if result is None:  
            logging.warning('Không thể lấy kết quả')  
        else:  
            if "KHÔNG CÓ" in result:  
                result_label.configure(text=result, fg="green")  
            else:  
                result_label.configure(text=result, fg="red")  
  
            with open('results.csv', 'a', newline='', encoding='utf-8') as f:  
                writer = csv.writer(f)  
                writer.writerow([url, result, datetime.datetime.now()])  
  
# Lấy URL từ clipboard  
def get_clipboard_url():  
    try:  
        import tkinter as tk  
        root = tk.Tk()  
        root.withdraw()  
        url = root.clipboard_get()  
        url_entry.delete(0, tk.END)
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
url_entry.insert(0, url)
except Exception as e:
    logging.warning(f'Lỗi khi lấy URL từ clipboard: {e}')

app = tk.Tk()
frame = tk.Frame(app)
clipboard_button = tk.Button(frame, text="Lấy URL", command=get_clipboard_url)
clipboard_button.configure(width=10, height=1)
clipboard_button.grid(row=3, column=1, padx=(0, 10), pady=(0, 10), sticky=tk.E)

submit_button = tk.Button(frame, text="Kiểm tra", command=check_url)
submit_button.configure(width=10, height=1)
submit_button.grid(row=4, column=0, columnspan=2)

result_label = tk.Label(frame, text="")
result_label.grid(row=5, column=0, columnspan=2, pady=(10, 0))

# Giao diện người dùng
app.title("Kiểm tra website thông qua phân tích Bộ cục trang web")
app.geometry("600x250")

frame.place(relx=0.5, rely=0.5, anchor=tk.CENTER)

title_label = tk.Label(frame, text="ỨNG DỤNG KIỂM TRA WEBSITE",
                      font=("Times New Roman", 16, "bold"))
title_label.grid(row=0, column=0, columnspan=2, pady=(0, 10))

title_label = tk.Label(frame, text="TÁC GIẢ: TRẦN THIỆN NHÂN", font=("Times New Roman", 14, "bold"))
title_label.grid(row=1, column=0, columnspan=2, pady=(0, 10))
```

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

```
url_label = tk.Label(frame, text="Nhập URL:")
url_label.grid(row=2, column=0, padx=(10, 5), pady=(10, 0), sticky=tk.W)

url_entry = tk.Entry(frame, width=50)
url_entry.grid(row=3, column=0, padx=(5, 10), pady=(0, 10), sticky=tk.N + tk.S)

app.mainloop()
```

Mô tả ngắn gọn các bước quan trọng:

- Đọc và xử lý dữ liệu: Chương trình đọc HTML từ file và rút trích các đặc trưng như số lượng các thẻ HTML khác nhau, số lượng liên kết nội bộ và bên ngoài, v.v.
- Huấn luyện mô hình máy học: Chương trình huấn luyện mô hình Support Vector Machines và Decision Trees trên dữ liệu đã được xử lý và đánh giá chúng trên tập kiểm tra. Nếu mô hình đã được huấn luyện trước đó, chúng sẽ được tải từ file.
- Tính toán ma trận nhầm lẫn: Dùng để đánh giá hiệu suất của mô hình bằng cách xem xét các trường hợp true positive, true negative, false positive, và false negative.
- Tạo Voting Classifier: Mô hình này kết hợp kết quả từ cả hai mô hình Support Vector Machines và Decision Trees để cải thiện hiệu suất.
- Kiểm tra URL: Người dùng nhập URL vào giao diện, chương trình lấy HTML từ URL đó, rút trích đặc trưng và sử dụng mô hình đã huấn luyện để dự đoán xem trang web có phải là phishing hay không.
- Hiển thị kết quả trên GUI: Kết quả dự đoán được hiển thị trên giao diện để người dùng có thể thấy.

Tiếp theo, chúng em cho chương trình chạy. File wp.log hiển thị logging của chương trình. Sau khi chạy xong chương trình sẽ hiển thị thông tin về độ chính xác và ma trận nhầm lẫn.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

The screenshot shows a Windows Command Prompt window titled "wp.log". The log file contains numerous INFO-level messages from the "DuyetDC" module, detailing the validation of various URLs. The log entries are timestamped and show the file path being checked, such as "file HTML validation_test\NotPhishing\1_Service-public_tr.html", "file HTML validation_test\NotPhishing\1_stackoverflow_com.html", etc. The log continues through several pages of similar entries, ending with a final message about training and validation.

```
2023-10-01 22:17:18,621 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\1_Service-public_tr.html
2023-10-01 22:17:18,621 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\1_ssa_gov.html
2023-10-01 22:17:18,621 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\1_stackoverflow_com.html
2023-10-01 22:17:18,621 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\1_stockwits_com.html
2023-10-01 22:17:18,628 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\1_tdameritrade_com.html
2023-10-01 22:17:18,628 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\1_tiktok_com.html
2023-10-01 22:17:18,628 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\1_vanguard_com.html
2023-10-01 22:17:18,628 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\1_whitebit_com.html
2023-10-01 22:17:18,628 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\1_zerodha_com.html
2023-10-01 22:17:18,628 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\1_zoosk_com.html
2023-10-01 22:17:18,628 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_aftenric_com.html
2023-10-01 22:17:18,628 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_archive_org.html
2023-10-01 22:17:18,628 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_bbco_uk.html
2023-10-01 22:17:18,628 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_bit_ly.html
2023-10-01 22:17:18,635 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_dropbox_com.html
2023-10-01 22:17:18,635 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_elpais_com.html
2023-10-01 22:17:18,635 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_enable-javascript_com.html
2023-10-01 22:17:18,635 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_google_co_jp.html
2023-10-01 22:17:18,635 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_google_de.html
2023-10-01 22:17:18,635 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_google_ru.html
2023-10-01 22:17:18,635 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_nih_gov.html
2023-10-01 22:17:18,635 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_ok_ru.html
2023-10-01 22:17:18,635 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_opera_com.html
2023-10-01 22:17:18,635 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_researchgate_net.html
2023-10-01 22:17:18,642 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_theguardian_com.html
2023-10-01 22:17:18,642 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\2_vk_com.html
2023-10-01 22:17:18,642 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_abc_es.html
2023-10-01 22:17:18,642 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_academia_edu.html
2023-10-01 22:17:18,642 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_addtoany_com.html
2023-10-01 22:17:18,648 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_aftenric_com.html
2023-10-01 22:17:18,649 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_amazon_es.html
2023-10-01 22:17:18,649 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_apache_org.html
2023-10-01 22:17:18,656 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_bild_de.html
2023-10-01 22:17:18,656 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_bitly_com.html
2023-10-01 22:17:18,656 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_books_google_com.html
2023-10-01 22:17:18,656 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_box_com.html
2023-10-01 22:17:18,656 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_canada_ca.html
2023-10-01 22:17:18,656 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_cbsnews_com.html
2023-10-01 22:17:18,663 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_cisco_com.html
2023-10-01 22:17:18,663 - INFO - DuyetDC xong file HTML validation_test\NotPhishing\3_cornell_edu.html
2023-10-01 22:17:22,280 - INFO - DuyetDC xong du lieu de training va validation
```

Hình 12 File logging wp.log ghi lại quá trình hoạt động của chương trình

Má trận nhầm lẫn cung cấp một cái nhìn chi tiết về hiệu suất của mô hình, giúp đánh giá mô hình từ nhiều khía cạnh hơn chỉ là độ chính xác.

The screenshot shows a Jupyter Notebook interface with a Python script named "WebPhishing.py" running. The notebook displays the following output:

```
Bịc xong file HTML validation_test\NotPhishing\3_cbsnews_com.html
Bịc xong file HTML validation_test\NotPhishing\3_cisco_com.html
Bịc xong file HTML validation_test\NotPhishing\3_cnll_fr.html
Bịc xong file HTML validation_test\NotPhishing\3_cornell_edu.html
Bà xử lý xong dữ liệu training và validation
Bà lưu xong mô hình SVM
Bà lưu xong mô hình Decision Tree
Độ chính xác của mô hình SVM: 66.66666666666666%
Độ chính xác của mô hình Decision Tree: 83.33333333333334%
Má trận nhầm lẫn cho SVM:
[[98 16]
 [60 54]]
Má trận nhầm lẫn cho Decision Tree:
[[97 17]
 [21 93]]
```

Hình 13 Thông tin về độ chính xác và ma trận nhầm lẫn

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

Thử nghiệm với trang web không phishing, chúng em lấy trang web của Báo Điện tử Chính phủ: <https://baochinhphu.vn/bo-chinh-tri.html>

The screenshot shows the homepage of the website 'Báo Điện tử Chính phủ'. At the top, there is a banner with the text 'CHÍNH PHỦ NƯỚC CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM' and the logo of the National Emblem of Vietnam. Below the banner, the main title 'Báo Điện tử Chính phủ' is displayed in large red letters. To the right of the title, it says 'Chủ nhật, 1/10/2023 | English | 中文'. The navigation bar includes links for 'TRANG CHỦ', 'CHÍNH TRỊ', 'KINH TẾ', 'VĂN HÓA', 'XÃ HỘI', 'KHOA GIÁO', 'QUỐC TẾ', 'GÓP Ý HIẾN KẾ', 'MỚI NHẤT', and a search icon. Below the navigation bar, there is a search bar with placeholder text 'Xây dựng nền kinh tế độc lập tự chủ gắn với hội nhập qu...' and a date range selector '25° - 36°C' with a weather icon.

I BỘ CHÍNH TRỊ

Đẩy nhanh xây dựng đường sắt đô thị: Cần từ bỏ vốn ODA

Kinh tế - 2 tháng trước
(Chinhphu.vn) - Các chuyên gia đều cho rằng, đổi với nguồn lực tài chính để xây dựng hệ thống đường sắt đô thị tại TPHCM, cần đề xuất cơ chế đột phá mới thực hiện được mục tiêu hoàn thành hệ thống 200 km đường sắt đô thị vào năm 2035 theo Kết luận 49 của Bộ...

Thước đo đổi với sự lãnh đạo của Đảng là cuộc sống tốt hơn của người dân

Chính trị - 3 tháng trước
(Chinhphu.vn) - "Có lẽ, đổi với sự lãnh đạo của Đảng, thước đo quan trọng nhất là cuộc sống tốt hơn của người dân. Sự đồng thuận, niềm tin của người dân đối với Đảng là mục tiêu quan trọng của Đảng và của Đảng bộ TPHCM".

TPHCM cần gì ở nghị quyết mới thay thế Nghị quyết 54?

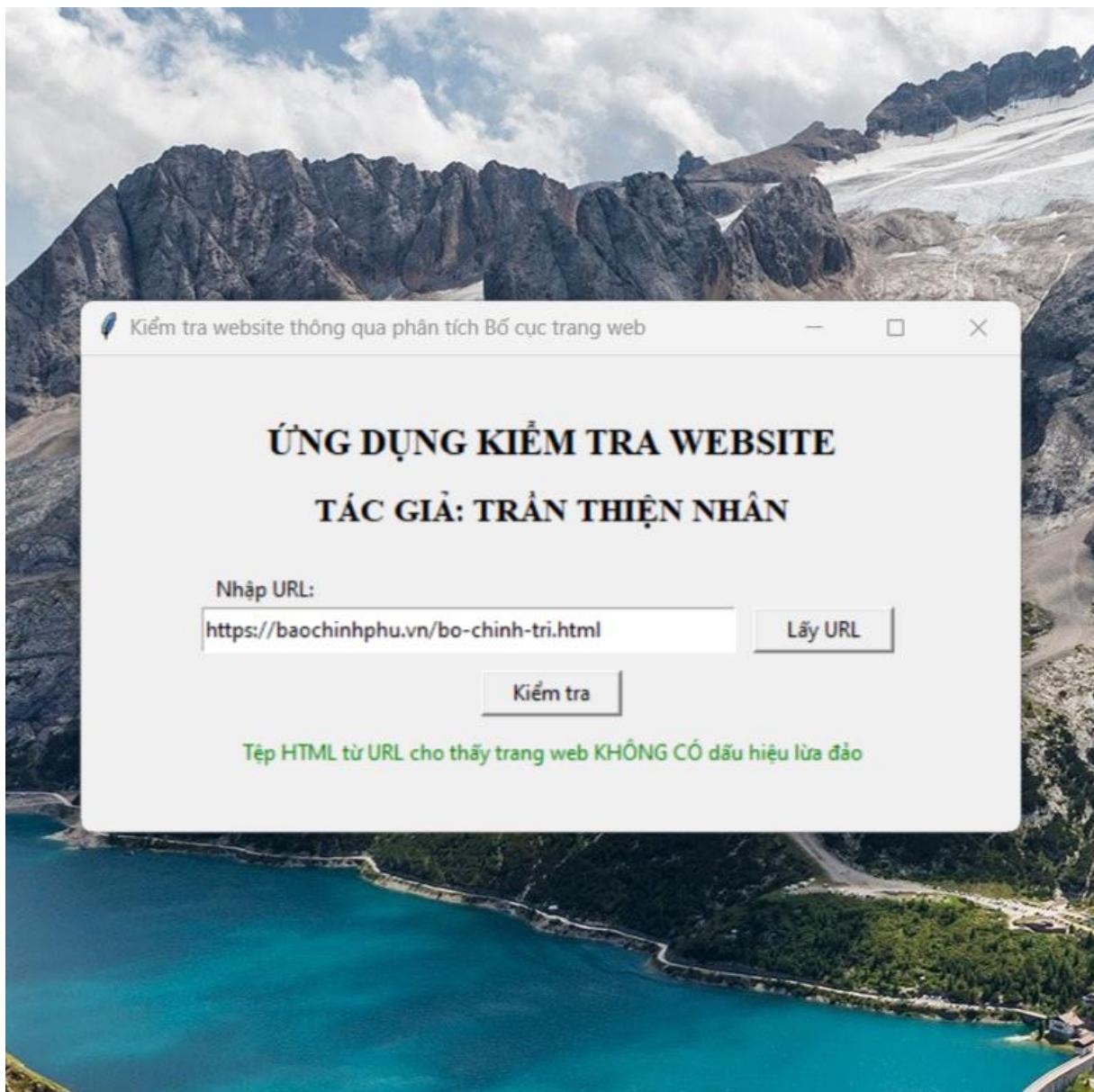
I ĐỌC NHIỀU

- Vietjet mở đường bay đến 5 thành phố lớn của Australia
- Thông cáo báo chí phiên họp Chính phủ thường kỳ tháng 9
- Phát triển nhà ở xã hội: Nhiều chính sách ưu đãi cho doanh nghiệp đầu tư
- Cán thanh tra toàn diện nhà xe Thành Bưởi

Hình 14 Trang Báo Điện tử Chính phủ

Và đây là kết quả của chương trình:

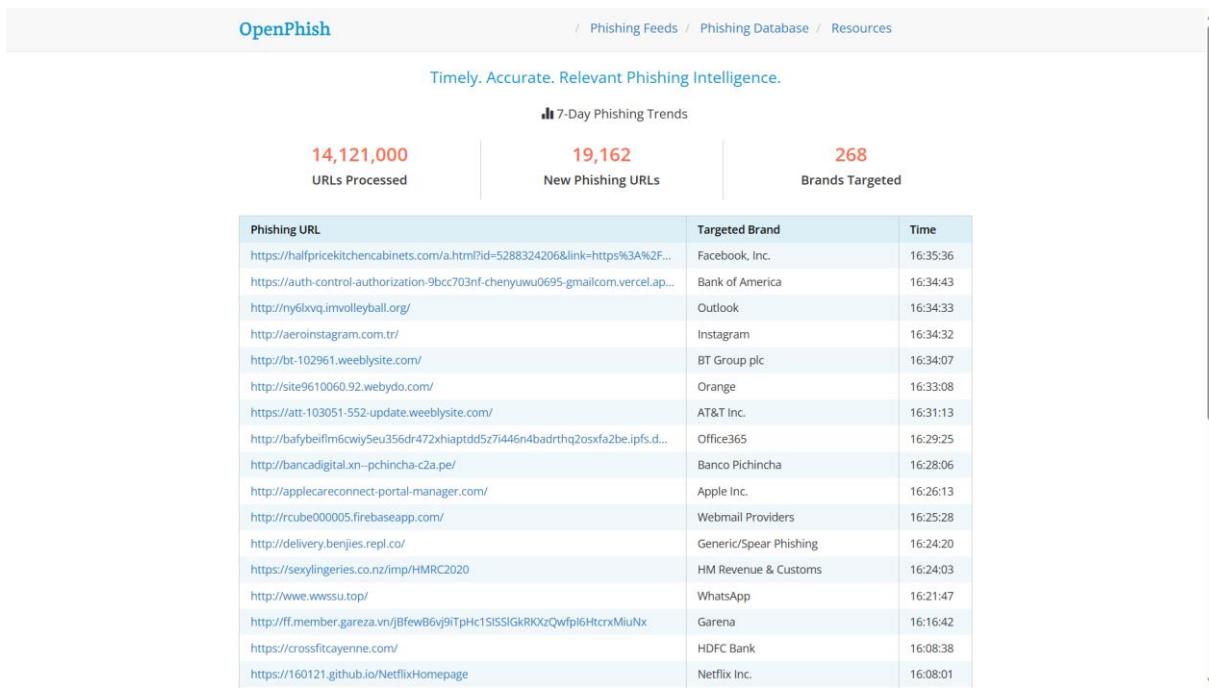
NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26



Hình 15 Chương trình hoạt động với trang web không phishing

Thử nghiệm với trang web phishing, chúng em lấy trang web bất kỳ từ trang <https://openphish.com/>, một trang web cung cấp thông tin về các URL độc hại được sử dụng trong các chiến dịch lừa đảo phishing.

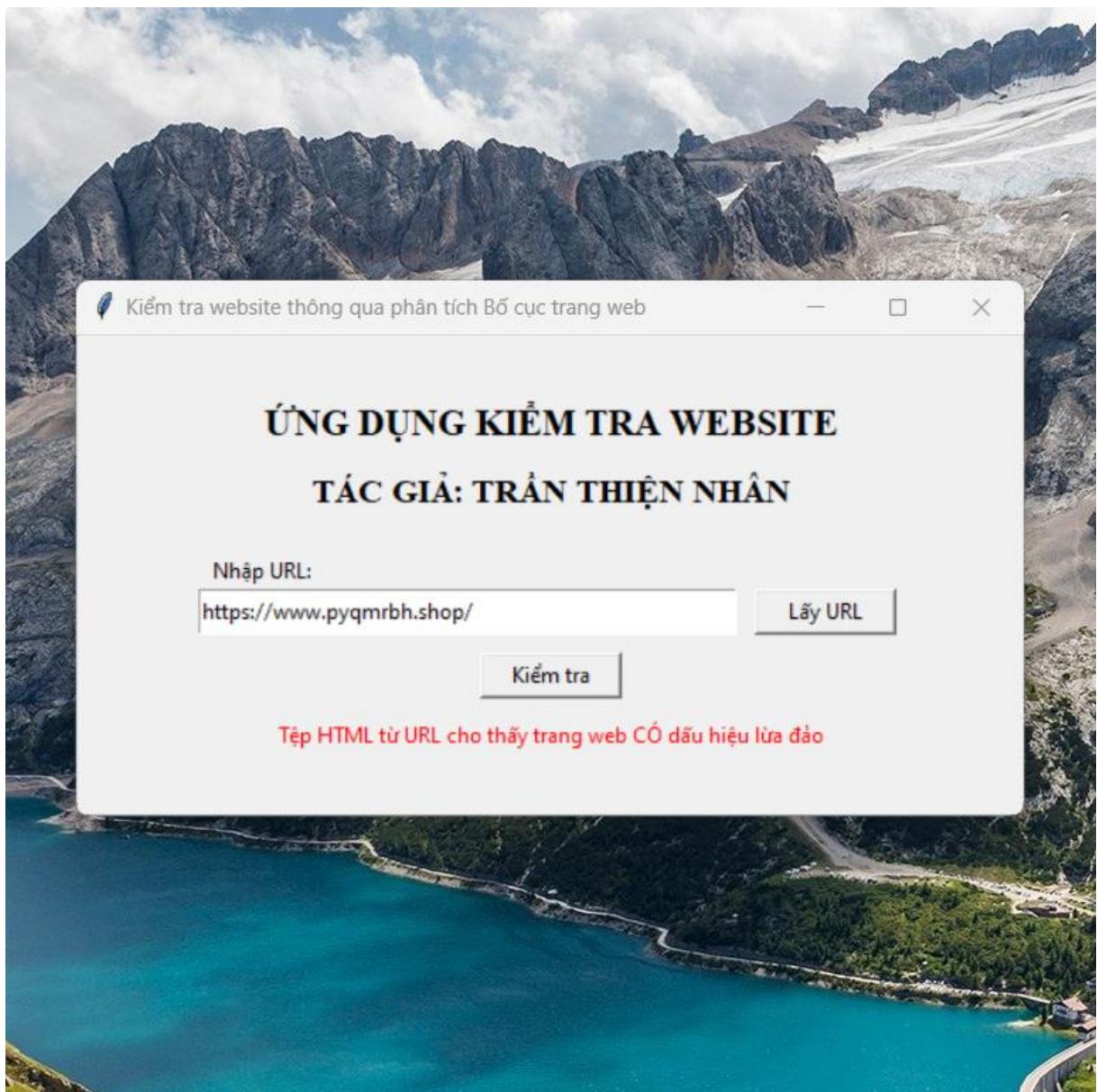
NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26



Hinh 16 Trang OpenPhish

Và đây là kết quả của chương trình:

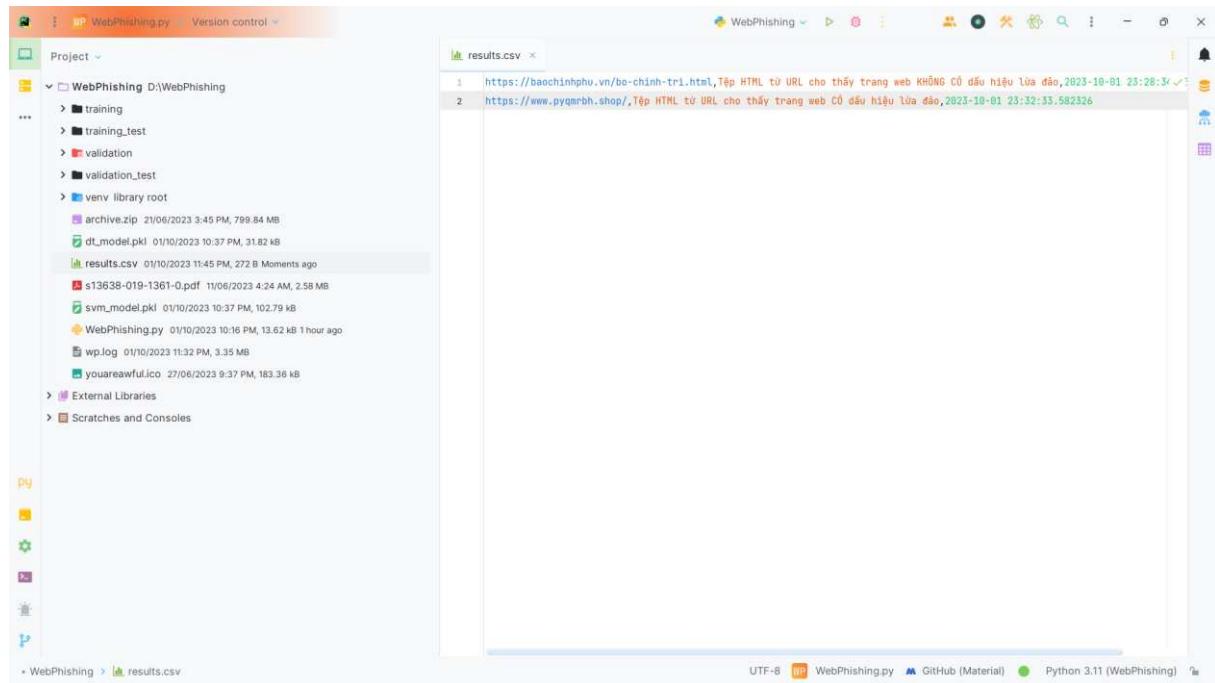
NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26



Hình 17 Chương trình hoạt động với trang web phishing

File results.csv ghi lại lịch sử kiểm tra trang web gồm URL, kết quả và thời điểm kiểm tra.

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26



Hình 18 Lịch sử kiểm tra từ file results.csv

KẾT LUẬN

1. Hiệu quả

Đây là một phương pháp mang tính mới lạ và độc đáo có hiệu quả trong việc đối phó với các trang web phishing hiện nay.

Chương trình đã ứng dụng phương pháp phân tích sâu rộng vào bộ cục trang web cùng với sự kết hợp của hai thuật toán học máy Support Vector Machines và Decision Trees giúp nâng cao khả năng phát hiện những trang web phishing.

2. Hạn chế

Bộ dữ liệu hiện tại chỉ ở mức trung bình vẫn cần được mở rộng và cải thiện. Độ lớn và đa dạng của bộ dữ liệu có thể là chìa khóa để tăng cường khả năng phát hiện trước các trang web phishing sáng tạo và tinh vi hơn.

Như mọi mô hình học máy khác, mô hình này cũng không phải lúc nào cũng chính xác. Đôi khi, có thể xuất hiện các kết quả lỗi, điều này yêu cầu phải có một chương trình được huấn luyện và điều chỉnh liên tục để tối ưu hiệu suất. Ngoài ra, để tăng độ chính xác, việc kết hợp với các kỹ thuật và phương pháp khác nhau trong việc phân loại và phát hiện trang web phishing là điều cần thiết nếu chương trình có sự thương mại.

3. Hướng phát triển trong tương lai

Để chương trình tiếp tục phát triển, việc mở rộng bộ dữ liệu là ưu tiên hàng đầu. Không chỉ giúp nâng cao độ chính xác, việc này còn giúp ta trang bị cho mô hình học máy sức mạnh cần thiết để đối diện với các biến thể và tình huống mới mà các trang web phishing có thể mang lại.

Tập trung vào việc nghiên cứu và phát triển các phương pháp kết hợp, tối ưu hóa mô hình hiện tại và tích hợp thêm nhiều kỹ thuật và phương pháp phân tích khác vào chương trình để nâng cao khả năng phát hiện và đối phó với các kỹ thuật phishing ngày càng phức tạp.

TÀI LIỆU THAM KHẢO

- [1] Le Dang Nguyen, Le Dac Nhuong, Le Trong Vinh, "Detecting phishing web pages based on DOM-tree structure and graph matching algorithm," *SoICT'14*, p. 280–285, 04 December 2014.
- [2] Tuan Anh Pham, Xuan Hoai Nguyen, Quang Uy Nguyen, "Phishing Attacks Detection Using Genetic Programming," in *Knowledge and Systems Engineering*, 2014, p. 185–195.
- [3] M Selvakumari, M Sowjanya, Sneha Das, S Padmavathi, "Phishing website detection using machine learning and deep learning techniques," *Journal of Physics: Conference Series*, pp. 25-26, 2021.
- [4] X. Wang, B. Wu, C. Wu, R. Yang, K. Zheng, "Phishing Website Detection Based on Deep Convolutional Neural Network and Random Forest Ensemble Learning," *Sensors (Basel, Switzerland)*, vol. 21, no. 24, p. 8281, 2021.
- [5] P. Zhang, L. Zhang, "PhishTrim: Fast and adaptive phishing detection based on deep representation learning," in *2020 IEEE International Conference on Web Services (ICWS)*, Beijing, China, 2020.
- [6] HandWiki, "HandWiki," March 2022. [Online]. Available: https://handwiki.org/wiki/Support_vector_machine. [Accessed 20 August 2023].
- [7] Sidharth, "PyCodeMates," 30 November 2022. [Online]. Available: <https://www.pycodemates.com/2022/07/support-vector-machines-detailed-overview.html>. [Accessed 21 August 2023].
- [8] IBM, "IBM," 17 August 2021. [Online]. Available: <https://www.ibm.com/docs/en/spss-modeler/saas?topic=models-how-svm-works>. [Accessed 21 August 2023].
- [9] Dataiku, "History of Data Science," Dataiku, 31 August 2021. [Online]. Available: <https://www.historyofdatascience.com/decision-tree-and-random-forest-algorithms-decision-drivers/>. [Accessed 20 August 2023].

NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN NHÓM 26

- [10] A. Awan, "datacamp," February 2023. [Online]. Available: <https://www.datacamp.com/tutorial/decision-tree-classification-python>. [Accessed 21 August 2023].
- [11] Kumar, Vivek, "TowardsMachineLearning.org," 2023. [Online]. Available: <https://towardsmachinelearning.org/decision-tree-algorithm/>. [Accessed 21 August 2023].
- [12] Brownlee, Jason, "Machine Learning Mastery," Guiding Tech Media, 13 July 2020. [Online]. Available: <https://machinelearningmastery.com/what-is-information-entropy/>. [Accessed 22 August 2023].
- [13] Hunter, Kempf, "kaggle," 28 February 2022. [Online]. Available: <https://www.kaggle.com/datasets/huntingdata11/phishing-website-html-classification>. [Accessed 22 August 2023].

TP. HCM, ngày 16 tháng 04 năm 2023

XÁC NHẬN CỦA GIẢNG VIÊN HƯỚNG DẪN

Th.S NGUYỄN HOÀNG THÀNH