

*Tp.HCM, ngày 12 tháng 01 năm 2025*

Sinh viên thực hiện:

CHAT3P05 - Nguyễn Phạm Lâm Duy

CHAT3P11 - Huỳnh Hoàng Lam

**BAN CƠ YẾU CHÍNH PHỦ**

HỌC VIỆN KỸ THUẬT MẬT MÃ

**Đề tài:**

**PHÂN LOẠI MÃ ĐỘC BẰNG HỌC MÁY**

Giảng viên hướng dẫn: Thầy Nguyễn An Khương

*Mục Lục*

[1. Giới thiệu đề tài 1](#_Toc187351745)

[1.1. Đặt vấn đề 1](#_Toc187351746)

[1.2. Mục tiêu và phạm vi đề tài 1](#_Toc187351747)

[1.2.1. Mục tiêu 1](#_Toc187351748)

[1.2.2. Phạm vi đề tài 2](#_Toc187351749)

[1.3. Ý nghĩa thực tiễn 3](#_Toc187351750)

[2. Định nghĩa mã độc và các loại mã độc phổ biến 4](#_Toc187351751)

[2.1. Mã độc là gì? 4](#_Toc187351752)

[2.1.1. Các loại mã độc phổ biến 4](#_Toc187351753)

[2.1.2. Phân tích các phương pháp phát hiện mã độc truyền thống 5](#_Toc187351754)

[2.1.3. Những thách thức trong phát hiện mã độc truyền thống 8](#_Toc187351755)

[2.1.4. Tổng quan về các phương pháp học máy trong bảo mật 10](#_Toc187351756)

[3. Giới thiệu các thuật toán học máy 11](#_Toc187351757)

[3.1. Random Forest 11](#_Toc187351758)

[3.2. K-Nearest Neighbors (KNN) 12](#_Toc187351759)

[3.3. Decision Tree 13](#_Toc187351760)

[4. Phân tích các kỹ thuật học máy được sử dụng trong phân loại mã độc 14](#_Toc187351761)

[4.1. Random Forest 14](#_Toc187351762)

[4.2. K-Nearest Neighbors (KNN) 15](#_Toc187351763)

[4.3. Decision Tree (Cây quyết định) 16](#_Toc187351764)

[4.4.1. Trích xuất đặc trưng mã độc từ dữ liệu tĩnh 17](#_Toc187351765)

[4.4.2. Trích xuất đặc trưng mã độc từ dữ liệu động 17](#_Toc187351766)

[4.5. Quy trình thu thập và xử lý dữ liệu mã độc 18](#_Toc187351767)

[Giai đoạn 1: Thu thập dữ liệu mã độc (Malware Collection) 18](#_Toc187351768)

[Giai đoạn 2: Tiền xử lý dữ liệu mã độc (Preprocessing) 19](#_Toc187351769)

[Giai đoạn 3: Xử lý dữ liệu mã độc (Data Processing) 19](#_Toc187351770)

[Giai đoạn 4: Phân tích và nhận diện mã độc (Malware Analysis) 19](#_Toc187351771)

[Giai đoạn 5: Lưu trữ và báo cáo (Storage & Reporting) 20](#_Toc187351772)

[5. Phân tích thuật toán học máy để xây dựng mô hình phân loại 20](#_Toc187351773)

[5.1. Quá trình huấn luyện, tối ưu hóa và đánh giá mô hình. 20](#_Toc187351774)

[5.1.1. Quá trình huấn luyện mô hình (Model Training Process) 20](#_Toc187351775)

[Bước 1: Chuẩn bị dữ liệu huấn luyện và kiểm tra 20](#_Toc187351776)

[Bước 2: Lựa chọn thuật toán và khởi tạo mô hình 20](#_Toc187351777)

[Bước 3: Huấn luyện mô hình (Training) 21](#_Toc187351778)

[5.1.2. Quá trình tối ưu hóa mô hình (Model Optimization Process) 22](#_Toc187351779)

[5.1.3. Feature Selection (Lựa chọn đặc trưng) 22](#_Toc187351780)

[5.1.4. Quá trình đánh giá mô hình (Model Evaluation Process) 23](#_Toc187351781)

[5.1.4.1. Báo cáo kết quả đánh giá 23](#_Toc187351782)

[5.1.4.2. Dự đoán mô hình 24](#_Toc187351783)

[5.1.4.3. Kết quả đánh giá và điều chỉnh mô hình 24](#_Toc187351784)

[5.1.4.4. Code Python hoàn chỉnh cho quá trình huấn luyện và đánh giá mô hình 25](#_Toc187351785)

[6. Thư viện xử lý dữ liệu 28](#_Toc187351786)

[6.1. Pandas 28](#_Toc187351787)

[6.2. Numpy 28](#_Toc187351788)

[6.3. Scikit-learn 28](#_Toc187351789)

[7. Đánh giá hiệu quả của các mô hình phân loại mã độc 29](#_Toc187351790)

[7.1. Đánh giá mô hình trực quan (Visualization) 31](#_Toc187351791)

[8.Kết luận và định hướng phát triển tương lai 35](#_Toc187351792)

[8.1. Kết Luận 35](#_Toc187351793)

[8.2. Định hướng phát triển tương lai 35](#_Toc187351794)

[Tài liệu kham khảo 37](#_Toc187351795)

*Danh sách bảng*

[Bảng 1: Công cụ hỗ trợ thu thập mã độc 19](#_Toc187277303)

[Bảng 2: Trích xuất đặc trưng của file PE 23](#_Toc187277304)

[Bảng 3: Đánh giá mô hình 29](#_Toc187277305)

**Lời Cảm Ơn**

*Với sự biết ơn sâu sắc dành cho những người đã cho tôi những lời chỉ dẫn, góp ý chân thành cùng sự động viên và khích lệ không ngừng để giúp tôi hoàn thành luận văn này, tôi xin được gửi gắm lời cảm ơn tới họ tại đây. Đầu tiên, em xin được bày tỏ lòng kính trọng và biết ơn sâu sắc đến thầy Nguyễn An Khương. Những đánh giá chi tiết cùng định hướng nền tảng thầy đặt ra vô cùng thực tế và hữu ích cho em trong quá trình nghiên cứu đề tài. Không chỉ dừng lại ở sự dẫn dắt, chỉ bảo về kiến thức, sự quan tâm sát sao và kĩ lưỡng của thầy cũng là nguồn cổ vũ to lớn để nhóm em có thể hoàn thành luân văn này. Nhóm em thật sự cảm thấy may mắn khi được hướng dẫn những bởi những thầy, người anh tận tâm và chu đáo như vậy. Lời nhận xét, góp ý cũng như sự động viên của thầy đã góp phần không nhỏ giúp tập nhóm của chúng tôi được hoàn thiện hơn rất nhiều.*

1. **Giới thiệu đề tài**
   1. **Đặt vấn đề**

Trong bối cảnh phát triển nhanh chóng của công nghệ thông tin, các cuộc tấn công mạng ngày càng trở nên phức tạp và nguy hiểm. Một trong những mối đe dọa lớn nhất đối với hệ thống mạng và dữ liệu là mã độc (malware), với các loại hình như virus, trojan, ransomware, spyware, và rootkit. Mã độc không chỉ gây thiệt hại về tài chính mà còn có thể làm suy yếu niềm tin vào các hệ thống điện toán và viễn thông. Việc phát hiện và phân loại mã độc một cách hiệu quả là nhiệm vụ quan trọng để bảo vệ hệ thống và dữ liệu người dùng.

Truyền thống, các phương pháp phát hiện mã độc chủ yếu dựa trên chữ ký (signature-based) hoặc phân tích hành vi (behavior-based). Tuy nhiên, các phương pháp này có một số hạn chế như không thể phát hiện được mã độc mới hoặc các biến thể của mã độc, và yêu cầu cập nhật liên tục các cơ sở dữ liệu chữ ký. Điều này tạo ra một lỗ hổng bảo mật lớn, vì những loại mã độc chưa được biết đến không thể bị phát hiện kịp thời.

Trong những năm gần đây, sự phát triển của trí tuệ nhân tạo (AI) và học máy (machine learning) đã mở ra những triển vọng mới trong việc phát hiện và phân loại mã độc. Các phương pháp học máy có khả năng học từ dữ liệu, nhận diện các mẫu và đặc điểm của mã độc mà không cần phải dựa vào các chữ ký sẵn có. Điều này giúp hệ thống bảo mật phát hiện được những mối đe dọa chưa biết, đồng thời nâng cao khả năng phân loại chính xác các loại mã độc.

Mặc dù có nhiều nghiên cứu đã chỉ ra tính hiệu quả của các phương pháp học máy trong việc phân loại mã độc, nhưng vẫn còn nhiều thách thức, như việc thu thập và xử lý dữ liệu huấn luyện, sự phức tạp của các mô hình học máy, và sự tương thích giữa các phương pháp học máy với các hệ thống bảo mật hiện tại. Do đó, nghiên cứu và ứng dụng các phương pháp học máy vào việc phân loại mã độc là một vấn đề quan trọng, không chỉ để cải thiện khả năng phát hiện mã độc mà còn để tăng cường sự bảo mật tổng thể của hệ thống.

Với những lý do trên, việc nghiên cứu, phát triển và ứng dụng các phương pháp học máy trong phân loại mã độc đang là một lĩnh vực được quan tâm và có tiềm năng to lớn trong việc giải quyết vấn đề bảo mật mạng hiện nay.

* 1. **Mục tiêu và phạm vi đề tài** 
     1. **Mục tiêu**

Mục tiêu chính của đề tài này là nghiên cứu và ứng dụng các phương pháp học máy để phân loại mã độc, nhằm nâng cao hiệu quả phát hiện và phòng ngừa các mối đe dọa mạng. Cụ thể, các mục tiêu của đề tài có thể được chia thành các nội dung sau:

* Nghiên cứu các phương pháp học máy trong phân loại mã độc
* Phát triển mô hình phân loại mã độc hiệu quả
* Tăng cường độ chính xác và khả năng phát hiện mã độc mới
* Áp dụng và tối ưu hóa dữ liệu huấn luyện
* Đánh giá và so sánh hiệu quả mô hình
* Đề xuất ứng dụng thực tế và giải pháp bảo mật
  + 1. **Phạm vi đề tài**

Đề tài này được xác định dựa trên các yếu tố chính như đối tượng nghiên cứu, phương pháp áp dụng và các giới hạn về tài nguyên cũng như thời gian. Cụ thể, phạm vi đề tài có thể được mô tả như sau:

**Đối tượng nghiên cứu**:

* **Mã độc (Malware)**: Nghiên cứu các loại mã độc phổ biến như virus, trojan, ransomware, spyware, worm, rootkit và các biến thể của chúng. Đặc biệt, tập trung vào các loại mã độc mới, các biến thể chưa được phát hiện, cũng như mã độc ẩn danh (obfuscated malware) hoặc có hành vi đáng ngờ (suspicious behavior).

**Phương pháp nghiên cứu**:

* **Học máy (Machine Learning)**: Nghiên cứu và áp dụng các phương pháp học máy giám sát (supervised learning) và học sâu (deep learning) để phân loại mã độc. Các thuật toán có thể được thử nghiệm bao gồm Decision Trees, Random Forests, Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), Neural Networks, và các mạng học sâu như CNN (Convolutional Neural Networks) hoặc RNN (Recurrent Neural Networks).

**Dữ liệu sử dụng**:

* **Bộ dữ liệu mã độc**: Sử dụng các bộ dữ liệu mã độc công khai có sẵn như CICIDS (Canadian Institute for Cybersecurity Intrusion Detection System), Kaggle, hoặc các bộ dữ liệu mã độc khác. Bộ dữ liệu này bao gồm các mẫu mã độc với đặc trưng chi tiết giúp xây dựng và huấn luyện mô hình học máy.
* **Tập huấn luyện và kiểm tra**: Dữ liệu huấn luyện sẽ được chia thành các tập huấn luyện và kiểm tra để đánh giá hiệu quả mô hình học máy.
  1. **Ý nghĩa thực tiễn**

Đề tài được nghiên cứu mang lại giá trị thực tiễn lớn trong việc nâng cao khả năng phát hiện và phòng chống mã độc. Học máy giúp phát hiện các mã độc mới và biến thể chưa biết mà phương pháp truyền thống không thể nhận diện, đồng thời giảm cảnh báo giả và nâng cao độ chính xác của hệ thống bảo mật. Việc ứng dụng học máy cũng giúp phát hiện và ngăn chặn tấn công mạng sớm, bảo vệ dữ liệu và giảm thiểu thiệt hại cho tổ chức, doanh nghiệp và người dùng cá nhân.

Ngoài ra, việc áp dụng học máy trong phân loại mã độc giúp giảm chi phí bảo trì và cập nhật cơ sở dữ liệu chữ ký, tối ưu hóa việc duy trì hệ thống bảo mật. Đề tài cũng thúc đẩy nghiên cứu và phát triển các phương pháp bảo mật thông minh, từ đó cung cấp giải pháp bảo vệ hiệu quả cho người dùng cá nhân và hệ thống mạng doanh nghiệp.

# **2. Định nghĩa mã độc và các loại mã độc phổ biến**

## **2.1. Mã độc là gì?**

**Mã độc (Malware)** là phần mềm hoặc mã máy tính được thiết kế để xâm nhập, làm hỏng hoặc gây rối loạn hoạt động của hệ thống máy tính, mạng máy tính hoặc thiết bị di động mà không có sự đồng ý của người dùng. Mã độc thường được sử dụng để thực hiện các hoạt động phá hoại, đánh cắp thông tin, hoặc kiểm soát hệ thống của người dùng mà không có sự phát hiện.

### **2.1.1. Các loại mã độc phổ biến**

* + **Virus** là một loại mã độc có khả năng sao chép và lây lan từ một máy tính này sang máy tính khác. Virus có thể lây lan qua các tệp đính kèm trong email, phần mềm tải về từ Internet, hoặc các thiết bị lưu trữ ngoài (USB, ổ cứng di động, v.v.). Virus có thể gây hư hỏng tệp tin hoặc làm gián đoạn hoạt động của hệ thống.
  + **Worm (Sâu máy tính)**: là một loại mã độc có khả năng tự sao chép và tự lây lan qua mạng mà không cần sự can thiệp của người dùng. Sâu có thể khai thác các lỗ hổng trong phần mềm hoặc giao thức mạng để xâm nhập vào hệ thống và lây lan đến các máy tính khác.
  + **Trojan (Trojan horse)**: là mã độc có dạng phần mềm hợp pháp hoặc phần mềm giả mạo, khiến người dùng vô tình cài đặt hoặc mở nó. Tuy nhiên, sau khi cài đặt, Trojan sẽ thực hiện các hành động nguy hiểm, chẳng hạn như đánh cắp thông tin cá nhân, tạo cửa hậu (backdoor) để kẻ tấn công điều khiển máy tính từ xa, hoặc phá hoại hệ thống.
  + **Ransomware (Phần mềm tống tiền)**: là loại mã độc khóa hoặc mã hóa dữ liệu của người dùng và yêu cầu tiền chuộc để mở khóa hoặc khôi phục lại dữ liệu. Ransomware có thể lây lan qua email, trang web độc hại, hoặc các phần mềm tải về. Nếu người dùng không trả tiền, dữ liệu có thể bị xóa hoặc mã hóa vĩnh viễn.
  + **Adware**: là phần mềm hiển thị quảng cáo không mong muốn trên máy tính của người dùng. Mặc dù không trực tiếp gây hại, nhưng adware có thể làm giảm hiệu suất hệ thống và gây khó chịu cho người dùng. Một số adware cũng thu thập thông tin về thói quen duyệt web của người dùng và gửi lại cho bên thứ ba mà không có sự đồng ý của người dùng.
  + **Spyware (Phần mềm gián điệp)**: là phần mềm được thiết kế để theo dõi hoạt động của người dùng mà không có sự cho phép của họ. Spyware có thể thu thập thông tin nhạy cảm, chẳng hạn như mật khẩu, thông tin ngân hàng, lịch sử duyệt web và gửi những dữ liệu này đến kẻ tấn công hoặc bên thứ ba.
  + **Rootkit**: là một loại mã độc được thiết kế để che giấu sự hiện diện của nó hoặc các phần mềm độc hại khác trong hệ thống. Rootkit có thể thay đổi cài đặt của hệ điều hành hoặc các chương trình bảo mật để người dùng hoặc phần mềm diệt virus không thể phát hiện hoặc xóa nó.
  + **Botnet**: là một mạng lưới các máy tính bị nhiễm mã độc (thường là trojan hoặc virus) và được điều khiển từ xa bởi một kẻ tấn công. Các máy tính trong botnet có thể được sử dụng để thực hiện các cuộc tấn công DDoS (tấn công từ chối dịch vụ phân tán), gửi thư rác, hoặc thực hiện các hành động trái phép khác mà người dùng không hề hay biết.
  + **Keylogger**: là phần mềm ghi lại tất cả các phím mà người dùng nhấn trên bàn phím. Keylogger thường được sử dụng để đánh cắp thông tin nhạy cảm như mật khẩu, thông tin ngân hàng hoặc các dữ liệu cá nhân khác.
  + **Backdoor**: là một cửa hậu trong hệ thống mà kẻ tấn công có thể sử dụng để truy cập vào máy tính hoặc mạng mà không cần xác thực. Backdoor thường được cài đặt thông qua Trojan hoặc virus và có thể cho phép kẻ tấn công điều khiển hệ thống từ xa mà không bị phát hiện.

### **2.1.2. Phân tích các phương pháp phát hiện mã độc truyền thống**

**a. Phân tích chữ ký (Signature-based Detection)**

Phương pháp phát hiện theo chữ ký là một trong những kỹ thuật phát hiện mã độc truyền thống phổ biến nhất. Nó hoạt động bằng cách so sánh các phần mềm hoặc tệp với một cơ sở dữ liệu chứa các chữ ký (signature) của các mã độc đã biết.

**Ưu điểm**:

* + Nhanh chóng và hiệu quả trong việc phát hiện các mã độc đã biết.
  + Ít gây tác động đến hiệu suất hệ thống khi thực hiện so sánh.

**Nhược điểm**:

* + Không phát hiện được mã độc chưa biết: Các biến thể mới của mã độc hoặc mã độc chưa có chữ ký trong cơ sở dữ liệu sẽ không bị phát hiện.
  + Cần cập nhật thường xuyên: Cơ sở dữ liệu chữ ký phải được cập nhật liên tục để theo kịp với các loại mã độc mới.

**b. Phân tích hành vi (Behavior-based Detection)**

Phát hiện theo hành vi tập trung vào việc theo dõi và phân tích hành vi của phần mềm khi nó chạy trong hệ thống. Các hành vi đáng ngờ như việc truy cập vào các tệp quan trọng, thay đổi cấu hình hệ thống, hay gửi dữ liệu không rõ nguồn gốc đến máy chủ từ xa có thể được ghi nhận và đánh dấu là có khả năng mã độc.

**Ưu điểm**:

* + Phát hiện được mã độc chưa biết: Phương pháp này có thể phát hiện các mã độc mới hoặc biến thể mã độc mà chưa có chữ ký trong cơ sở dữ liệu.
  + Giúp nhận diện hành vi bất thường: Phát hiện những hành động nguy hiểm mà phần mềm thực hiện, chẳng hạn như truy cập trái phép vào dữ liệu nhạy cảm.

**Nhược điểm**:

* + Tốn tài nguyên hệ thống: Việc theo dõi và phân tích hành vi của phần mềm có thể ảnh hưởng đến hiệu suất của hệ thống.
  + Cần phải có ngưỡng xác định hành vi đáng ngờ: Đôi khi các phần mềm hợp pháp cũng có hành vi tương tự với mã độc, gây ra cảnh báo giả.

**c. Phân tích tĩnh (Static Analysis)**

Phân tích tĩnh là phương pháp phân tích phần mềm mà không cần thực thi mã. Quá trình này sẽ phân tích mã nguồn hoặc mã máy (binary) của phần mềm để tìm ra các đặc điểm hoặc mẫu mã độc có thể có, chẳng hạn như các hàm mã độc, sự kết nối đến các máy chủ từ xa đáng ngờ, hoặc các lệnh hệ thống không hợp lệ.

**Ưu điểm**:

* + Phát hiện sớm: Vì không cần phải thực thi mã, phương pháp này giúp phát hiện mã độc trước khi nó gây hại.
  + An toàn: Không làm ảnh hưởng đến hệ thống vì không yêu cầu chạy phần mềm.

**Nhược điểm**:

* + Khó khăn với mã đã được mã hóa hoặc ẩn giấu: Các mã độc có thể sử dụng các kỹ thuật như mã hóa hoặc đóng gói (packing) để ẩn mình, khiến việc phân tích tĩnh trở nên khó khăn.
  + Không thể phát hiện hành vi mã độc: Phân tích tĩnh không thể quan sát được hành vi của phần mềm khi chạy trong hệ thống, điều này làm giảm khả năng phát hiện các mối nguy tiềm ẩn.

**d. Phân tích động (Dynamic Analysis)**

Phân tích động là phương pháp phân tích mã độc khi phần mềm đang thực thi trong môi trường giả lập hoặc trong hệ thống thực tế (sandbox). Quá trình này theo dõi hành vi của phần mềm và xem xét cách nó tương tác với hệ thống, tệp tin, mạng, và các tài nguyên hệ thống khác.

**Ưu điểm**:

* + Phát hiện hành vi mã độc: Giúp phát hiện những hành động mà phần mềm thực hiện trong quá trình hoạt động, chẳng hạn như truy cập trái phép hoặc thay đổi hệ thống.
  + Có thể phát hiện mã độc chưa biết: Phát hiện những hành vi bất thường không dựa trên chữ ký hay mẫu có sẵn.

**Nhược điểm**:

* + Cần tài nguyên: Việc theo dõi hành vi và thực thi phần mềm trong môi trường giả lập yêu cầu tài nguyên phần cứng và phần mềm đáng kể.
  + Không phải lúc nào cũng hiệu quả: Một số mã độc có thể nhận diện được môi trường giả lập và thay đổi hành vi của chúng để tránh bị phát hiện.

**e. Phân tích heuristic (Heuristic Analysis)**

Phân tích heuristic sử dụng các thuật toán và phương pháp thống kê để phát hiện mã độc dựa trên các mẫu hành vi hoặc đặc điểm mã độc điển hình. Phương pháp này không dựa vào chữ ký cụ thể mà tìm kiếm các dấu hiệu hoặc hành vi có thể chỉ ra mã độc.

**Ưu điểm**:

* + Phát hiện mã độc chưa biết: Nhờ vào việc phát hiện các đặc điểm hoặc hành vi đáng ngờ, phương pháp này có thể phát hiện các mối đe dọa mới mà không cần có chữ ký.
  + Cải thiện khả năng phát hiện: Có thể phát hiện các mối đe dọa mà các phương pháp phát hiện chữ ký không thể nhận diện được.

**Nhược điểm**:

* + Cảnh báo giả: Phân tích heuristic có thể tạo ra nhiều cảnh báo giả nếu không có các thuật toán chính xác.
  + Khó khăn trong việc tối ưu hóa: Các thuật toán heuristic cần phải được cải thiện thường xuyên để giảm thiểu cảnh báo giả và tăng độ chính xác.

### **2.1.3. Những thách thức trong phát hiện mã độc truyền thống**

**Mã độc chưa biết và biến thể mới**

Thách thức: Các phương pháp phát hiện truyền thống, đặc biệt là phương pháp phát hiện theo chữ ký, không thể phát hiện được các mã độc mới chưa có chữ ký hoặc biến thể mới của mã độc đã biết. Khi kẻ tấn công thay đổi hoặc mã hóa mã độc, chữ ký cũ trở nên vô dụng.

**Mã độc tinh vi và ẩn giấu**

Thách thức: Mã độc ngày càng trở nên tinh vi hơn, sử dụng các kỹ thuật như mã hóa (encryption), đóng gói (packing), hoặc thay đổi mã (polymorphic, metamorphic) để tránh bị phát hiện bởi các công cụ chống virus truyền thống. Các mã độc này có thể thay đổi dạng thức sau mỗi lần thực thi, khiến cho các cơ sở dữ liệu chữ ký khó theo kịp.

**Cảnh báo giả (False Positives)**

Thách thức: Phương pháp phát hiện theo hành vi và phân tích heuristic có thể tạo ra cảnh báo giả khi những phần mềm hợp pháp thực hiện các hành động tương tự hành vi của mã độc. Điều này có thể làm giảm hiệu quả và độ tin cậy của hệ thống bảo mật, khiến người dùng hoặc quản trị viên không chú ý đến các mối nguy thật sự.

**Kỹ thuật tấn công ngày càng phức tạp**

Thách thức: Các kỹ thuật tấn công ngày càng trở nên phức tạp hơn, bao gồm kỹ thuật rootkit, backdoor, hoặc các kỹ thuật điều khiển từ xa mà không cần tiếp xúc trực tiếp với hệ thống mục tiêu. Những mã độc này có thể làm việc ẩn danh trong hệ thống mà không tạo ra dấu vết rõ ràng.

**Quá tải thông tin và khối lượng dữ liệu lớn**

Thách thức: Các hệ thống bảo mật cần phải xử lý một lượng lớn dữ liệu và các sự kiện từ nhiều nguồn khác nhau, bao gồm các thiết bị, ứng dụng, và các phương tiện lưu trữ. Điều này có thể tạo ra quá tải thông tin và làm giảm khả năng phát hiện mã độc kịp thời.

**Thiếu khả năng nhận diện mã độc đa nền tảng**

Thách thức: Mã độc hiện nay không chỉ nhắm đến một hệ điều hành hoặc nền tảng duy nhất (như Windows), mà có thể lây lan trên nhiều nền tảng như Linux, macOS, Android, và iOS. Điều này đòi hỏi các phương pháp phát hiện mã độc phải đa nền tảng, điều này tạo ra một thách thức lớn trong việc phát triển và duy trì các công cụ phát hiện hiệu quả trên tất cả các nền tảng.

**Tấn công bằng các kỹ thuật xã hội (Social Engineering)**

Thách thức: Mã độc có thể được phát tán thông qua các cuộc tấn công xã hội như phishing, spear-phishing, hoặc các chiến dịch email giả mạo. Những cuộc tấn công này không cần phải dựa vào mã độc truyền thống, mà thay vào đó nhắm vào người dùng để lừa họ cài đặt hoặc mở tệp chứa mã độc.

**Tấn công từ bên trong (Insider Threats)**

Thách thức: Mã độc không chỉ đến từ bên ngoài mà còn có thể được cài đặt hoặc phát tán bởi chính những người bên trong tổ chức, như nhân viên hoặc các đối tác. Những cuộc tấn công này thường khó phát hiện vì chúng được thực hiện bởi những người có quyền truy cập hợp pháp vào hệ thống.

**Thời gian phản ứng chậm**

Thách thức: Các phương pháp phát hiện truyền thống đôi khi không thể phản ứng kịp thời với các cuộc tấn công, đặc biệt là khi mã độc có thể lây lan hoặc thay đổi trong thời gian ngắn. Việc phát hiện và phản hồi kịp thời là rất quan trọng để hạn chế thiệt hại.

### **2.1.4. Tổng quan về các phương pháp học máy trong bảo mật**

**a. Học máy giám sát (Supervised Learning)**

Học máy giám sát là phương pháp học máy mà trong đó mô hình được huấn luyện trên một tập dữ liệu đã được gán nhãn. Mục tiêu là học cách phân loại hoặc dự đoán các nhãn (labels) từ các đặc trưng của dữ liệu (features). Trong bảo mật mạng, các ứng dụng của học máy giám sát bao gồm:

* Phát hiện xâm nhập (Intrusion Detection Systems - IDS): Hệ thống IDS sử dụng học máy giám sát để phát hiện các hành vi xâm nhập hoặc tấn công. Các mô hình được huấn luyện trên các tập dữ liệu có chứa các cuộc tấn công và hành vi hợp lệ, sau đó sử dụng các đặc trưng như IP, cổng, gói tin mạng để phân loại lưu lượng mạng là hợp lệ hay có tấn công.
* Phân loại email phishing: Các mô hình học máy giám sát có thể phân loại email vào các nhóm hợp lệ hoặc phishing (lừa đảo), dựa trên các đặc trưng như nội dung, tiêu đề email, hoặc địa chỉ người gửi.
* Phát hiện mã độc (Malware Detection): Học máy giám sát có thể phân tích các tệp hoặc hành vi trong hệ thống và phân loại chúng thành các nhóm mã độc và mã hợp lệ.
* Phát hiện tấn công DDoS: Mô hình học máy giám sát có thể giúp phát hiện các cuộc tấn công từ chối dịch vụ phân tán (DDoS) dựa trên các đặc điểm lưu lượng mạng.

**Các thuật toán học máy giám sát phổ biến:**

* Hồi quy Logistic (Logistic Regression)
* Cây quyết định (Decision Trees)
* Máy vector hỗ trợ (Support Vector Machines - SVM)
* K-nearest neighbor (K-NN)
* Mạng nơ-ron nhân tạo (Artificial Neural Networks - ANN)

**b. Học máy không giám sát (Unsupervised Learning)**

Học máy không giám sát là phương pháp học máy trong đó mô hình không có nhãn dữ liệu và mục tiêu là tìm ra cấu trúc ẩn trong dữ liệu. Các ứng dụng của học máy không giám sát trong bảo mật mạng bao gồm:

* **Phát hiện mối đe dọa mới**: Hệ thống có thể phát hiện các mẫu hành vi bất thường hoặc mối đe dọa mới mà chưa có dữ liệu mẫu. Phát hiện này có thể dựa trên việc tìm các nhóm lưu lượng hoặc hành vi không bình thường trong mạng.
* **Phát hiện hành vi bất thường (Anomaly Detection)**: Các mô hình học máy không giám sát có thể phát hiện hành vi bất thường trong các hệ thống mạng mà không cần phải có dữ liệu về các cuộc tấn công trước đó. Điều này rất hữu ích khi đối mặt với các mối đe dọa chưa biết.

**Các thuật toán học máy không giám sát phổ biến:**

* K-means clustering
* DBSCAN (Density-Based Spatial Clustering of Applications with Noise)
* Principal Component Analysis (PCA)
* Autoencoders (sử dụng trong phát hiện bất thường)

**c. Học máy bán giám sát (Semi-Supervised Learning)**

Học máy bán giám sát kết hợp cả hai phương pháp giám sát và không giám sát. Nó sử dụng một lượng nhỏ dữ liệu đã được gán nhãn và một lượng lớn dữ liệu không được gán nhãn để huấn luyện mô hình. Phương pháp này rất hữu ích khi có ít dữ liệu được gán nhãn nhưng lại có nhiều dữ liệu không gán nhãn có sẵn.

# **3. Giới thiệu các thuật toán học máy**

## **3.1. Random Forest**

Random Forest là một thuật toán học máy thuộc nhóm ensemble learning, nghĩa là thuật toán này kết hợp nhiều mô hình học máy yếu để tạo thành một mô hình mạnh mẽ. Cụ thể, Random Forest sử dụng một tập hợp các cây quyết định (Decision Trees) để thực hiện phân loại hoặc hồi quy, và kết quả cuối cùng được đưa ra bằng cách bỏ phiếu từ các cây quyết định trong rừng.

**Cách hoạt động:**

* + Bootstrap Aggregating (Bagging): Thuật toán này tạo ra nhiều mẫu con từ tập dữ liệu gốc thông qua phương pháp bootstrap sampling (chọn mẫu ngẫu nhiên với hoàn lại), sau đó huấn luyện một cây quyết định cho mỗi mẫu con.
  + Chọn đặc trưng ngẫu nhiên: Trong mỗi nút của cây quyết định, thay vì sử dụng tất cả các đặc trưng, thuật toán chỉ chọn ngẫu nhiên một tập con của các đặc trưng để xây dựng cây quyết định.
  + Kết hợp kết quả: Khi đã có nhiều cây quyết định, kết quả phân loại (hoặc dự đoán hồi quy) được đưa ra bằng cách bỏ phiếu đa số (cho bài toán phân loại) hoặc lấy trung bình (cho bài toán hồi quy).

**Ưu điểm:**

* Khả năng chống overfitting tốt nhờ việc sử dụng nhiều cây và việc chọn đặc trưng ngẫu nhiên.
* Dễ sử dụng và có thể giải quyết các bài toán phân loại và hồi quy.
* Tốt cho các bài toán có dữ liệu phức tạp, không cần chuẩn bị dữ liệu quá kỹ.

**Nhược điểm:**

* Mô hình có thể trở nên khó giải thích vì nó sử dụng nhiều cây.
* Tốc độ huấn luyện và dự đoán có thể chậm nếu rừng có quá nhiều cây.

## **3.2. K-Nearest Neighbors (KNN)**

K-Nearest Neighbors (KNN) là một thuật toán học máy đơn giản và dễ hiểu, thuộc nhóm thuật toán học máy không giám sát. KNN được sử dụng chủ yếu trong các bài toán phân loại và hồi quy, hoạt động dựa trên nguyên lý "các điểm dữ liệu giống nhau sẽ gần nhau trong không gian đặc trưng".

**Cách hoạt động:**

* Phân loại: Để phân loại một điểm dữ liệu, thuật toán tìm kiếm K điểm gần nhất (neighbors) trong không gian đặc trưng của nó (theo khoảng cách Euclidean hoặc các đo lường khác). Kết quả phân loại của điểm mới là lớp chiếm đa số trong số K điểm gần nhất.
* Hồi quy: Thay vì trả về lớp chiếm đa số, KNN sẽ tính giá trị trung bình (hoặc trung vị) của các giá trị mục tiêu của K điểm gần nhất để dự đoán giá trị liên tục.

**Ưu điểm:**

* Đơn giản và dễ hiểu.
* Không yêu cầu huấn luyện mô hình, vì việc phân loại hay dự đoán chỉ xảy ra khi có yêu cầu.
* Phù hợp với các bài toán có không gian đặc trưng nhỏ và dữ liệu không có sự phân tách rõ ràng.

**Nhược điểm:**

* Hiệu suất kém với dữ liệu có chiều cao (high-dimensional data) vì "curse of dimensionality" (tính phức tạp tăng lên khi số chiều của dữ liệu tăng).
* Tính toán tốn kém, nhất là khi có lượng lớn dữ liệu vì thuật toán phải tính khoảng cách với tất cả các điểm dữ liệu trong bộ dữ liệu.

## **3.3. Decision Tree**

Cây quyết định (Decision Tree) là một thuật toán học máy giám sát được sử dụng trong các bài toán phân loại và hồi quy. Thuật toán này xây dựng một mô hình dưới dạng cây, trong đó mỗi nút là một câu hỏi hoặc điều kiện phân loại, và các nhánh thể hiện kết quả của các câu hỏi đó.

**Cách hoạt động:**

Chọn đặc trưng: Cây quyết định được xây dựng thông qua việc chia nhỏ tập dữ liệu tại mỗi nút dựa trên các đặc trưng của nó. Mục tiêu là tìm cách chia dữ liệu sao cho các phân nhóm tại mỗi nút có sự khác biệt lớn nhất giữa các lớp (trong bài toán phân loại) hoặc các giá trị mục tiêu (trong bài toán hồi quy).

**Tiêu chí phân chia: Các tiêu chí phổ biến để chia dữ liệu là:**

* + Gini Impurity: Đo lường mức độ không thuần nhất của các lớp tại một nút.
  + Entropy: Đo lường mức độ không chắc chắn trong việc phân loại.
  + Mean Squared Error (MSE): Được sử dụng trong các bài toán hồi quy.

**Ưu điểm:**

* Dễ hiểu và dễ trực quan hóa.
* Không yêu cầu tiền xử lý dữ liệu phức tạp (ví dụ: không cần chuẩn hóa hoặc chuyển đổi dữ liệu).
* Có thể xử lý cả dữ liệu phân loại và dữ liệu liên tục.

**Nhược điểm:**

* **Overfitting**: Cây quyết định dễ bị overfit nếu không được cắt tỉa (pruning) hoặc điều chỉnh độ sâu của cây.
* **Không ổn định**: Một thay đổi nhỏ trong dữ liệu có thể dẫn đến một cấu trúc cây hoàn toàn khác nhau.
* **Giải thích mô hình**: Khi cây trở nên quá sâu và phức tạp, nó có thể khó giải thích và trực quan hóa.
  1. **Phân tích các kỹ thuật học máy được sử dụng trong phân loại mã độc**
  2. **Random Forest**

**Random Forest** là một thuật toán **ensemble learning** sử dụng tập hợp nhiều cây quyết định (Decision Trees) để cải thiện độ chính xác và giảm overfitting.

**Cách thức hoạt động:**

**Bootstrap Aggregating (Bagging)**:

Một trong những kỹ thuật chính của Random Forest là sử dụng **Bagging**, hay còn gọi là "Bootstrap Aggregating". Từ bộ dữ liệu huấn luyện ban đầu, ta tạo ra nhiều mẫu con (subsets) ngẫu nhiên bằng cách chọn ngẫu nhiên các điểm dữ liệu với phép lặp lại. Mỗi mẫu con này sẽ được dùng để huấn luyện một cây quyết định độc lập.

*Giả sử, ta có một bộ dữ liệu huấn luyện*

|  |
| --- |
|  |

Các mẫu con *D1,D2*,…,*DT* ​ được lấy từ D, và mỗi mẫu con sẽ huấn luyện một cây quyết định.

**Quy trình huấn luyện các cây quyết định**:

Mỗi cây trong Random Forest được huấn luyện trên một mẫu con khác nhau.

Trong mỗi nút của mỗi cây quyết định, thuật toán không xét tất cả các đặc trưng mà chỉ chọn một tập con ngẫu nhiên của các đặc trưng.

Giả sử tập đặc trưng là:



Tại mỗi nút, thuật toán chọn ngẫu nhiên *m* đặc trưng (*với m <p*) để tìm đặc trưng tối ưu phân chia.

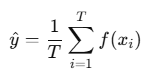
**Kết quả**:

* + Phân loại: Sau khi huấn luyện xong các cây, Random Forest đưa ra kết quả bằng cách bỏ phiếu đa số. Mỗi cây sẽ "bỏ phiếu" cho một lớp, và lớp có nhiều phiếu nhất sẽ được chọn làm kết quả.



(Trong đó  là dự đoán từ cây thứ i, và mode là giá trị xuất hiện nhiều nhất)

* + **Hồi quy**: Trong bài toán hồi quy, kết quả được tính là **trung bình** các dự đoán từ các cây quyết định.



**Công thức tính độ chính xác của cây quyết định:**

* **Gini Impurity**: Được sử dụng trong phân loại, là một thước đo mức độ không thuần nhất của các lớp tại một nút:

A number with a number on it

Description automatically generated with medium confidence

(Trong đó ​ là xác suất của lớp iii trong bộ dữ liệu D và k là số lớp)

* **Entropy**: Được sử dụng để đánh giá độ không chắc chắn của việc phân loại, với công thức:

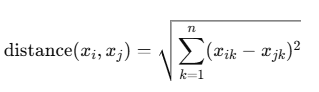
A black and white math equation

Description automatically generated

* 1. **K-Nearest Neighbors (KNN)**

**Phân loại**:

Để phân loại một điểm dữ liệu mới x0​, thuật toán tìm **K điểm gần nhất** với x0​ trong tập dữ liệu huấn luyện. Các điểm gần nhất được xác định dựa trên một **đo lường khoảng cách** (thường là khoảng cách Euclidean).



**Dự đoán**:

Kết quả phân loại của điểm x0x\_0x0​ sẽ là lớp chiếm đa số trong số các điểm gần nhất.

A math equation with black text

Description automatically generated with medium confidence

**Hồi quy**:

Trong bài toán hồi quy, dự đoán sẽ là giá trị trung bình của các giá trị mục tiêu của K điểm gần nhất.

A mathematical equation with numbers and symbols

Description automatically generated

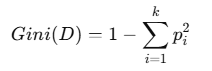
* 1. **Decision Tree (Cây quyết định)**

**Tiêu chí phân chia**:

Tại mỗi nút của cây, thuật toán chọn một đặc trưng để phân chia dữ liệu. Mục tiêu là tìm đặc trưng sao cho nó chia dữ liệu thành các nhóm sao cho các nhóm này càng thuần nhất càng tốt.

*Các tiêu chí phổ biến để lựa chọn đặc trưng là:*

**Gini Impurity** (được sử dụng trong phân loại):



(Trong đó *pi* là xác suất của lớp *i* trong tập *D*)

**Entropy** (còn gọi là độ hỗn loạn hoặc độ không chắc chắn trong phân loại):

A black and white math equation

Description automatically generated

(Trong đó *pi* là xác suất của lớp *i*)

**Mean Squared Error (MSE)** (được sử dụng trong hồi quy):

A black and white math equation

Description automatically generated

**Phân chia**:

Cây sẽ tiếp tục phân chia dữ liệu tại mỗi nút cho đến khi một trong các điều kiện dừng sau đây xảy ra:

* + Tất cả các điểm trong nút đã thuộc cùng một lớp (hoặc là các giá trị mục tiêu giống nhau trong hồi quy).
  + Độ sâu của cây vượt quá một ngưỡng nhất định.
  + Số lượng điểm dữ liệu trong một nút thấp hơn một ngưỡng tối thiểu.

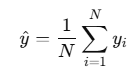
**Dự đoán**:

Khi một điểm dữ liệu mới đi qua cây, nó sẽ đi qua các nút từ gốc đến các nút lá. Mỗi nút trong cây thực hiện một kiểm tra điều kiện để xác định nhánh tiếp theo. Cuối cùng, điểm dữ liệu sẽ đến một nút lá, nơi nó nhận được dự đoán.

**Phân loại**: Kết quả là lớp chiếm đa số trong các điểm dữ liệu ở nút lá.



**Hồi quy**: Kết quả là giá trị trung bình của các giá trị mục tiêu trong nút lá.



* 1. **Cách thức trích xuất đặc trưng mã độc từ dữ liệu tĩnh và động**
     1. **Trích xuất đặc trưng mã độc từ dữ liệu tĩnh**

Dữ liệu tĩnh là thông tin thu thập từ tệp mã độc mà không cần thực thi nó. Phân tích dữ liệu tĩnh giúp tránh rủi ro lây nhiễm mã độc và cung cấp cái nhìn sâu về cấu trúc của tệp.

**Các đặc trưng tĩnh phổ biến cần trích xuất**

* **Metadata:** Tên tệp, kích thước tệp, ngày tạo, hash (MD5, SHA-1, SHA-256).
* **Header Information:**
* Thông tin trong PE (Portable Executable) header, ELF header (trên Linux).
* Entry point, kích thước các section, các thư viện được liên kết.
* **Opcode Sequence (Chuỗi mã lệnh):** Trích xuất các mã lệnh opcode từ file thực thi.
* **Danh sách API/Imports:** Các hàm API mà mã độc gọi đến (ví dụ: CreateFile, VirtualAlloc).
* **Strings:** Các chuỗi văn bản được lưu trữ trong file, như: URL, tên miền, địa chỉ IP, thông điệp lỗi.
* **Entropy:** Đo mức độ ngẫu nhiên của tệp (entropy cao có thể là dấu hiệu mã hóa hoặc packer).
  + 1. **Trích xuất đặc trưng mã độc từ dữ liệu động**

Dữ liệu động là thông tin thu thập được khi thực thi mã độc trong môi trường ảo hóa hoặc sandbox để quan sát hành vi thực tế của nó.

**Các đặc trưng động phổ biến cần trích xuất**

* **Quá trình tạo file, registry, và network**
  + Các file mới được tạo hoặc chỉnh sửa.
  + Các khóa registry bị chỉnh sửa hoặc thêm mới.
  + Các kết nối mạng ra ngoài (địa chỉ IP, cổng, giao thức).
* **Hành vi của tiến trình (Process Behavior)**
  + Quá trình nào được tạo ra, ID của tiến trình, các API được gọi.
  + Ghi nhận các tiến trình con được sinh ra.
  + **Hooking API và Inject Code:** Mã độc thường hook vào các hàm API quan trọng để giám sát hoặc kiểm soát hệ thống.
* **Phân tích bộ nhớ (Memory Analysis)**
  + Ghi nhận các thay đổi trong bộ nhớ khi mã độc hoạt động.
  + Trích xuất mã được giải mã hoặc giải nén trong bộ nhớ.

## **4.5. Quy trình thu thập và xử lý dữ liệu mã độc**

Việc thu thập và xử lý dữ liệu mã độc là bước quan trọng trong nghiên cứu bảo mật, giúp phát hiện và ngăn chặn các mối đe dọa tiềm tàng. Dưới đây là hướng dẫn chi tiết về các giai đoạn chính của quy trình, từ thu thập dữ liệu mã độc đến xử lý, phân tích, và lưu trữ kết quả.

### **Giai đoạn 1**: Thu thập dữ liệu mã độc (Malware Collection)

* Nguồn thu thập mã độc phổ biến
  + Threat Intelligence Feeds: Các nguồn thông tin mối đe dọa từ các tổ chức bảo mật (VirusTotal, MalwareBazaar, Abuse.ch, Hybrid Analysis).
  + HoneyPot Systems: Hệ thống giả lập để thu hút mã độc từ các cuộc tấn công mạng.
  + Dark Web & Forums: Các diễn đàn ngầm và thị trường chợ đen.
  + Public Repositories: Các kho lưu trữ công khai như GitHub hoặc các diễn đàn nghiên cứu bảo mật.
  + Phishing Emails: Thu thập mã độc từ các chiến dịch email lừa đảo.
* Công cụ hỗ trợ thu thập mã độc

|  |  |  |
| --- | --- | --- |
| **Công cụ** | **Mô tả** | **Link** |
| VirusTotal API | Dịch vụ phân tích file/tệp hash | https://www.virustotal.com |
| MalwareBazaar | Kho lưu trữ mã độc công khai | https://bazaar.abuse.ch |
| Cuckoo Sandbox | Sandbox để phân tích hành vi mã độc | https://cuckoosandbox.org |
| Dionaea | Honeypot để thu thập mã độc | https://github.com/DinoTools/dionaea |
| SpiderFoot | OSINT công cụ tự động hóa thu thập dữ liệu | https://www.spiderfoot.net |

Bảng 1: Công cụ hỗ trợ thu thập mã độc

**Giai đoạn 2:** Tiền xử lý dữ liệu mã độc (Preprocessing)

* Lọc và xác minh tệp mã độc
* Loại bỏ các tệp trùng lặp dựa trên hash (MD5, SHA-256).
* Kiểm tra định dạng file (PE file, ELF file, Script file).
  + Gắn nhãn (Labeling)
* Mã độc (Malicious) hoặc Không độc hại (Benign).
* Dựa vào thông tin từ VirusTotal hoặc các dịch vụ Threat Intelligence.
* Phân loại theo gia đình mã độc (Malware Family): Ví dụ: Ransomware, Trojan, Spyware, Worm, Adware.
  + Phân tích tĩnh cơ bản: Trích xuất các đặc trưng tĩnh như hash, kích thước file, thư viện API.

**Giai đoạn 3:** Xử lý dữ liệu mã độc (Data Processing)

* Trích xuất đặc trưng tĩnh (Static Feature Extraction)
  + Metadata (hash, kích thước file, thời gian tạo).
  + Opcode sequence.
  + Danh sách API được import.
* Trích xuất đặc trưng động (Dynamic Feature Extraction)
  + Hành vi tiến trình, kết nối mạng, và thay đổi registry.
  + Thu thập thông tin về mã độc khi nó thực thi trong môi trường giả lập.
* Chuẩn hóa dữ liệu (Normalization): Chuyển đổi các đặc trưng về cùng một định dạng.
* Lưu trữ vào cơ sở dữ liệu (Database Storage): Lưu trữ dữ liệu mã độc đã xử lý vào cơ sở dữ liệu NoSQL (MongoDB) hoặc SQL để phục vụ quá trình phân tích sau này.

### **Giai đoạn 4:** Phân tích và nhận diện mã độc (Malware Analysis)

* Phân tích tĩnh: Sử dụng các công cụ như IDA Pro, Ghidra để phân tích mã nguồn và cấu trúc tệp.
* Phân tích động: Sử dụng sandbox như Cuckoo hoặc Any.Run để quan sát hành vi mã độc trong môi trường thực thi.
* Áp dụng Machine Learning: Sử dụng các thuật toán học máy (Random Forest, SVM, Deep Learning) để phát hiện mã độc dựa trên đặc trưng đã trích xuất.

### **Giai đoạn 5:** Lưu trữ và báo cáo (Storage & Reporting)

* Lưu trữ dữ liệu:
  + Lưu vào cơ sở dữ liệu để tiện cho việc tìm kiếm và cập nhật.
  + Bảo mật dữ liệu để tránh lộ lọt mẫu mã độc.
* Tạo báo cáo: Tạo báo cáo chi tiết về mã độc, bao gồm hành vi, đặc trưng tĩnh/động và phương pháp phát hiện.

1. **Phân tích thuật toán học máy để xây dựng mô hình phân loại**

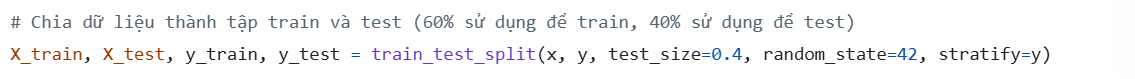
* **Tiền xử lý dữ liệu**

Trước khi chọn thuật toán, dữ liệu cần được xử lý và chuẩn hóa để phù hợp với mô hình học máy. Các bước bao gồm:

* + Loại bỏ dữ liệu trùng lặp và không hợp lệ.
  + Gắn nhãn (label) dữ liệu thành 2 nhóm: Malicious và Benign.
  + Trích xuất các đặc trưng tĩnh và động.
  + Chuyển đổi dữ liệu thành dạng số để làm đầu vào cho mô hình.
* **Các thuật toán học máy phân loại mã độc**
* Thuật toán Random Forest (RF)
* Thuật toán K-Nearest Neighbors (KNN)
* Thuật toán Decision Tree
  1. **Quá trình huấn luyện, tối ưu hóa và đánh giá mô hình.**
     1. **Quá trình huấn luyện mô hình (Model Training Process)**

**Bước 1:** Chuẩn bị dữ liệu huấn luyện và kiểm tra

* Training Set (Dữ liệu huấn luyện): Để mô hình học các mẫu và đặc trưng của dữ liệu.
* Test Set (Dữ liệu kiểm tra): Để đánh giá hiệu suất mô hình trên dữ liệu mới.



**Quy tắc chia dữ liệu:** 75% cho huấn luyện, 25% cho kiểm tra.

**Bước 2:** Lựa chọn thuật toán và khởi tạo mô hình

* **Thuật toán Random Forest (RF)**

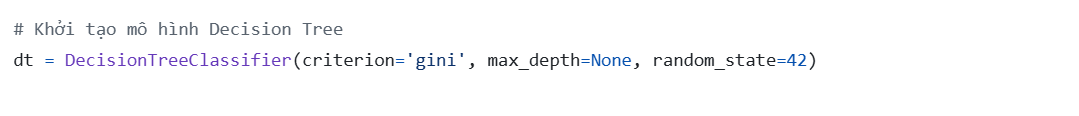
**A white background with black text

Description automatically generated**

* **Thuật toán K-Nearest Neighbors (KNN)**

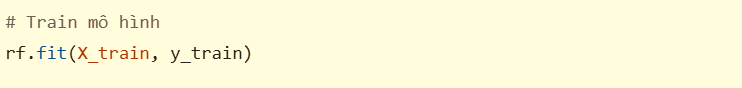
****

* **Thuật toán Decision Tree**

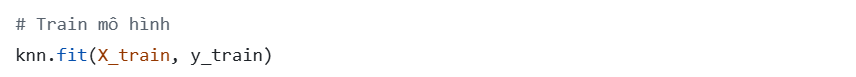
****

**Bước 3:** Huấn luyện mô hình (Training)

* **Thuật toán Random Forest (RF)**

****

* **Thuật toán K-Nearest Neighbors (KNN)**

****

* **Thuật toán Decision Tree**

****

**Trong đó:**

* X\_train là dữ liệu đầu vào.
* y\_train là nhãn phân loại (malicious hoặc benign).
  + 1. **Quá trình tối ưu hóa mô hình (Model Optimization Process)**

Trong thuật toán K-Nearest Neighbors (KNN) sử dụng GridSearchCV để tìm kiếm toàn bộ không gian siêu tham số.

A screenshot of a computer code

Description automatically generated

* + 1. **Feature Selection (Lựa chọn đặc trưng)**

Lựa chọn các đặc trưng quan trọng để giảm bớt nhiễu và tăng hiệu suất mô hình.

|  |  |  |  |
| --- | --- | --- | --- |
| IMAGE\_DOS\_HEADER (6)   * e\_cblp * e\_cp * e\_cparhdr * e\_maxalloc * e\_sp * e\_lfanew | FILE\_HEADER (17)   * NumberOfSections * CreationYear * FH\_char0 * FH\_char1 * FH\_char2 * FH\_char3 * FH\_char4 * FH\_char5 * FH\_char6 * FH\_char7 * FH\_char8 * FH\_char9 * FH\_char10 * FH\_char11 * FH\_char12 * FH\_char13 * FH\_char14 | OPTIONAL\_HEADER (37)   * MajorLinkerVersion * MinorLinkerVersion * SizeOfCode * SizeOfInitializedData * SizeOfUninitializedData * AddressOfEntryPoint * BaseOfCode * BaseOfData * ImageBase * SectionAlignment * FileAlignment * MajorOperatingSystemVersion * MinorOperatingSystemVersion * MajorImageVersion * MinorImageVersion * MajorSubsystemVersion * MinorSubsystemVersion * SizeOfImage * SizeOfHeaders * CheckSum * Subsystem * OH\_DLLchar0 * OH\_DLLchar1 * OH\_DLLchar2 * OH\_DLLchar3 * OH\_DLLchar4 * OH\_DLLchar5 * OH\_DLLchar6 * OH\_DLLchar7 * OH\_DLLchar8 * OH\_DLLchar9 * OH\_DLLchar10 * SizeOfStackReserve * SizeOfStackCommit * SizeOfHeapReserve * SizeOfHeapCommit * LoaderFlags | Derived Header (8)   * sus\_sections * non\_sus\_sections * packer * packer\_type * E\_text * E\_data * filesize * E\_file * fileinfo |

Bảng 2: Trích xuất đặc trưng của file PE

* + 1. **Quá trình đánh giá mô hình (Model Evaluation Process)**
       1. **Báo cáo kết quả đánh giá**
* **Thuật toán Random Forest (RF)**

**A screenshot of a computer program

Description automatically generated**

* **Thuật toán K-Nearest Neighbors (KNN)**

**A computer screen shot of a computer program

Description automatically generated**

* **Thuật toán Decision Tree**

**A computer code with text

Description automatically generated with medium confidence**

* + - 1. **Dự đoán mô hình**

**A computer code with text

Description automatically generated with medium confidence**

* + - 1. **Kết quả đánh giá và điều chỉnh mô hình**
* Nếu mô hình bị **underfitting** (hiệu suất thấp trên cả dữ liệu huấn luyện và kiểm tra): Tăng số lượng đặc trưng hoặc sử dụng mô hình phức tạp hơn.
* Nếu mô hình bị **overfitting** (hiệu suất cao trên dữ liệu huấn luyện nhưng thấp trên dữ liệu kiểm tra): Sử dụng kỹ thuật regularization hoặc giảm số lượng đặc trưng.
  + - 1. **Code Python hoàn chỉnh cho quá trình huấn luyện và đánh giá mô hình**

**Mô hình Random Forest**

**A screenshot of a computer code

Description automatically generated**

**A computer screen shot of a computer code

Description automatically generated**

**Thuật toán K-Nearest Neighbors (KNN)**

**A screenshot of a computer code

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**Thuật toán Decision Tree**

**A screenshot of a computer code

Description automatically generated**

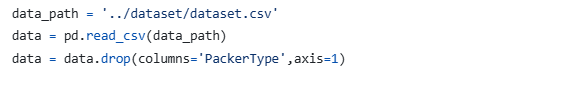
**A computer screen shot of a computer code

Description automatically generated**

1. **Thư viện xử lý dữ liệu**
   1. **Pandas**

* **Mục đích**: Xử lý và phân tích dữ liệu dưới dạng bảng.
* **Chức năng chính**: Đọc, ghi dữ liệu từ các file CSV, Excel; tiền xử lý dữ liệu, lọc và biến đổi dữ liệu.





* 1. **Numpy**
* **Mục đích:** Xử lý dữ liệu số học hiệu quả với mảng đa chiều (arrays).
* **Chức năng chính:** Tối ưu hóa các phép toán đại số và xử lý dữ liệu nhanh hơn Pandas.



A close-up of a computer code

Description automatically generated

* 1. **Scikit-learn**
* **Mục đích:** Xây dựng mô hình học máy cơ bản.
* **Chức năng chính:** Cung cấp các thuật toán học máy như Random Forest, SVM, Naive Bayes, KNN, Logistic Regression, v.v.



A close-up of a computer code

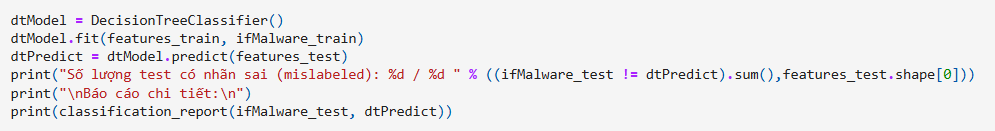
Description automatically generated

# **7. Đánh giá hiệu quả của các mô hình phân loại mã độc**

|  |  |  |
| --- | --- | --- |
| **Chỉ số** | **Mô tả** | **Công thức** |
| Accuracy (Độ chính xác) | Tỷ lệ dự đoán đúng trên tổng số mẫu. |  |
| Precision (Độ chính xác của dự đoán mã độc) | Tỷ lệ mẫu dự đoán là mã độc đúng trên tổng số mẫu dự đoán là mã độc. |  |
| Recall (Sensitivity/Độ nhạy) | Khả năng phát hiện chính xác mã độc trong dữ liệu. |  |
| Specificity (Độ đặc hiệu) | Khả năng nhận diện chính xác các mẫu là lành tính. |  |
| F1-Score | Trung bình điều hòa giữa Precision và Recall. |  |
| ROC-AUC (Receiver Operating Characteristic - Area Under Curve) | Đánh giá khả năng phân biệt giữa mã độc và lành tính. | Giá trị từ 0 đến 1. Giá trị càng gần 1, mô hình càng tốt. |

Bảng 3: Đánh giá mô hình

* + **Decision Tree**



**Kết quả mô hình:**

A screenshot of a computer screen

Description automatically generated

* + **Random Forest**

A close-up of a computer code

Description automatically generated

**Kết quả mô hình:**

A screenshot of a computer screen

Description automatically generated

* + **KNN (K Nearest Neighbors)**

A computer code with text

Description automatically generated with medium confidence

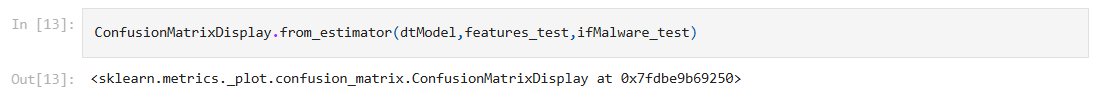
**Kết quả mô hình:**

A screenshot of a computer

Description automatically generated

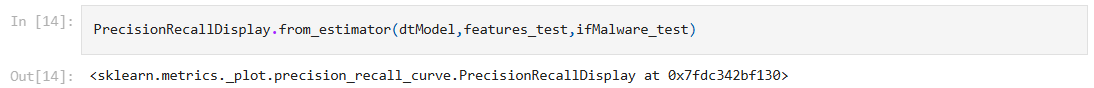
## **7.1. Đánh giá mô hình trực quan (Visualization)**

**- Decision tree**



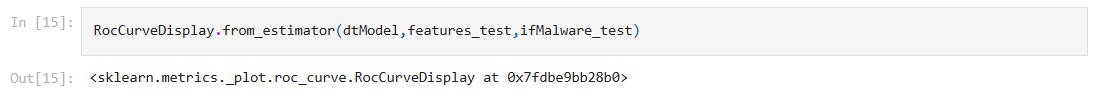
A chart of a color chart

Description automatically generated with medium confidence



A graph with numbers and lines

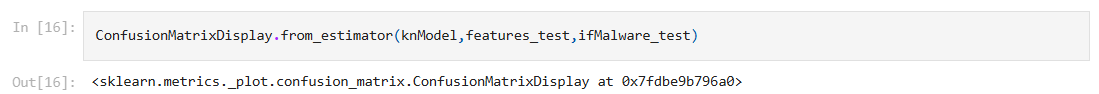
Description automatically generated



A graph with a line

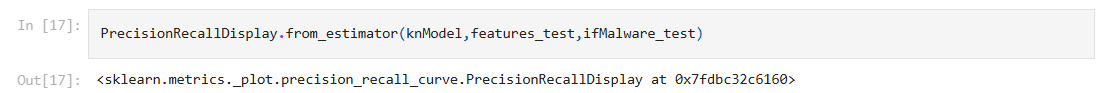
Description automatically generated

**- Random Forest**



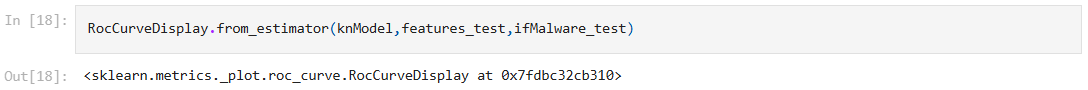
A chart of a number of colored squares

Description automatically generated with medium confidence



A graph with a line

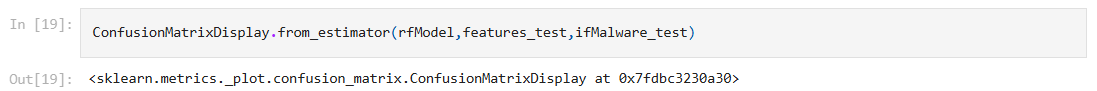
Description automatically generated with medium confidence



A graph with a line

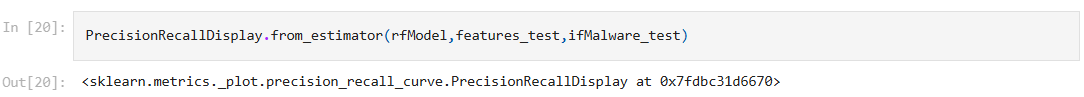
Description automatically generated

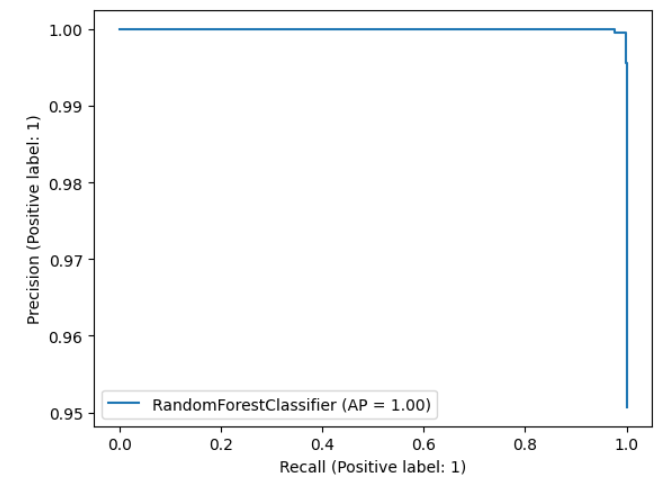
**- KNN (K Nearest Neighbors)**

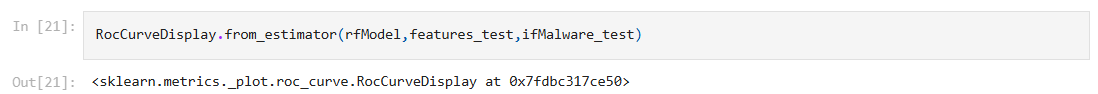


A chart of a color chart

Description automatically generated with medium confidence







A graph with a number of text

Description automatically generated with medium confidence

# **8.Kết luận và định hướng phát triển tương lai**

## **8.1. Kết Luận**

Việc sử dụng các thuật toán học máy (Machine Learning) trong phát hiện và phân loại mã độc là xu hướng tất yếu trong lĩnh vực an toàn thông tin. Thay vì dựa vào các phương pháp truyền thống như chữ ký (signature-based) hoặc heuristic-based, học máy cho phép hệ thống phát hiện các mối đe dọa mới mà chưa từng xuất hiện trước đó, thông qua việc phân tích các đặc trưng của dữ liệu mã độc.

Qua phân tích các mô hình phổ biến như KNN, Decision Tree, và Random Forest, ta nhận thấy rằng mỗi mô hình có ưu nhược điểm riêng. Trong đó, Random Forest là một trong những lựa chọn hiệu quả nhất để xử lý bài toán phân loại mã độc nhờ khả năng tổng quát hóa tốt và khả năng chống overfitting. Tuy nhiên, để đạt được kết quả tốt nhất, cần phải kết hợp các mô hình và tối ưu hóa dữ liệu đặc trưng.

Mặc dù các phương pháp học máy giúp tăng hiệu quả trong phát hiện mã độc, nhưng vẫn còn tồn tại nhiều thách thức như:

* Sự đa dạng và tinh vi của mã độc mới.
* Dữ liệu mất cân bằng giữa mã độc và phần mềm lành tính.
* Độ phức tạp và chi phí tính toán của các mô hình lớn.

Do đó, cần có các giải pháp phát triển và tối ưu hóa mô hình để tăng cường khả năng phát hiện sớm và giảm thiểu sai sót.

## **8.2. Định hướng phát triển tương lai**

**Tối ưu hóa trích xuất đặc trưng mã độc**

* Hiện nay, việc trích xuất đặc trưng từ dữ liệu tĩnh (static analysis) và dữ liệu động (dynamic analysis) còn nhiều hạn chế.
* Hướng phát triển là kết hợp các phương pháp hybrid analysis, trong đó sử dụng cả đặc trưng tĩnh và động để mô hình có cái nhìn toàn diện hơn về hành vi của mã độc.
* Tận dụng các công cụ như API call sequences, behavior graphs, và network traffic patterns để trích xuất các đặc trưng hành vi phức tạp của mã độc.

**Sử dụng các mô hình học sâu (Deep Learning)**

* Các thuật toán học sâu như Convolutional Neural Networks (CNN) và Recurrent Neural Networks (RNN) đang được nghiên cứu để phân loại mã độc.
* Deep Learning có khả năng tự động trích xuất đặc trưng từ dữ liệu mà không cần phụ thuộc hoàn toàn vào việc lựa chọn đặc trưng thủ công.
* Tuy nhiên, các mô hình học sâu đòi hỏi tài nguyên tính toán lớn và dữ liệu huấn luyện phong phú, đây là một thách thức cần được giải quyết.

# Tài liệu kham khảo

|  |  |
| --- | --- |
| [1] | "Akhtar, M. S., & Feng, T. (2022). Malware analysis and detection using machine learning algorithms. Symmetry, 14(11), 2304.". |
| [2] | "Dolesi, K., Steinbach, E., Velasquez, A., Whitaker, L., Baranov, M., & Atherton, L. (2024). A machine learning approach to ransomware detection using opcode features and k-nearest neighbors on windows. Authorea Preprints.". |
| [3] | "Abu Al-Haija, Q., Odeh, A., & Qattous, H. (2022). PDF malware detection based on optimizable decision trees. Electronics, 11(19), 3142.". |
| [4] | *Acharya, J., Chaudhary, A., Chhabria, A., & Jangale, S. (2021, May). Detecting malware, malicious URLs and virus using machine learning and signature matching. In 2021 2nd International Conference for Emerging Technology (INCET) (pp. 1-5). IEEE..* |