

**NIÊN LUẬN CƠ SỞ**

**NGÀNH AN TOÀN THÔNG TIN**

**ĐỀ TÀI**

**THỐNG KÊ VÀ ĐÁNH GIÁ CÁC MÔ HÌNH MÁY HỌC TRUYỀN THỐNG TRONG PHÁT HIỆN XÂM NHẬP MẠNG**

**Sinh viên: Ngô Đức Thắng Mã số sinh viên: B2203737 Khóa: 48**



**NIÊN LUẬN CƠ SỞ**

**NGÀNH AN TOÀN THÔNG TIN**

# KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

**ĐỀ TÀI**

**THỐNG KÊ VÀ ĐÁNH GIÁ CÁC MÔ HÌNH MÁY HỌC TRUYỀN THỐNG TRONG PHÁT HIỆN XÂM NHẬP MẠNG**

**Giảng viên hướng dẫn:**

**TS. Nguyễn Hữu Vân Long**

**Sinh viên thực hiện: Ngô Đức Thắng**

**MSSV: B2203737**

**Khóa: 48**

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

🙠 🕮 🙢

………………………………………………………………………………………..

……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. ……………………………………………………………………………………….. …………………………………………………………………………………………

# MỤC LỤC

[TÓM LƯỢC 1](#_heading=h.lvqx479wiu4s)

[ABSTRACT 2](#_heading=h.1z1q7qn51qba)

[CHƯƠNG 1: GIỚI THIỆU 3](#_heading=h.q1o9kubrvh45)

[1.1 Lý do chọn đề tài 3](#_heading=h.roxbn6wigbks)

[1.2 Mục tiêu đề tài 3](#_heading=h.754747exjwz8)

[1.3 Đối tượng nghiên cứu 4](#_heading=h.o24hin1lm7us)

[1.4 Phạm vi nghiên cứu 4](#_heading=h.ljhzscw4uiol)

[1.5 Phương pháp nghiên cứu 4](#_heading=h.muw11mvledc3)

[1.6 Ý nghĩa khoa học và thực tiễn của đề tài 5](#_heading=h.4g0qiu9tq71h)

[1.7 Bố cục luận văn 6](#_heading=h.v49djn31ksuz)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 7](#_heading=h.4kxnetwgk2yx)

[2.1 Hệ thống phát hiện xâm nhập mạng, máy học và vai trò của máy học trong hệ thống phát hiện xâm nhập mạng 7](#_heading=h.i2mulnfwfsnw)

[2.1.1 Hệ thông phát hiện xâm nhập mạng (Network-based IDS) 7](#_heading=h.svdycwcu7j93)

[2.1.2 Máy học (Machine learning) và vai trò trong hệ thống phát hiện xâm nhập mạng 8](#_heading=h.8mu930xyghkc)

[2.2 Tập dữ liệu CSE-CIC-IDS2018 8](#_heading=h.osz5ljrljuda)

[2.3 Các phương pháp máy học truyền thống 9](#_heading=h.bru6b3x1o69o)

[2.3.1 K-Nearest Neighbors (KNN) 9](#_heading=h.jxlokjgdpnuu)

[2.3.2 Gaussian Naive Bayes (GaussianNB) 10](#_heading=h.f4fbfivtl2aw)

[2.3.3 Decision Tree (DT) 10](#_heading=h.5h0qr2r14psd)

[2.3.4 Random Forest (RF) 11](#_heading=h.bme595k8xahz)

[2.3.5 Extreme Gradient Boost (XGBoost) 12](#_heading=h.vryp16ggqq9k)

[2.4 Các chỉ số đánh giá mô hình 12](#_heading=h.emljtl9q2511)

[2.5 Các kỹ thuật tiền xử lý dữ liệu 13](#_heading=h.p2dcuzbc82qz)

[2.6. Kỹ thuật cross-validation 15](#_heading=h.4zjnimb43w0s)

[CHƯƠNG 3: PHƯƠNG PHÁP THỰC HIỆN 16](#_heading=h.emmj7e4ph2u6)

[3.1 Tổng quan bài toán 16](#_heading=h.ctzwfvrfo0y2)

[3.2 Tiền xử lý dữ liệu 16](#_heading=h.bk01xjvnd3nm)

[3.2.1 Chuyển đổi và gộp dữ liệu 16](#_heading=h.nolz64hwrobs)

[3.2.2 Trực quan hóa dữ liệu 17](#_heading=h.7wdwnkh0lzx3)

[3.2.3 Làm sạch dữ liệu 18](#_heading=h.bl8abtxks8hc)

[3.2.4 Mã hóa nhãn 19](#_heading=h.m9pccua2l1rl)

[3.2.5 Chuẩn hóa dữ liệu 20](#_heading=h.w6c4qle5shst)

[3.2.6 Lựa chọn đặc trưng 20](#_heading=h.mk0270qeqx58)

[3.2.7 Xử lí mất cân bằng 21](#_heading=h.ya9i7ajr4zmi)

[3.2.8 Chia dữ liệu 23](#_heading=h.j27tvhd776lj)

[3.3 Huấn luyện và đánh giá các mô hình 23](#_heading=h.3vu5pdo6dj9o)

[CHƯƠNG 4: KẾT QUẢ THỰC NGHIỆM 26](#_heading=h.trt7aciybh13)

[4.1 Thu thập kết quả 26](#_heading=h.v7z9kecezzq)

[4.2 Môi trường thực nghiệm 26](#_heading=h.nahcwrzaumqb)

[4.3 Kết quả thực nghiệm 27](#_heading=h.r9m3l8xo7jl5)

[CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 41](#_heading=h.dk11y5g6ksop)

[5.1 Kết luận 41](#_heading=h.qexxse2xarbo)

[5.2 Hạn chế và hướng phát triển 42](#_heading=h.8qvj9wd56spi)

[5.2.1 Hạn chế 42](#_heading=h.az2z83us8koq)

[5.2.2 Hướng phát triển 43](#_heading=h.i34u5rf5m6l8)

[TÀI LIỆU THAM KHẢO 43](#_heading=h.9cbtdvn1vbd4)

**MỤC LỤC HÌNH ẢNH**

[Hình 1. Hình minh họa kiến trúc Hệ thống phát hiện xâm nhập mạng 8](#_Toc205050936)

[Hình 2. Hình minh họa thuật toán KNN 10](#_Toc205050937)

[Hình 3. Hình minh hoạt thuật toán GaussianNB 11](#_Toc205050938)

[Hình 4. Hình minh họa thuật toán Decision Tree 12](#_Toc205050939)

[Hình 5. Hình minh họa thuật toán Random Forest 13](#_Toc205050940)

[Hình 6. Hình minh họa thuật toán XGBoost 13](#_Toc205050941)

[Hình 7. Hình minh họa AUC ROC 15](#_Toc205050942)

[Hình 8. Hình minh họa kĩ thuật Cross-Validation 18](#_Toc205050943)

[Hình 9. Hình minh họa bài toán lớn của đề tài 19](#_Toc205050944)

[Hình 10. Các file PARQUET nhận về sau khi tải từ Kaggle 19](#_Toc205050945)

[Hình 11. File CSV lớn sau khi chuyển đổi và gộp 10 file PARQUET lại 20](#_Toc205050946)

[Hình 12. Các thông tin sau khi được in ra 20](#_Toc205050947)

[Hình 13. Phân phối các nhãn theo loại 7 lưu lượng 21](#_Toc205050948)

[Hình 14. Số đặc trưng có giá trị 0 nhiều hơn 98% 21](#_Toc205050949)

[Hình 15. Kết quả số dòng bị thiếu, trùng lặp hoặc chứa giá trị vô cùng 22](#_Toc205050950)

[Hình 16. Phân bố nhãn trong tập dữ liệu sau khi mã hóa nhãn 23](#_Toc205050951)

[Hình 17. Top 30 đặc trưng quan trọng nhất 24](#_Toc205050952)

[Hình 18. Phân bố nhãn sau SMOTE 25](#_Toc205050953)

[Hình 19. Phân bố nhãn sau Random Under Sampler 25](#_Toc205050954)

[Hình 20. Các file dữ liệu sẽ được sử dụng để huấn luyện và đánh giá mô hình 26](#_Toc205050955)

[Hình 21. Các tổ tham số của GaussianNB 27](#_Toc205050956)

[Hình 22. Các tổ hợp tham số của Decision Tree 28](#_Toc205050957)

[Hình 23. Các tổ hợp tham số của Random Forest 28](#_Toc205050958)

[Hình 25. Các thông số kĩ thuật của phần cứng sử dụng trong đề tài 29](#_Toc205050959)

[Hình 26. Bảng thống kê kết quả các mô hình trên tập chưa cân bằng 30](#_Toc205050960)

[Hình 27. Biểu đồ so sánh hiệu suất KNN trên tập CV và Test 30](#_Toc205050961)

[Hình 28. Biểu đồ so sánh hiệu suất GaussianBayes trên tập CV và Test 31](#_Toc205050962)

[Hình 29. Biểu đồ so sánh hiệu suất DecisionTree trên tập CV và Test 31](#_Toc205050963)

[Hình 30. Biểu đồ so sánh hiệu suất Random Forest trên tập CV và Test 31](#_Toc205050964)

[Hình 31. Biểu đồ so sánh hiệu suất XGBoost trên tập CV và Test 32](#_Toc205050965)

[Hình 32. Ma trận nhầm lẫn của mô hình KNN 32](#_Toc205050966)

[Hình 33. Ma trận nhầm lẫn của mô hình GaussianNB 32](#_Toc205050967)

[Hình 34. Ma trận nhầm lẫn của mô hình DecisionTree 33](#_Toc205050968)

[Hình 35. Ma trận nhầm lẫn của mô hình Random Forest 33](#_Toc205050969)

[Hình 36. Ma trận nhầm lẫn của mô hình XGBoost 33](#_Toc205050970)

[Hình 37. Bảng thống kê kết quả các mô hình trên tập đã SMOTE 34](#_Toc205050971)

[Hình 38. Biểu đồ so sánh hiệu suất KNN trên tập CV và Test (SMOTE) 35](#_Toc205050972)

[Hình 39. Biểu đồ so sánh hiệu suất GaussianNB trên tập CV và Test (SMOTE) 35](#_Toc205050973)

[Hình 40. Biểu đồ so sánh hiệu suất DecisionTree trên tập CV và Test (SMOTE) 35](#_Toc205050974)

[Hình 41. Biểu đồ so sánh hiệu suất Random Forest trên tập CV và Test (SMOTE) 36](#_Toc205050975)

[Hình 42. Biểu đồ so sánh hiệu suất XGBoost trên tập CV và Test (SMOTE) 36](#_Toc205050976)

[Hình 43. Ma trận nhầm lẫn KNN(SMOTE) 36](#_Toc205050977)

[Hình 44. Ma trận nhầm lẫn GaussianNB (SMOTE) 37](#_Toc205050978)

[Hình 45. Ma trận nhầm lẫn DecisionTree (SMOTE) 37](#_Toc205050979)

[Hình 46. Ma trận nhầm lẫn Random Forest (SMOTE) 37](#_Toc205050980)

[Hình 47. Ma trận nhầm lẫn XGBoost (SMOTE) 38](#_Toc205050981)

[Hình 48. Bảng thống kê kết quả các mô hình trên tập đã RandomUnderSampler 39](#_Toc205050982)

[Hình 49. Biểu đồ so sánh hiệu suất KNN trên tập CV và Test (RandomUnderSampler) 39](#_Toc205050983)

[Hình 50. Biểu đồ so sánh hiệu suất GaussianNB trên tập CV và Test (RandomUnderSampler) 39](#_Toc205050984)

[Hình 51. Biểu đồ so sánh hiệu suất DecisionTree trên tập CV và Test (RandomUnderSampler) 40](#_Toc205050985)

[Hình 52. Biểu đồ so sánh hiệu suất RandomForest trên tập CV và Test (RandomUnderSampler) 40](#_Toc205050986)

[Hình 53. Biểu đồ so sánh hiệu suất XGBoost trên tập CV và Test (RandomUnderSampler) 40](#_Toc205050987)

[Hình 54. Ma trận nhầm lẫn KNN (RandomUnderSampler) 41](#_Toc205050988)

[Hình 55. Ma trận nhầm lẫn GaussianNB (RandomUnderSampler) 41](#_Toc205050989)

[Hình 56. Ma trận nhầm lẫn DecisionTree (RandomUnderSampler) 41](#_Toc205050990)

[Hình 57. Ma trận nhầm lẫn RandomForest (RandomUnderSampler) 42](#_Toc205050991)

[Hình 58. Ma trận nhầm lẫn XGBoost (RandomUnderSampler) 42](#_Toc205050992)

**DANH MỤC TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| Từ viết tắt | Viết đầy đủ |
| IDS | Intrusion Detection System |
| NIDS | Netword-based Intrusion Detection System |
| CV | Cross-Validation |
| AUC | Area Under the Curve |
| KNN | K-Nearest Neighbors |
| GNB/GaussianNB | Gaussian Naive Bayes |
| DT | Decision Tree |
| RF | Random Forest |
| XG/XGBoost | Extreme Gradient Boosting |
| SMOTE | Synthetic Minority Over-sampling Technique |
| FN | False Negative |
| FP | False Positive |
| TP | True Positive |
| TN | True Negative |
| ADASYN | Adaptive Synthetic Sampling |
| LSTM | Long Short-term Memory |
| CNN | Convolutional Neural Network |

**LỜI CẢM ƠN**

*Lời đầu tiên, em muốn nói rằng việc thực hiện đề tài “Thống kê và đánh giá các mô hình máy học truyền thống trong phát hiện xâm nhập mạng” là một thách thức đối với em. Đề tài không chỉ yêu cầu kiến thức, kỹ thuật chuyên môn mà còn cần sự kiên nhẫn, tỉ mỉ và kỹ năng tổng hợp, đánh giá kết quả. Trong suốt quá trình thực hiện đề tài này, em đã gặp không ít khó khăn, nhưng đây cũng là cơ hội để học hỏi và rèn luyện bản thân.*

*Em xin chân thành cảm ơn Trường Đại học Cần Thơ, đặc biệt là Trường Công nghệ thông tin và truyền thông đã tạo điều kiện học tập và môi trường tốt để em có thể thực hiện và hoàn thành đề tài này.*

*Em cũng xin gửi lời cảm ơn sâu sắc đến Tiến sĩ Nguyễn Hữu Vân Long, thầy đã tận tình hướng dẫn, góp ý rất nhiều và định hướng trong suốt quá trình em thực hiện đề tài. Nhờ sự giúp đỡ và chỉ dẫn của thầy, em đã hiểu rõ hơn về các kiến thức chuyên ngành và cách tiếp cận vấn đề một cách khoa học hơn.*

*Bên cạnh đó, em xin cảm ơn gia đình đã luôn là chỗ dựa tinh thần vững chắc, và cảm ơn các bạn bè đã hỗ trợ, chia sẻ tài liệu, góp ý cũng như động viên em trong suốt thời gian thực hiện đề tài này.*

*Mặc dù còn nhiều thiếu sót nhưng em đã cố gắng hết sức để hoàn thành đề tài. Em hy vọng sẽ nhận được những góp ý để hoàn thiện hơn trong tương lai.*

*Cuối lời em xin chúc thầy sức khỏe, thành công và hạnh phúc!*

Ngô Đức Thắng

Lớp An toàn thông tin K48

# TÓM LƯỢC

## 

Trong bối cảnh các mối đe dọa an ninh mạng ngày càng gia tăng, hệ thống phát hiện xâm nhập mạng (Network-based Intrusion Detection System – NIDS) đóng vai trò then chốt trong việc phát hiện và ngăn chặn các hành vi tấn công. Việc lựa chọn mô hình học máy phù hợp cho NIDS là một thách thức quan trọng nhằm cân bằng giữa hiệu quả phát hiện và chi phí tính toán.

Đề tài tập trung vào việc thống kê và so sánh hiệu năng của các mô hình học máy truyền thống gồm: Decision Tree (DT), Random Forest (RF), XGBoost, K-Nearest Neighbors (KNN) và Gaussian Naive Bayes (GNB) trong bài toán phát hiện xâm nhập mạng, sử dụng bộ dữ liệu CSE-CIC-IDS2018. Dữ liệu được tiền xử lý qua các bước làm sạch, chuẩn hóa, lựa chọn đặc trưng, và xử lý mất cân bằng bằng cả oversampling và undersampling. Các mô hình được đánh giá trên cả hai tập dữ liệu: trước và sau cân bằng.

Các chỉ số đánh giá bao gồm: Accuracy, Precision, Recall, F1-score, AUC, thời gian thực thi và kích thước mô hình. Để đảm bảo tính khách quan và ổn định, kỹ thuật 5-fold cross-validation được áp dụng trong quá trình thực nghiệm. Một số tổ hợp tham số cơ bản được áp dụng cho từng mô hình dựa trên tài liệu tham khảo từ các nghiên cứu trước.

Kết quả ban đầu cho thấy mô hình XGBoost đạt hiệu năng tốt nhất về cả độ chính xác và khả năng tổng quát. Random Forest và Decision Tree cũng cho kết quả khá cao nhưng hiệu suất chưa bằng XGBoost. Quá trình thực nghiệm được thực hiện bằng Python, sử dụng thư viện scikit-learn, trong môi trường phát triển Visual Studio Code.

# ABSTRACT

In the context of increasingly sophisticated cybersecurity threats, Network-based Intrusion Detection Systems (NIDS) play a crucial role in detecting and preventing malicious activities within networks. Selecting an appropriate machine learning model for IDS remains a critical challenge in balancing detection performance and computational cost.

This study focuses on the statistical comparison and evaluation of the performance of traditional machine learning models, including Decision Tree (DT), Random Forest (RF), XGBoost, K-Nearest Neighbors (KNN), and Gaussian Naive Bayes (GNB), in the task of network intrusion detection using the CSE-CIC-IDS2018 dataset. The dataset is preprocessed through cleaning, normalization, feature selection, and handling class imbalance using both oversampling and undersampling techniques. All models are evaluated on both the original imbalanced dataset and the balanced dataset.

Evaluation metrics include Accuracy, Precision, Recall, F1-score, AUC, execution time, and model size. To ensure objective and reliable results, 5-fold cross-validation is applied throughout the experiments. A limited number of basic hyperparameter configurations for each model are used, selected based on settings reported in related studies.

Preliminary results indicate that XGBoost achieves the best overall performance in terms of accuracy and generalization ability. Random Forest and Decision Tree also perform well but do not surpass XGBoost. All experiments are conducted in Python, using the scikit-learn library within the Visual Studio Code development environment.

# CHƯƠNG 1: GIỚI THIỆU

## 1.1 Lý do chọn đề tài

Trong thời đại công nghệ thông tin phát triển mạnh mẽ, các hệ thống mạng máy tính ngày càng trở nên phổ biến và đóng vai trò quan trọng trong hầu hết mọi lĩnh vực của đời sống xã hội. Tuy nhiên, đi cùng với sự phát triển đó là sự gia tăng không ngừng của các cuộc tấn công mạng với mức độ tinh vi và quy mô ngày càng lớn. Điều này đặt ra yêu cầu cấp thiết cho việc xây dựng các hệ thống phát hiện xâm nhập mạng (Network-based Intrusion Detection Systems – NIDS) nhằm đảm bảo an toàn và ổn định cho hệ thống thông tin.

Trong lĩnh vực này, các mô hình học máy đang được sử dụng ngày càng nhiều để hỗ trợ phát hiện xâm nhập hiệu quả hơn. Tuy nhiên, giữa rất nhiều mô hình khác nhau, việc lựa chọn mô hình phù hợp không phải là điều dễ dàng, bởi mỗi mô hình có ưu và nhược điểm riêng về độ chính xác, tốc độ thực thi và khả năng tổng quát.

Trên cơ sở đó, em đã quyết định thực hiện đề tài “Thống kê và đánh giá các mô hình máy học truyền thống trong phát hiện xâm nhập mạng” lựa chọn hướng tiếp cận là thống kê và đánh giá một số mô hình học máy truyền thống phổ biến, nhằm tìm hiểu rõ hơn về hiệu năng của từng mô hình trong bối cảnh cụ thể là bài toán phát hiện xâm nhập mạng sử dụng bộ dữ liệu CSE-CIC-IDS2018.

## 1.2 Mục tiêu đề tài

Đề tài hướng đến việc thống kê và đánh giá hiệu năng của các mô hình học máy truyền thống trong phát hiện xâm nhập mạng, bao gồm: Decision Tree, Random Forest, XGBoost, K-Nearest Neighbors và Gaussian Naive Bayes.

Tập dữ liệu sử dụng là bộ CSE-CIC-IDS2018, được tiền xử lý với các bước: làm sạch, chuẩn hóa, lựa chọn đặc trưng, và xử lý mất cân bằng bằng kỹ thuật oversampling và undersampling. Tập dữ liệu được chia theo tỷ lệ 80% huấn luyện và 20% kiểm tra, đồng thời áp dụng kỹ thuật 5-fold cross-validation để đảm bảo độ tin cậy trong quá trình đánh giá.

Các mô hình được thử nghiệm với một vài bộ tham số được tham khảo từ các nghiên cứu trước. Các chỉ số được sử dụng để so sánh hiệu năng bao gồm: Accuracy, Precision, Recall, F1-score, AUC, thời gian thực thi (huấn luyện và đánh giá) và kích thước mô hình.

Mục tiêu cuối cùng của đề tài là không chỉ thực nghiệm và áp dụng kiến thức học máy đã học, mà còn tìm ra mô hình phù hợp hơn trong việc phát hiện xâm nhập mạng, từ đó góp phần cung cấp cơ sở tham khảo cho các nghiên cứu hoặc ứng dụng tiếp theo trong lĩnh vực an toàn thông tin.

## 1.3 Đối tượng nghiên cứu

Đối tượng nghiên cứu của đề tài là các mô hình học máy truyền thống phổ biến như Decision Tree, Random Forest, XGBoost, K-Nearest Neighbors và Gaussian Naive Bayes, được ứng dụng rộng rãi trong hệ thống phát hiện xâm nhập mạng (Intrusion Detection System – IDS).

## 1.4 Phạm vi nghiên cứu

Phạm vi nghiên cứu của đề tài là tìm hiểu và thống kê hiệu năng của các mô hình học máy truyền thống đang được sử dụng phổ biến trong bài toán phát hiện xâm nhập mạng. Từ đó, đề tài tiến hành thực nghiệm trên môi trường offline và đánh giá sáu mô hình cụ thể trên bộ dữ liệu CSE-CIC-IDS2018 đã qua tiền xử lý.

## 1.5 Phương pháp nghiên cứu

Phương pháp nghiên cứu tài liệu:

- Đề tài tham khảo các tài liệu, giáo trình và bài báo khoa học có liên quan đến lĩnh vực học máy và phát hiện xâm nhập mạng. Ngoài ra, em cũng tìm hiểu thêm các nghiên cứu trước đó sử dụng cùng bộ dữ liệu CSE-CIC-IDS2018 để hiểu rõ hơn cách lựa chọn đặc trưng, xử lý mất cân bằng và tinh chỉnh mô hình.

Phương pháp thu thập dữ liệu:

- Dữ liệu được sử dụng trong đề tài là tập dữ liệu CSE-CIC-IDS2018 – một bộ dữ liệu công khai mô phỏng nhiều dạng tấn công mạng khác nhau. Dữ liệu được tải từ nguồn chính thức trên Kaggle.com và được xử lý lại để phục vụ cho quá trình huấn luyện mô hình.

Phương pháp nghiên cứ thực nghiệm:

- Tiến hành tiền xử lý dữ liệu bao gồm: trực quan hóa, làm sạch dữ liệu, chuẩn hóa bằng StandardScaler, lựa chọn đặc trưng bằng mô hình Random Forest, và xử lý mất cân bằng bằng hai phương pháp: SMOTE (tăng mẫu thiểu số) và RandomUnderSampler (giảm mẫu đa số).

- Các mô hình học máy truyền thống được sử dụng trong đề tài gồm: Decision Tree, Random Forest, XGBoost, K-Nearest Neighbors và Gaussian Naive Bayes. Các mô hình này được xây dựng và huấn luyện trong môi trường Visual Studio Code.

- Để đánh giá hiệu năng mô hình, dữ liệu được chia theo tỷ lệ 80/20 cho train và test. Đồng thời, kỹ thuật 5-fold Stratified Cross-Validation cũng được áp dụng để đảm bảo tính khách quan và ổn định của kết quả.

- Các chỉ số đánh giá bao gồm: Accuracy, Precision, Recall, F1-score, AUC, thời gian thực thi và kích thước mô hình. Các kết quả trung gian và cuối cùng đều được lưu lại bằng Excel, đồng thời mô hình huấn luyện cũng được xuất dưới dạng file JOBLIB cho các mục đích sâu xa hơn như làm ứng dụng.

- Cuối cùng, kết quả được trực quan hóa bằng biểu đồ và ma trận nhầm lẫn để giúp dễ dàng so sánh và phân tích hiệu năng giữa các mô hình.

## 1.6 Ý nghĩa khoa học và thực tiễn của đề tài

Ý nghĩa khoa học: Đề tài góp phần làm rõ hiệu quả của các mô hình học máy truyền thống trong bài toán phát hiện xâm nhập mạng, thông qua việc đánh giá và so sánh dựa trên nhiều tiêu chí khác nhau như độ chính xác, tốc độ, khả năng tổng quát và kích thước mô hình để có thể làm cơ sở tham khảo cho các nghiên cứu tiếp theo trong lĩnh vực phát hiện xâm nhập mạng bằng học máy.

Ý nghĩa thực tiễn: Ngày nay tấn công mạng ngày càng tinh vi và phổ biến nên lựa chọn được một mô hình học máy phù hợp để phát hiện xâm nhập là rất quan trọng nhằm đảm bảo an toàn cho hệ thống thông tin. Kết quả từ đề tài giúp định hướng cho việc lựa chọn mô hình phù hợp, từ đó hỗ trợ tốt hơn trong việc triển khai các hệ thống NIDS trong thực tế.

## 1.7 Bố cục luận văn

Chương 1: Giới thiệu

Chương 2: Cơ sở lý thuyết

Chương 3: Phương pháp thực hiện

Chương 4: Kết quả thực nghiệm

Chương 5: Kết luận và hướng phát triển

Tổng kết chương 1: Trong chương này, đề tài đã nêu rõ mục tiêu , lý do chọn đề tài, đối tượng nghiên cứu, phạm vi nghiên cứu, các phương pháp nghiên cứu đề tài và ý nghĩa khoa học, thực tiễn của đề tài lựa chọn. Trong tiếp theo, đề tài sẽ trình bày các cơ sở lý thuyết liên quan đến các phương pháp nghiên cứu.

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

Trong chương này, đề tài sẽ trình bày những cơ sở lý thuyết liên quan đến việc áp dụng học máy trong phát hiện xâm nhập mạng. Nội dung bao gồm:

- Tổng quan về Hệ thống phát hiện xâm nhập mạng (NIDS), máy học (Machine learning) và vai trò của máy học trong hệ thống phát hiện xâm nhập (NIDS).

- Giới thiệu về tập dữ liệu CSE-CIC-IDS2018 và các mô hình học máy truyền thống Decision Tree, Random Forest, XGBoost, K-Nearest Neighbors và Gaussian Naive Bayes.

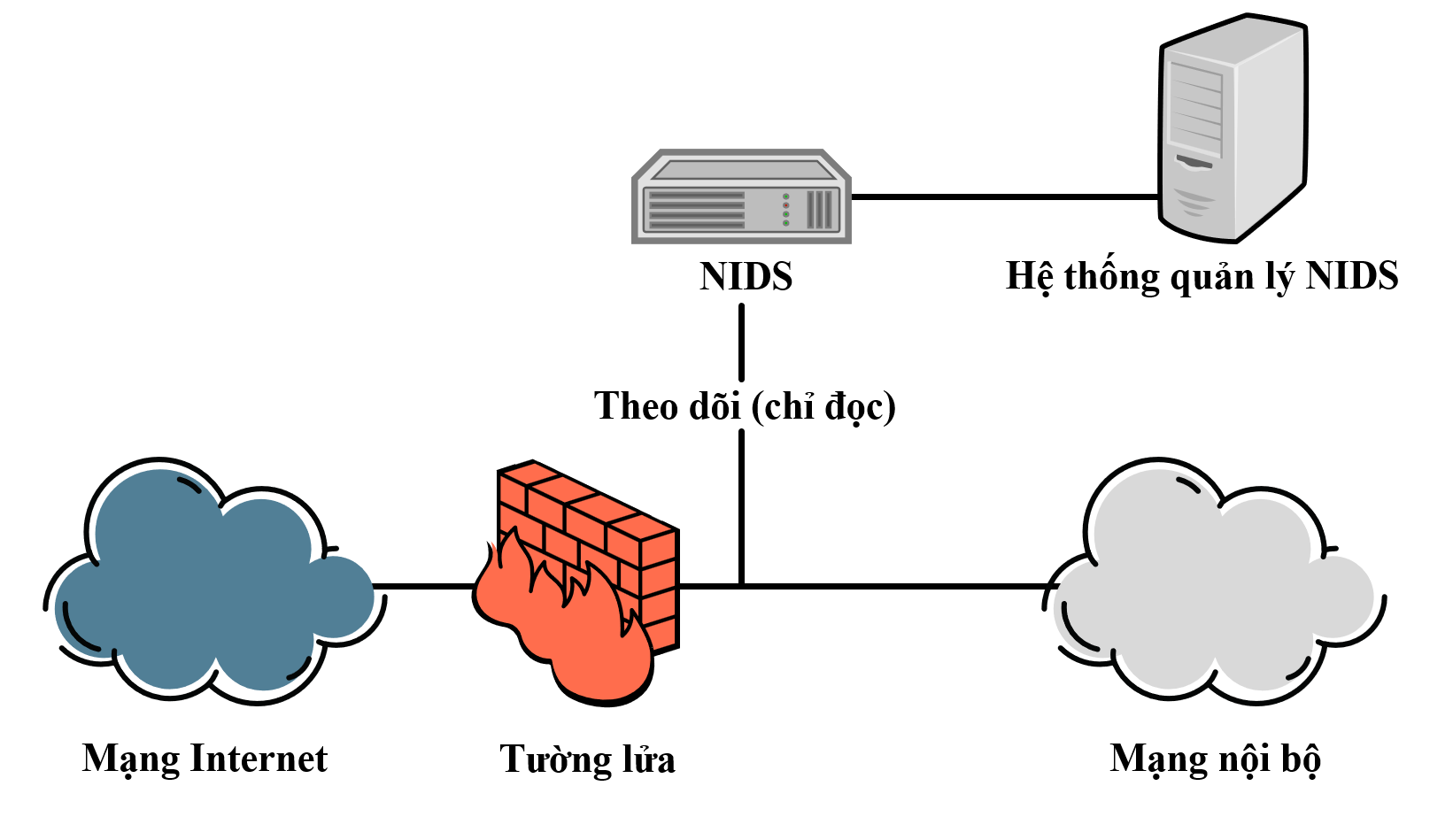
- Các chỉ số đánh giá mô hình như Accuracy, Precision, Recall, F1-score, AUC, thời gian thực thi và kích thước mô hình.

- Các kỹ thuật tiền xử lý dữ liệu, bao gồm chuẩn hóa dữ liệu, lựa chọn đặc trưng, và xử lý mất cân bằg dữ liệu (SMOTE, RandomUnderSampler).

## 2.1 Hệ thống phát hiện xâm nhập mạng, máy học và vai trò của máy học trong hệ thống phát hiện xâm nhập mạng

### 2.1.1 Hệ thông phát hiện xâm nhập mạng (Network-based IDS)

Hệ thống phát hiện xâm nhập mạng, thường gọi là NIDS (Network Intrusion Detection System), là một công cụ bảo mật có chức năng giám sát lưu lượng mạng để phát hiện các hành vi đáng ngờ, vi phạm chính sách hoặc các cuộc tấn công mạng. NIDS hoạt động bằng cách phân tích các gói tin đi qua mạng tại những vị trí chiến lược quan trọng như router hoặc firewall rồi báo cáo về trung tâm quản lý NDIS hoặc trực tiếp đến người quản trị hệ thống.



Hình 1. Hình minh họa kiến trúc Hệ thống phát hiện xâm nhập mạng

### 2.1.2 Máy học (Machine learning) và vai trò trong hệ thống phát hiện xâm nhập mạng

Máy học là một lĩnh vực thuộc trí tuệ nhân tạo (AI), cho phép hệ thống học từ dữ liệu đầu vào mà không cần phải được lập trình rõ ràng từng bước. Nói một cách đơn giản, máy học giúp máy tính tự rút ra kinh nghiệm từ dữ liệu, để từ đó đưa ra các quyết định hoặc dự đoán tốt hơn.

Máy học được xem là một giải pháp tiềm năng để nâng cao hiệu quả phát hiện xâm nhập nhờ khả năng tự học và thích nghi với dữ liệu mới trong bối cảnh lưu lượng mạng ngày nay càng gia tăng mạnh mẽ về lưu lượng và tính phức tạp. Trong hệ thống NIDS, các mô hình học máy có thể được huấn luyện từ dữ liệu mạng đã được gán nhãn, từ đó nhận biết được các hành vi bất thường mà không cần phải định nghĩa sẵn từng loại tấn công. Điều này giúp hệ thống phát hiện được cả các mẫu tấn công đã biết và chưa biết với độ chính xác cao hơn.

## 2.2 Tập dữ liệu CSE-CIC-IDS2018

Tập dữ liệu CSE-CIC-IDS2018 là một tập dữ liệu lưu lượng mạng do Viện An ninh mạng Canada (CIC) và Cơ quan An ninh Truyền thông (CSE) xây dựng. Đây là một bộ dữ liệu khá toàn diện, mô phỏng sát với thực tế khi mạng bị tấn công. Đây là tập dữ liệu có độ uy tín rất cao, được dùng rộng rãi trong các bài nghiên cứu về hệ thống phát hiện xâm nhập mạng.

Dữ liệu được tạo ra bằng cách ghi lại lưu lượng mạng thật, sau đó dùng công cụ CICFlowMeter để trích xuất các đặc trưng (feature) ở cấp độ luồng từ các tệp pcap. Vì thế, dữ liệu vừa phong phú vừa phản ánh được hành vi thật sự của người dùng và hacker trên môi trường mạng, rất phù hợp để nghiên cứu các hệ thống phát hiện xâm nhập (NIDS).

## 2.3 Các phương pháp máy học truyền thống

### 2.3.1 K-Nearest Neighbors (KNN)

Thuật toán K-Nearest Neighbor (KNN) là một thuật toán học có giám sát (supervised learning). Tuy nhiên, không giống như nhiều thuật toán học có giám sát khác, KNN không cần giai đoạn huấn luyện riêng biệt. nó hoạt động dựa trên dữ liệu từ một tập mẫu ban đầu đã biết nhãn. Khi cần phân loại một điểm dữ liệu mới, thuật toán sẽ tìm K điểm gần nhất trong tập dữ liệu mẫu để quyết định nhãn mà điểm mới này nên được gán vào, cụ thể:

* Khoảng cách từ điểm dữ liệu mới đến các điểm trong tập mẫu sẽ được tính để xác định các điểm gần nhất. Các công thức phổ biến dùng để tính khoảng cách gồm có: Euclidean, Manhattan, và Minkowski với công thức tính sau:
* Khoảng cách Euclidean:

Euclidean =

* Khoảng cách Manhattan:

Manhattan =

* Khoảng cách Minkowski:

Minkowski

Trong đó:

là các giá trị của thuộc tính thứ I của 2 điểm dữ liệu.

N là số thuộc tính (số chiều).

K là tham số điều chỉnh mức độ tổng quát trong công thức Minkowski.

(k = 1 là Manhattan, k = 2 là Euclidean).

* Ưu điểm: dễ hiểu, dễ cài đặt, linh hoạt với các loại dữ liệu khác nhau, hiệu quả cao khi dữ liệu có ranh giới phân lớp rõ ràng.
* Nhược điểm: tốc độ rất chậm khi dữ liệu lớn, tốn bộ nhớ và nhạy cảm với nhiễu, hiệu năng cũng có thể bị giảm nếu tập dữ liệu lớn.

A diagram of a diagram of a class

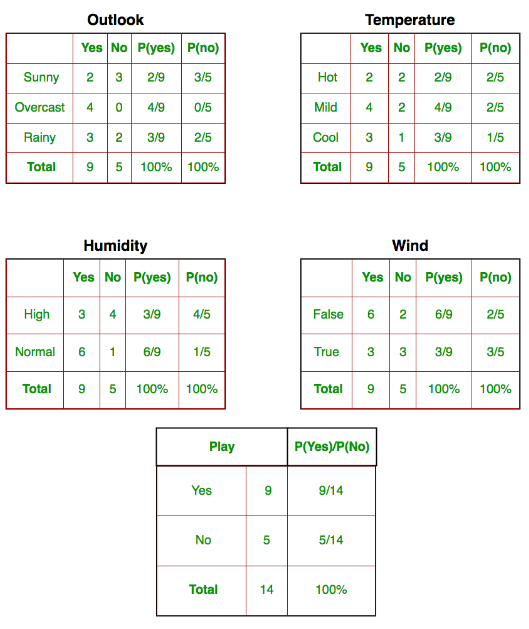
Description automatically generated

Hình 2. Hình minh họa thuật toán KNN

### 2.3.2 Gaussian Naive Bayes (GaussianNB)

Naive Bayes là một kỹ thuật học máy dựa trên xác suất, thường được sử dụng cho các bài toán phân loại. Thuật toán này dựa trên định lý Bayes, cho phép ước lượng khả năng xảy ra của một sự kiện dựa trên kiến thức có sẵn về các yếu tố liên quan. Trong trường hợp của Naive Bayes, ta giả định rằng các đặc trưng (features) trong tập dữ liệu là độc lập có điều kiện.

Mặc dù giả định này khá đơn giản (vì thế mới gọi là "Naive" – ngây thơ), nhưng Naive Bayes lại hoạt động rất hiệu quả trong thực tế, đặc biệt là trong các ứng dụng như phân loại văn bản hoặc lọc thư rác. Thuật toán này có ưu điểm là dễ cài đặt, tốc độ xử lý nhanh và phù hợp với các tập dữ liệu lớn, nên được sử dụng rộng rãi trong nhiều bài toán phân loại khác nhau.



Hình 3. Hình minh hoạt thuật toán GaussianNB

### 2.3.3 Decision Tree (DT)

Cây quyết định (Decision Tree – DT) là một thuật toán học máy cơ bản, được sử dụng cho cả bài toán phân loại và hồi quy. Thuật toán này chia nhỏ dữ liệu thành các tập con dựa trên giá trị của các đặc trưng đầu vào. Ở mỗi bước, nó chọn đặc trưng và ngưỡng chia sao cho việc phân loại đạt hiệu quả cao nhất. Mỗi nút trong của cây đại diện cho một điều kiện chia, còn mỗi nút lá là kết quả đầu ra – có thể là nhãn phân loại hoặc giá trị số. Việc chia tiếp tục cho đến khi đạt điều kiện dừng như độ sâu tối đa hoặc số mẫu tối thiểu tại nút lá. Các đặc điểm khác của Decision Tree đó là:

* Ưu điểm: cấu trúc đơn giản, dễ hiểu và xử lý nhanh với dữ liệu nhỏ và vừa.
* Nhược điểm: Dễ bị overfitting nếu không cắt tỉa, nhạy cảm với dữ liệu nhiễu. Hiệu suất giảm với dữ liệu lớn hoặc phức tạp.
* Khi xây dựng cây quyết định, người ta thường sử dụng các chỉ số như Entropy, Information Gain, hoặc Chỉ số Gini (Gini Index) để đánh giá mức độ phân chia dữ liệu, các công thức tính như sau:
* Entropy(S) hay E(S) =

​ + Gain(A, S) = E(S) −

* Gini = 1 −

### Introduction to Decision Tree Algorithm in Machine Learning | Analytics Steps

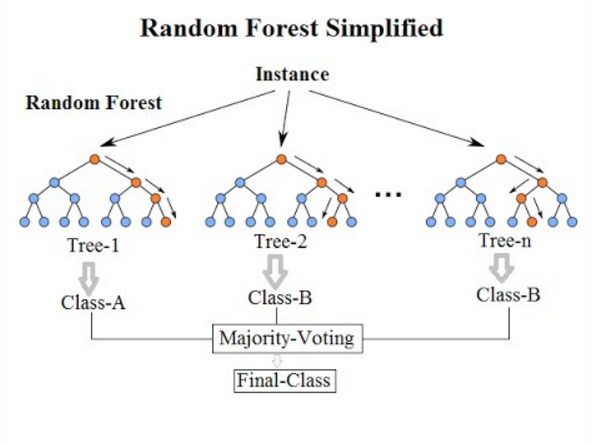
Hình 4. Hình minh họa thuật toán Decision Tree

### 2.3.4 Random Forest (RF)

Random Forest (RF) là một phương pháp học máy có giám sát (supervised learning), được sử dụng phổ biến cho cả hai bài toán phân loại và hồi quy. Thuật toán này hoạt động bằng cách huấn luyện một tập hợp các cây quyết định (decision trees), mỗi cây được xây dựng từ một tập con ngẫu nhiên của dữ liệu huấn luyện và một tập con ngẫu nhiên của các đặc trưng.

Trong bài toán phân loại, RF dự đoán bằng cách lấy đa số phiếu từ các cây trong rừng; còn trong bài toán hồi quy, RF tính trung bình dự đoán của các cây. Việc kết hợp nhiều cây giúp tăng độ chính xác và độ ổn định của mô hình. Các ưu nhược điểm của Random Forest:

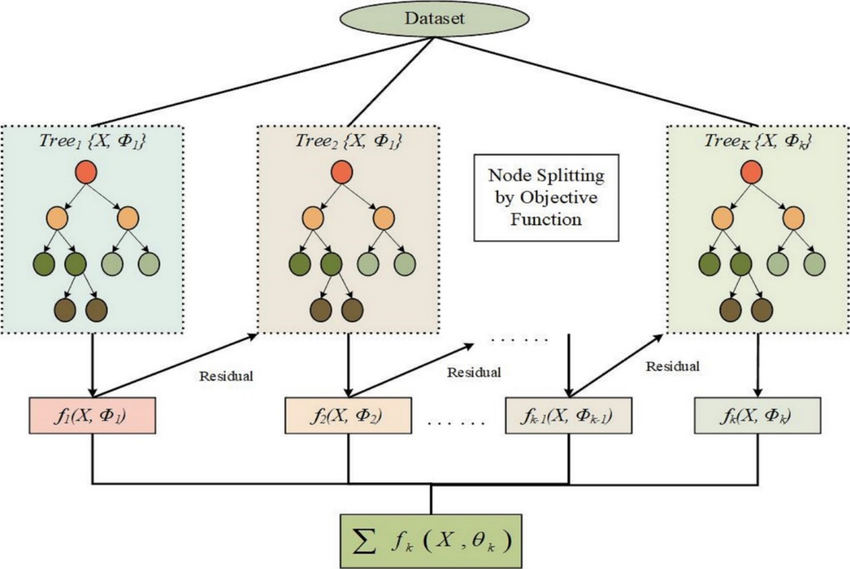
* Ưu điểm: Dễ sử dụng, độ chính xác thường cao, ổn định và mạnh mẽ, giảm overfitting, thích hợp với dữ liệu hỗn hợp.
* Nhược điểm: có thể tốn tài nguyên tính toán hơn và thời gian huấn luyện lâu hơn, có thể không tối ưu với các lớp có hiếm.



Hình 5. Hình minh họa thuật toán Random Forest

### 2.3.5 Extreme Gradient Boosting (XGBoost)

XGBoost là một phương pháp học máy rất hiệu quả, nổi bật với độ chính xác cao trong dự đoán và tốc độ xử lý nhanh. Đây là một kỹ thuật học tổ hợp (ensemble learning), hoạt động bằng cách kết hợp dự đoán từ nhiều cây quyết định để tạo ra mô hình mạnh mẽ hơn.



Hình 6. Hình minh họa thuật toán XGBoost

Điểm nổi bật của XGBoost là tập trung vào việc khắc phục các hạn chế của các phương pháp gradient boosting hiện có. Nó thực hiện điều này bằng cách huấn luyện các mô hình đơn giản theo từng bước lặp, nhằm sửa lỗi và tối ưu hóa hiệu suất. Thuật toán cũng sử dụng các kỹ thuật như chính quy hóa (regularization) và tính toán song song (parallelization) để tăng cường hiệu quả.

* Ưu điểm: Độ chính xác cao, tốc độ nhanh và hiệu quả vượt trội, có khả năng xử lý quan hệ phức tạp và được dùng rất rộng rãi.
* Nhược điểm: Việc tinh chỉnh (tuning) có thể phức tạp vì có nhiều tham số, chi phí tính toán có thể cao.

## 2.4 Các chỉ số đánh giá mô hình

Các chỉ số thường được dùng để đánh giá một mô hình bao gồm:

* Accuracy (Độ chính xác): Tỷ lệ tổng số dự đoán đúng trên toàn bộ dữ liệu. Đây là chỉ số tổng quát đo lường mức độ chính xác của mô hình.

Accuracy =

* Precision (Độ chính xác theo lớp dương):Tỷ lệ giữa số lượng dự đoán đúng của lớp dương (TP) trên tổng số lần mô hình dự đoán là dương (TP + FP).

Precision =

→ Dùng để đánh giá việc giảm sai dương tính (false positives).

* Recall (Khả năng tìm đúng lớp dương):Tỷ lệ giữa số lượng dự đoán đúng của lớp dương (TP) trên tổng số mẫu thực sự là dương (TP + FN).

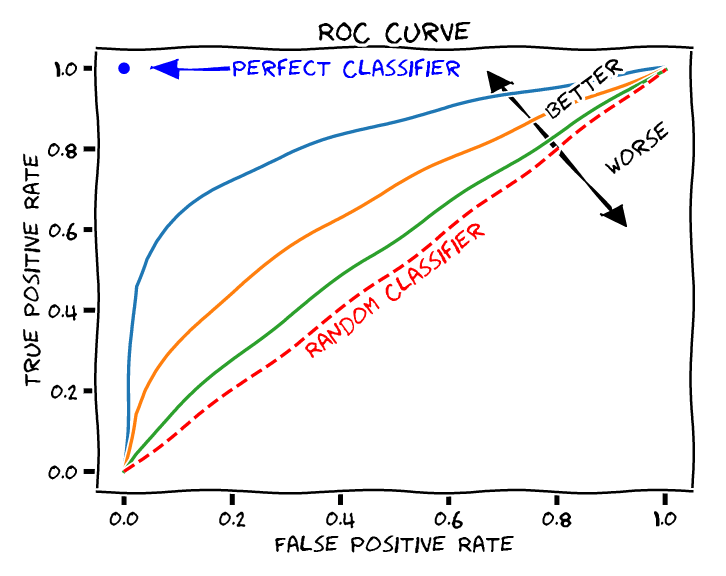
Recall =

→ Phản ánh khả năng tìm ra toàn bộ mẫu dương.

* F1-score: Trung bình điều hòa (harmonic mean) giữa Precision và Recall, giúp cân bằng giữa hai chỉ số này.

F1-score =

* Trong đó, TPi​ là số lượng dự đoán đúng của lớp thứ i (True Positive), FPi​ là số lượng dự đoán sai dương của lớp i (False Positive), FNi ​là số lượng bỏ sót (False Negative) của lớp i, l là số lượng lớp trong bài toán phân loại đa lớp, và β là hệ số cân bằng giữa precision và recall. Giá trị β phổ biến nhất là 1, khi đó chỉ số F trở thành trung bình điều hòa (harmonic mean) của precision và recall.
* AUC (Area Under the Curve): Diện tích dưới đường cong ROC, đo khả năng phân biệt giữa các lớp. AUC càng gần 1 thì mô hình càng tốt.



Hình 7. Hình minh họa AUC ROC

* Execution Time (Thời gian thực thi): Được dùng để đánh giá hiệu suất tính toán của mô hình khi huấn luyện và dự đoán. Mô hình nhanh thường được ưu tiên trong hệ thống thời gian thực.
* Model Size (Kích thước mô hình): Được đo bằng dung lượng bộ nhớ mà mô hình chiếm dụng sau khi huấn luyện. Mô hình nhẹ giúp triển khai dễ dàng và tiết kiệm tài nguyên.

## 2.5 Các kỹ thuật tiền xử lý dữ liệu

Các kỹ thuật tiền xử lý dữ liệu, bao gồm làm sạch, chuẩn hóa dữ liệu, lựa chọn đặc trưng, và xử lý mất cân bằng dữ liệu (SMOTE, RandomUnderSampler):

* Làm sạch dữ liệu: là quá trình xử lý các giá trị thiếu, không hợp lệ hoặc trùng lặp trong tập dữ liệu. Đề thực hiện các việc loại bỏ các bản ghi không cần thiết hoặc bị trùng, xử lý các giá trị bị thiếu (nếu có), xóa đi các cột đặc trưng gây nhiễu và chuyển đổi các giá trị về kiểu dữ liệu phù hợp. Việc làm sạch giúp nâng cao chất lượng dữ liệu và đảm bảo quá trình huấn luyện mô hình diễn ra ổn định, chính xác hơn.
* Chuẩn hóa dữ liệu: phương pháp chuẩn hóa z-score (hay còn gọi là standardization) để đưa các đặc trưng về cùng một thang đo. Mỗi giá trị đặc trưng sẽ được chuyển đổi theo công thức:

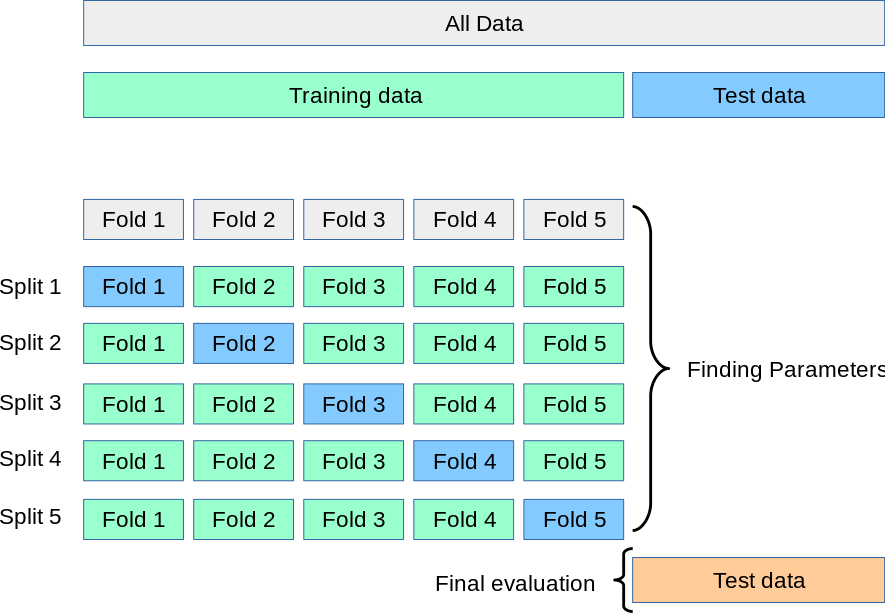
* Trong đó, x là giá trị gốc, là giá trị trung bình của đặc trưng, là độ lệch chuẩn.
* Chuẩn hóa giúp đảm bảo rằng tất cả các đặc trưng có phân phối với trung bình bằng 0 và độ lệch chuẩn bằng 1, từ đó giúp mô hình học máy hoạt động ổn định và tránh thiên lệch bởi các đặc trưng có giá trị lớn.
* Lựa chọn đặc trưng (feature selection): là bước trong tiền xử lý để chọn ra những đặc trưng (feature) có ảnh hưởng lớn đến kết quả dự đoán, đồng thời loại bỏ các đặc trưng không cần thiết hoặc gây nhiễu. Việc này giúp mô hình học nhanh hơn, giảm độ phức tạp và tránh bị overfitting. Random Forest có hỗ trợ lựa chọn đặc trưng bằng cách ưu tiên chọn các đặc trưng giúp giảm nhiều nhất độ hỗn loạn (như Gini index hoặc entropy) tại mỗi nút. Dựa trên số lần và mức độ mà từng đặc trưng được sử dụng trong các lần chia, thuật toán sẽ tính toán một chỉ số quan trọng (feature importance) cho từng đặc trưng. Các đặc trưng sau đó được xếp hạng theo chỉ số này.
* Xử lý mất cân bằng: Trong các bài toán học máy, đặc biệt là phân loại, dữ liệu thường bị mất cân bằng – tức là số lượng mẫu giữa các lớp không đồng đều. Để giải quyết vấn đề này, một trong những phương pháp được sử dụng phổ biến là SMOTE (Synthetic Minority Over-sampling Technique).
* SMOTE là kỹ thuật tạo thêm mẫu mới (synthetic samples) cho lớp thiểu số bằng cách sử dụng thuật toán K-láng giềng gần nhất (KNN). Thay vì sao chép lại các mẫu cũ, SMOTE tạo ra điểm dữ liệu mới bằng cách lấy hiệu giữa một điểm dữ liệu và hàng xóm gần nhất của nó, nhân với một số ngẫu nhiên từ 0 đến 1, rồi cộng kết quả đó lại với điểm ban đầu. Nhờ đó, các mẫu được tạo ra nằm giữa các điểm dữ liệu thật, giúp mô hình học được ranh giới phân lớp tốt hơn.
* Random Under Sampler là kĩ thuật giảm số lượng mẫu của lớp chiếm đa số. Cách làm này giúp cân bằng dữ liệu một cách đơn giản, nhanh chóng, đồng thời giảm độ chênh lệch giữa hai lớp, từ đó giúp mô hình học tốt hơn các đặc trưng của lớp thiểu số.

## 2.6. Kỹ thuật cross-validation kết hợp đánh giá trên Test

Cross-Validation (CV) là kỹ thuật đánh giá mô hình bằng cách chia nhỏ dữ liệu huấn luyện thành nhiều phần, huấn luyện và kiểm tra lặp lại trên các phần này, để đảm bảo kết quả đánh giá khách quan hơn. Trong đó, K-Fold Cross-Validation là phương pháp phổ biến, chia dữ liệu thành K phần, luân phiên dùng mỗi phần làm tập kiểm tra, phần còn lại để huấn luyện.

Tuy nhiên, nếu dữ liệu bị mất cân bằng giữa các lớp, việc chia ngẫu nhiên có thể gây lệch phân bố. Do đó, đề tài sử dụng Stratified K-Fold Cross-Validation, tức là chia đều số mẫu mỗi lớp cho các phần, đảm bảo tỷ lệ giữa các lớp được giữ nguyên trong mỗi phần. Cách làm này giúp mô hình được đánh giá công bằng và ổn định hơn trong các bài toán phân loại mất cân bằng như IDS.

Cross-validation được sử dụng phổ biến vì giúp tránh tình trạng đánh giá thiên lệch, nhất là khi dữ liệu không quá lớn. Nó cho phép đánh giá mô hình một cách khách quan, đồng thời hỗ trợ chọn ra những mô hình hoạt động tốt nhất trước khi đưa đi kiểm tra thực sự trên tập dữ liệu chưa từng thấy (test set).



Hình 8. Hình minh họa kĩ thuật Cross-Validation

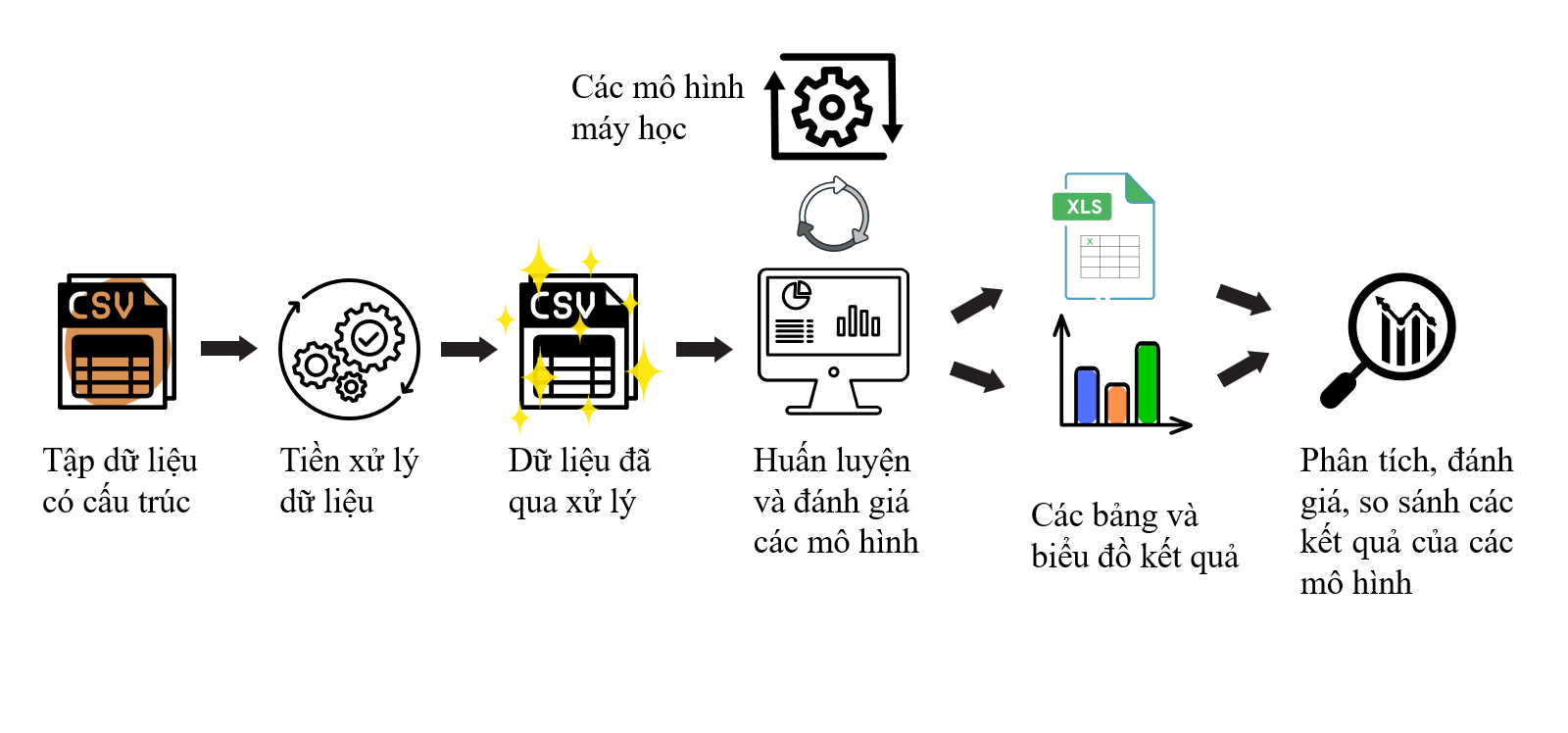
**Tổng kết chương 2:** Trong chương này, đề tài đã trình bày cơ sở lý thuyết và nền tảng để bắt tay vào thực nghiệm. Nội dung của chương tiếp theo sẽ trình bày các phương pháp thực hiện xây dựng các mô hình và thống kê, đánh giá.

# CHƯƠNG 3: PHƯƠNG PHÁP THỰC HIỆN

Bài toán toàn cục của đề tài sẽ được trình bày cụ thể ở chương thứ 3 này, bao gồm các công việc như tiền xử lý dữ liệu, huấn luyện các mô hình, thống kê, đánh giá và so sánh các mô hình.

## 3.1 Tổng quan bài toán

Đề tài được sẽ được triển khai với 3 giai đoạn cụ thể như sau:



Hình 9. Hình minh họa bài toán lớn của đề tài

## 3.2 Tiền xử lý dữ liệu

Tiền xử lý dữ bao gồm 8 bước chính: Chuyển đổi và gộp dữ liệu, trực quan hóa dữ liệu, làm sạch dữ liệu, mã hóa nhãn, chuẩn hõa dữ liệu, lựa chọn đặc trưng, xử lý mất cân bằng và chia dữ liệu.

### 3.2.1 Chuyển đổi và gộp dữ liệu

Do tập dữ liệu khi tải về từ Kaggle là 10 tệp parquet, thực hiện chuyển đổi kiểu dữ liệu parquet thành kiểu dữ liệu CSV cho cả 10 files. Để thuận tiện trong quá trình xử lý, 10 file CSV mới được gộp lại thành 1 file CSV thống nhất với tổng dung lượng là 2,237,060 KB (hơn 2GB).

A screen shot of a computer

Description automatically generated

Hình 10. Các file PARQUET nhận về sau khi tải từ Kaggle



Hình 11. File CSV lớn sau khi chuyển đổi và gộp 10 file PARQUET lại

### 3.2.2 Trực quan hóa dữ liệu

Trực quan hóa dữ liệu giúp quan sát được phân phối đặc trưng, tên các lớp, tỉ lệ lớp.

*A screenshot of a computer

Description automatically generated*

Hình 12. Các thông tin sau khi được in ra

*A graph with blue squares and numbers

Description automatically generated*

Hình 13. Phân phối các nhãn theo loại 7 lưu lượng

### 3.2.3 Làm sạch dữ liệu

Các bước làm sạch dữ liệu bao gồm:

* Nhập các thư viện hỗ trợ pandas.
* Kiểm tra các cột chứa dữ liệu toàn 0 hoặc hơn 98% dữ liệu chứa trong cột toàn 0, điều này có thể gây nhiễu trong tập dữ liệu:

A screenshot of a computer program

Description automatically generated

Hình 14. Số đặc trưng có giá trị 0 nhiều hơn 98%

* Kiểm tra giá trị thiếu, trùng lặp, chứa giá trị vô cùng

A white text with black text

Description automatically generated

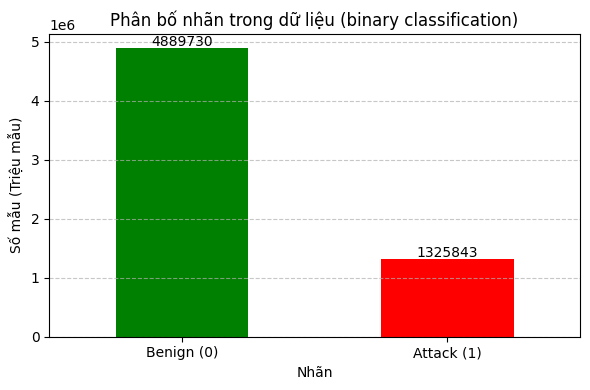
Hình 15. Kết quả số dòng bị thiếu, trùng lặp hoặc chứa giá trị vô cùng

* Vậy cần xóa các cột chứa hơn 98% giá trị 0 và các dòng bị lặp, sau đó chuyển đổi kiểu dữ liệu cho toàn bộ giá trị về float64, cuối cùng lưu kết quả ra file mới để thực hiện bước tiếp theo.

### 3.2.4 Mã hóa nhãn

Các bước thực hiện mã hóa nhãn gồm:

* Mã hóa các mẫu nhãn Benign về 0, và các kiểu tấn công thành 1 để thực hiện phân loại nhị phân (binary classification). Sau khi mã hóa lưu lại để thực hiện bước kế tiếp.
* Quan sát hình ở dưới đây, ta có thể thấy tập dữ liệu bị mất cân bằng, vậy cần phải có những giải pháp xử lý mất cân bằng phù hợp. Huấn luyện với tập dữ liệu bị mất cân bằng cũng là một lựa chọn thực tế vì các tập dữ liệu về phát hiện xâm nhập mạng thường mất cân bằng, nhưng kết quả có thể không khả quan.



Hình 16. Phân bố nhãn trong tập dữ liệu sau khi mã hóa nhãn

### 3.2.5 Chuẩn hóa dữ liệu

Dữ liệu được chuẩn hóa bằng phương pháp StandardScaler, giúp chuyển các đặc trưng về phân phối có trung bình bằng 0 và độ lệch chuẩn bằng 1. Sau khi chuẩn hóa lưu lại file để thực hiện bước tiếp theo.

### 3.2.6 Lựa chọn đặc trưng

Các bước lựa chọn đặc trưng phù hợp cho tập dữ liệu:

* Để lựa chọn được đặc trưng, cần nhập vào các thư viện hỗ trợ từ, sau đó đọc dữ liệu sau khi chuẩn hóa vào để bắt đầu huấn luyện mô hình Random Forest sử dụng hàm feature\_importances để xem độ quan trọng của các đặc trưng:

A graph with numbers and text

Description automatically generated

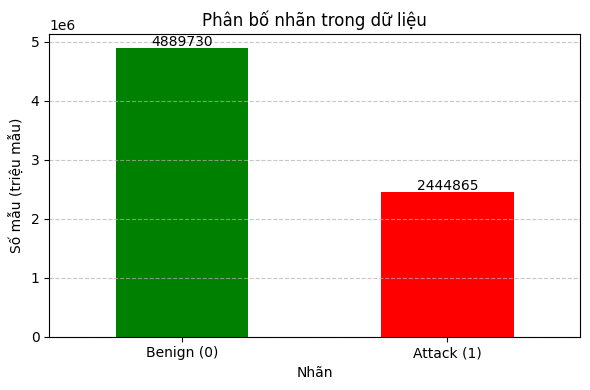
Hình 17. Top 30 đặc trưng quan trọng nhất

* Từ đây lựa chọn ra 15 đặc trưng quan trọng nhất bằng cách chọn ngưỡng chấp nhận là lớn hơn 0.025. Lưu lại file để thực hiện bước tiếp theo.

### 3.2.7 Xử lí mất cân bằng

Có 2 cách để xử lý mất cân bằng cho tập dữ liệu này:

* Cách xử lý mất cân bằng đầu tiên là smote, nhập vào thư viện smote từ thư viện imblearn.over\_sampling. Tính toán tỉ lệ để cân bằng số lượng mẫu nhãn ‘1’ lên đến 50% nhãn ‘0’ để hạn chế overfitting và giữ được sự phân bố tự nhiên của dữ liệu thay vì cân bằng lên đạt tỉ lệ 1:1.



Hình 18. Phân bố nhãn sau SMOTE

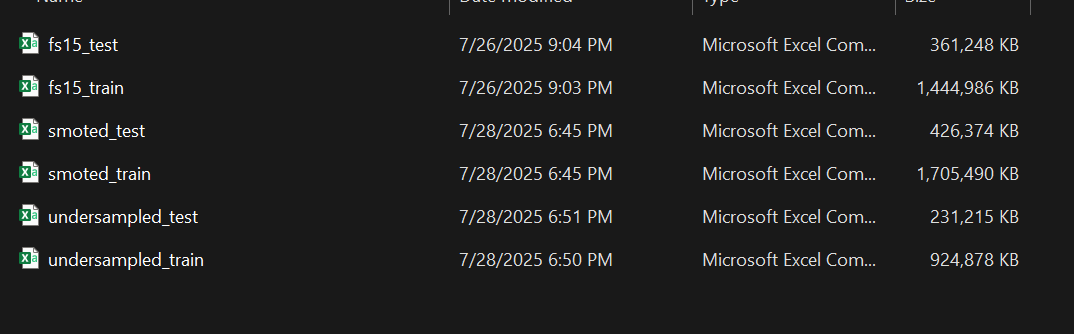
* Cách xử lý mất cân bằng thứ 2 là random under sampler, ta cũng nhập vào thư viện tính toán tỉ lệ để giảm nhãn ‘0’ xuống một lượng bằng 2 lần nhãn ‘1’. Không nên giảm quá nhiều vì sẽ làm mất mát thông tin.



Hình 19. Phân bố nhãn sau Random Under Sampler

### 3.2.8 Chia dữ liệu

Dùng hàm train\_test\_split để thực hiện chia 3 tập dữ liệu: sau lựa chọn đặc trưng (chưa cân bằng), cân bằng với SMOTE và cân bằng với RandomUnderSampler với công thức chia là 80% train và 20% test. Sau khi chia lưu vào thư mục riêng để sử dụng huấn luyện mô hình ở bước sau.



Hình 20. Các file dữ liệu sẽ được sử dụng để huấn luyện và đánh giá mô hình

## 

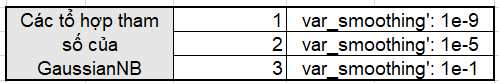
## 3.3 Huấn luyện và đánh giá các mô hình

Mỗi mô hình sẽ huấn luyện và đánh giá 3 lần với 3 tập dữ liệu: tập dữ liệu chưa cân bằng, tập dữ liệu cân bằng với SMOTE và tập dữ liệu cân bằng với RandomUnderSampler. Sau khi có các tập dữ liệu được chia ra tập train và test với tỉ lệ 8:2, tương ứng với mỗi tập dữ và mỗi mô hình sẽ thực hiện nạp vào các thư viện phù hợp cho mô hình đó. Tiếp theo, việc cài đặt các mô hình sẽ có các bước làm giống nhau:

* Sau khi nạp vào các thư viện cần thiết, các mô hình học máy được huấn luyện trên tập train và đánh giá bằng nghi thức 5-fold Stratified Cross-Validation. Quá trình huấn luyện ở mỗi mô hình được thực hiện nhiều lần với các tổ hợp tham số khác nhau, nhằm kiểm tra độ ảnh hưởng của từng cấu hình đến kết quả phân loại. Các tổ hợp này được chọn dựa trên giá trị mặc định trong thư viện hoặc tham khảo từ các nghiên cứu trước.
* Sau khi hoàn tất đánh giá bằng cross-validation, mỗi mô hình tiếp tục được kiểm thử trên tập test chưa biết để kiểm tra khả năng tổng quát hóa.
* Kết quả từ cả hai bước được lưu vào file XLSX, bao gồm các chỉ số: Accuracy (%), Precision (%), Recall (%), F1-score (%), AUC (%), Execution time (s) và Model size (KB) để thống kê và so sánh.
* Cuối cùng in các biểu đồ:
* Biểu đồ nhóm cột để so sánh kết quả trên CV và trên tập dữ liệu Test với số nhóm cột tương ứng với số tổ hợp tham số thể hiện 3 chỉ số Accuracy (%), Recall (%) và AUC (%).
* Ma trận nhầm lẫn cho từng tổ tham số để xem các kết quả đánh giá.

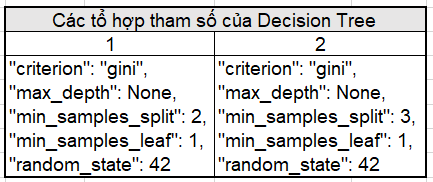
Điểm riêng duy nhất đó là mỗi mô hình sẽ dùng các bộ tham số khác nhau do thuật toán khác nhau, dưới đây là các tổ hợp tham số của từng mô hình và lý do vì sao đề tài sử dụng:

* K-Nearest Neighbors (KNN): KNN sử dụng bộ tham số mặc định từ thư viện sciket-learn như k=5, khoảng cách Euclidean và trọng số đều. Cách sử dụng cấu hình mặc định này giúp đơn giản hóa quá trình huấn luyện, đồng thời cũng được áp dụng trong một số nghiên cứu trước đó như trong nghiên cứu của Songma et al. (2023).
* Gaussian Naive Bayes (GaussianNB): Quá trình huấn luyện sử dụng 3 giá trị khác nhau của tham số var\_smoothing, lần lượt là 1e-9, 1e-5 và 1e-1 để tránh trường hợp chia cho 0 trong tính xác xuất. 3 giá trị được chọn đại diện cho mức làm mượt thấp, trung bình và cao cho GaussianNB.



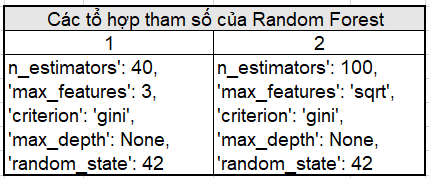
Hình 21. Các tổ tham số của GaussianNB

* Decision Tree (DT): Quá trình huấn luyện Decision Tree sử dụng 2 tổ hợp tham số khác nhau được tham khảo từ nghiên cứu của Songma et al. (2023) và từ thư viện mặc định của scikit-learn.



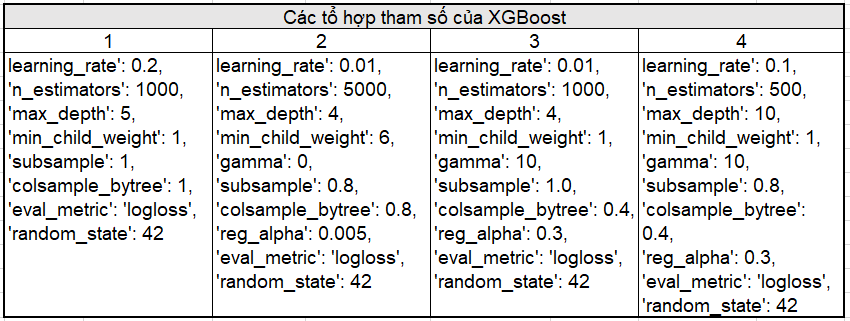
Hình 22. Các tổ hợp tham số của Decision Tree

* Random Forest (RF): Quá trình huấn luyện Random Forest sử dụng 2 tổ hợp tham số khác nhau được tham khảo từ nghiên cứu của Songma et al. (2023) và từ thư viện mặc định của scikit-learn.



Hình 23. Các tổ hợp tham số của Random Forest

* Extreme Gradient Boost (XGBoost): Quá trình huấn luyện XGBoost tham khảo sử dụng 4 tổ hợp tham số khác nhau cho ra kết quả khả quan trong 2 bài nghiên cứu của Songma et al. (2023) và nghiên cứu của Chimphlee et al. (2024).

Hình 24. Các tổ hợp tham số của XGBoost

**Tổng kết chương 3:** Tại chương 3 này, đề tài đã trình bày phương pháp thực hiện tiền xử lý, huấn luyện và đánh giá các mô hình sử dụng nhiều tổ hợp tham số khác nhau trên nhiều tập dữ liệu. Đến đây, bức tranh bài toán lớn đã hoàn thành được một nữa, phần còn lại ở các chương sau sẽ là thu các kết quả thực nghiệm và so sánh, đánh giá.

# CHƯƠNG 4: KẾT QUẢ THỰC NGHIỆM

## 4.1 Thu thập kết quả

Kết quả được thu thâp từ tập dữ liệu CSE-CIC-IDS2018 của viện nghiên cứu an ninh mạng thuộc Đại học New Brunswick (University of New Brunswick – UNB), Canada. Tập dữ liệu có 16,233,002 dòng, với 7 kiểu lưu lượng gồm Benign (bình thường), DDos, Dos, Brute force, Botnet, tấn công xâm nhập (Infilteration), tấn công Web. Tuy nhiên, phiên bản được đề tài sử dụng là phiên bản tinh gọn lại tập dữ liệu CSE-CIC-IDS2018 gốc, được đăng lên Kaggle bởi Nghiên cứu viên sau tiến sĩ Laurens D'hooge tại Đại học Ghent (Bỉ), tập dữ liệu CSE-CIC-IDS2018 được tinh gọn này có 6,659,532 dòng, 77 đặc trưng và gồm có 10 file kiểu dữ liệu PARQUET.

## 4.2 Môi trường thực nghiệm

Các công cụ được sử dụng trong đề tài:

* Môi trường lập trình tích hợp (IDE) là Visual Studio Code.
* Ngôn ngữ lập trùng được dùng là Python với các thư viện hỗ trợ như: scikit-learn, pandas, matplotlib,...

Cấu hình máy tính được sử dụng trong đề tài:

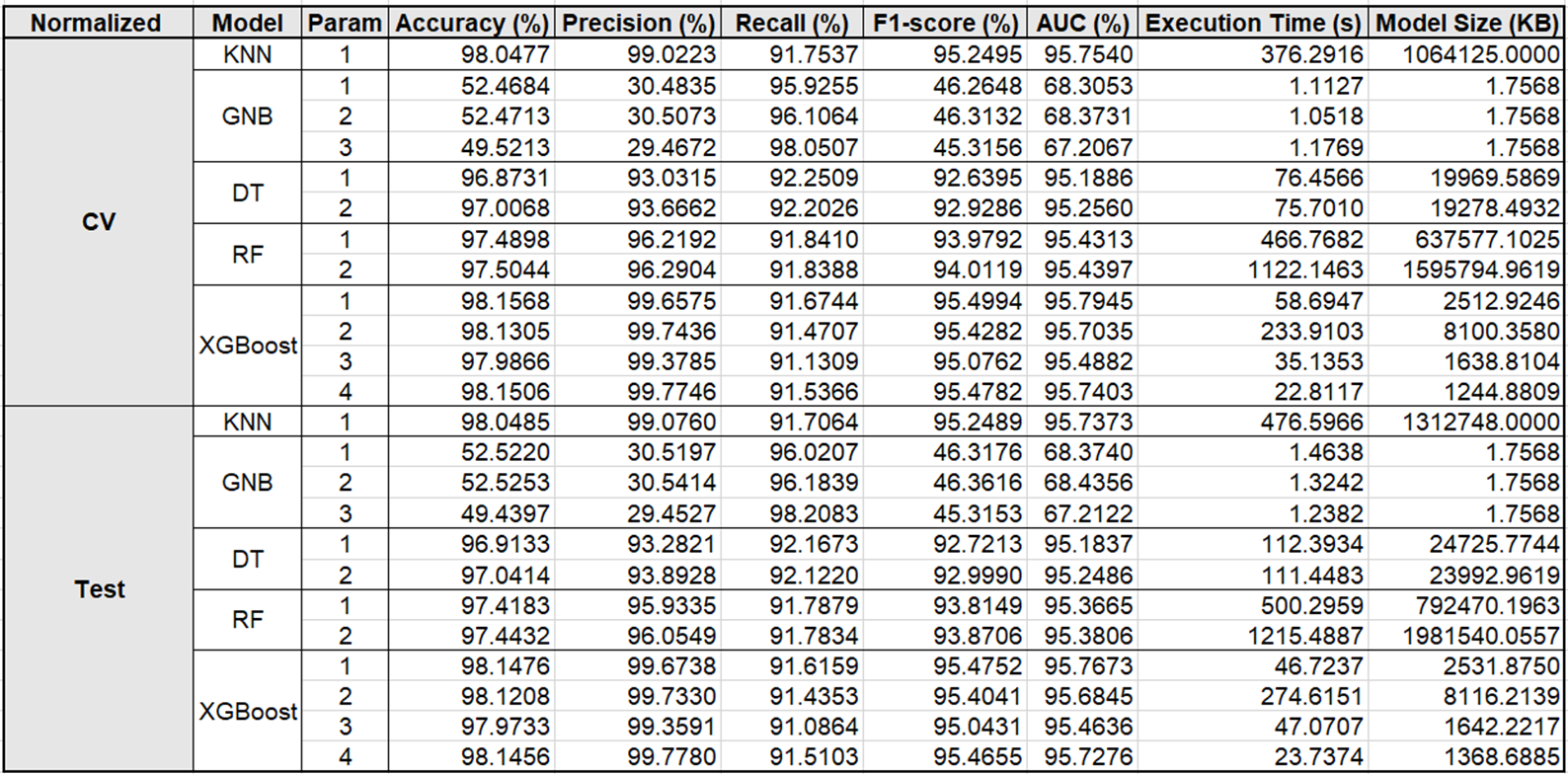
|  |  |
| --- | --- |
| Phần cứng | Thông số kỹ thuật |
| CPU | Intel(R) Core(TM) i7-12700H (20 nhân) 2.3GHz |
| Ram | 16GB DDR4 |
| Rom | SSD 512GB |

Hình 25. Các thông số kĩ thuật của phần cứng sử dụng trong đề tài

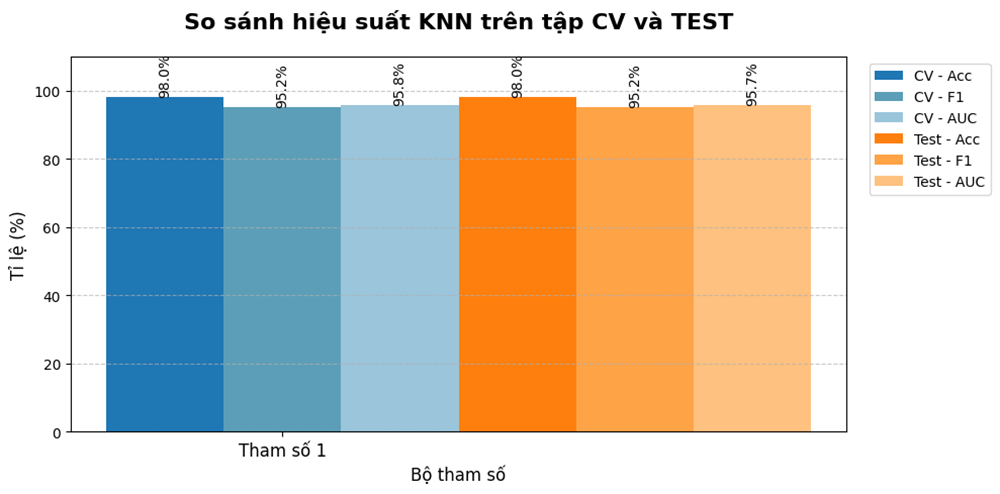
## 4.3 Kết quả thực nghiệm

Sau khi tìm hiểu và nghiên cứu cơ sở lý thuyết, áp dụng các kiến thức và kĩ năng thực hành để hoàn tất quá trình tiền xử lý, huấn luyện và đánh giá các mô hình, các kết quả thu được các kết quả như sau:

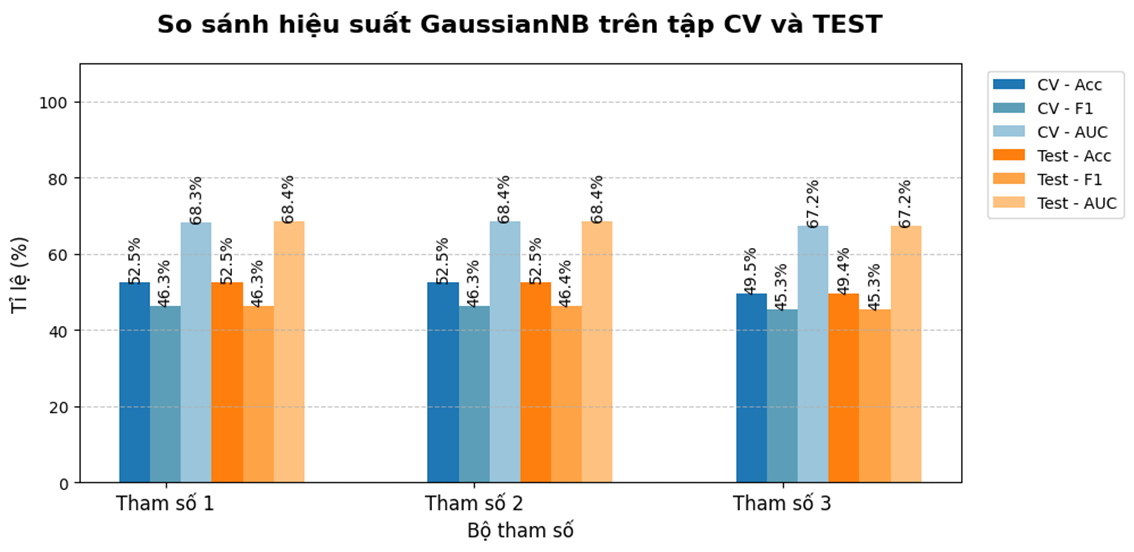
* Đối với tập dữ liệu chưa cân bằng:

****

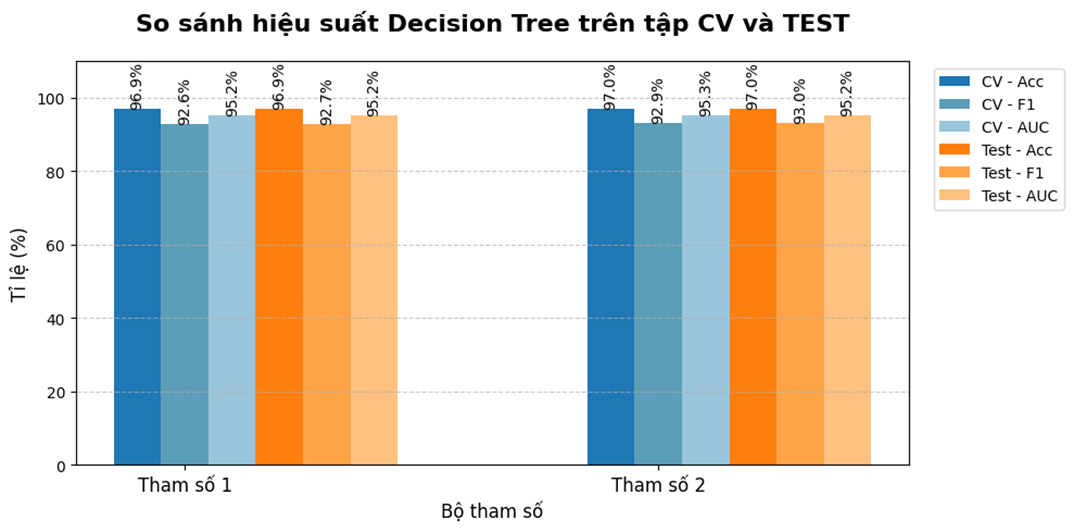
Hình 26. Bảng thống kê kết quả các mô hình trên tập chưa cân bằng

**

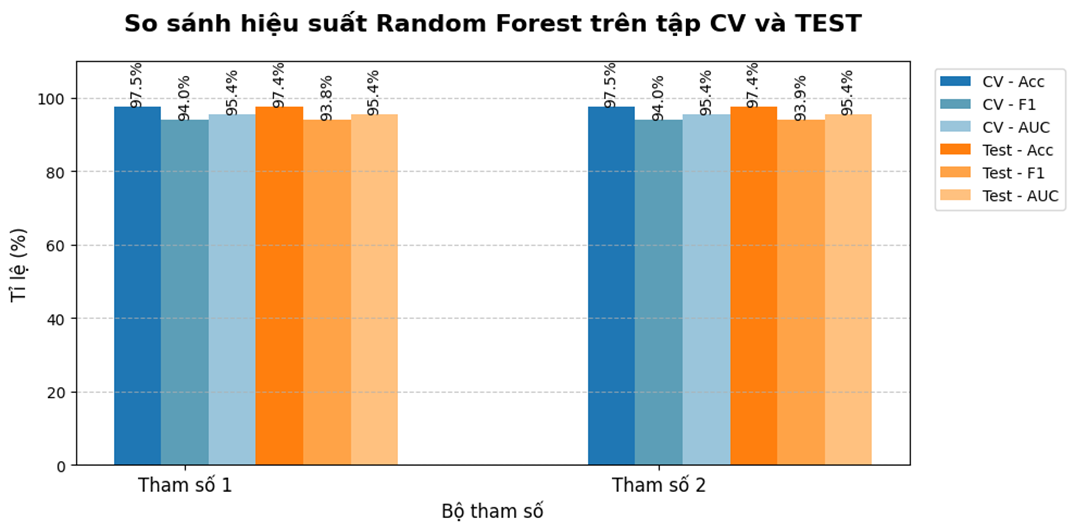
Hình 27. Biểu đồ so sánh hiệu suất KNN trên tập CV và Test

**

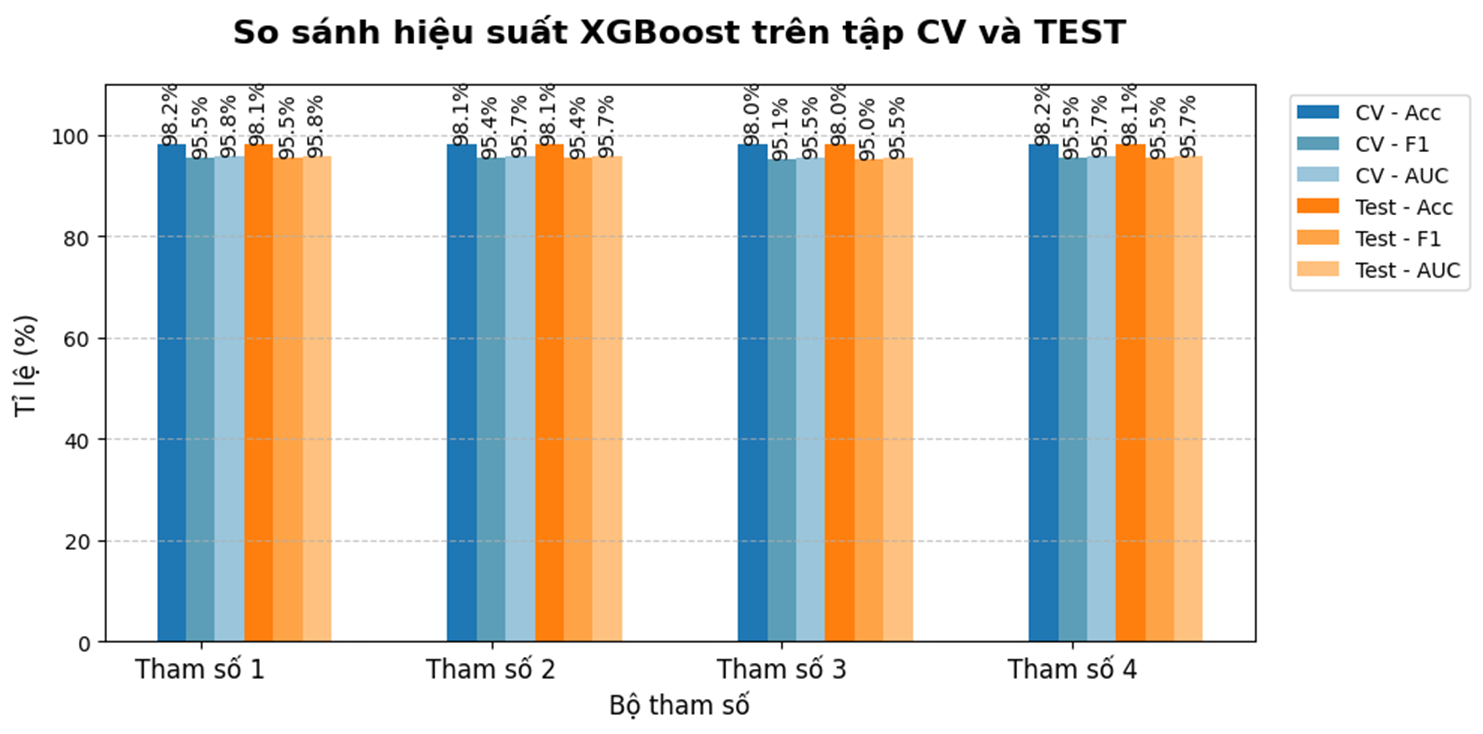
Hình 28. Biểu đồ so sánh hiệu suất GaussianBayes trên tập CV và Test

**

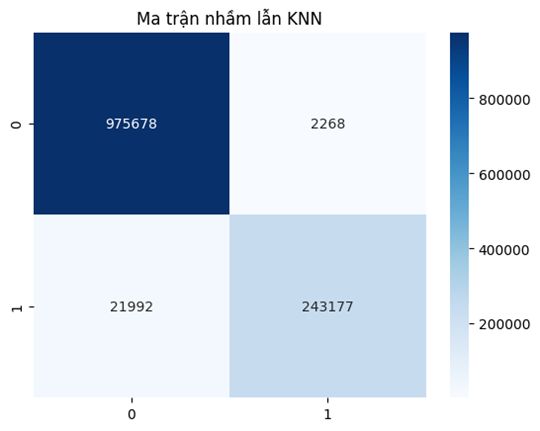
Hình 29. Biểu đồ so sánh hiệu suất DecisionTree trên tập CV và Test

**

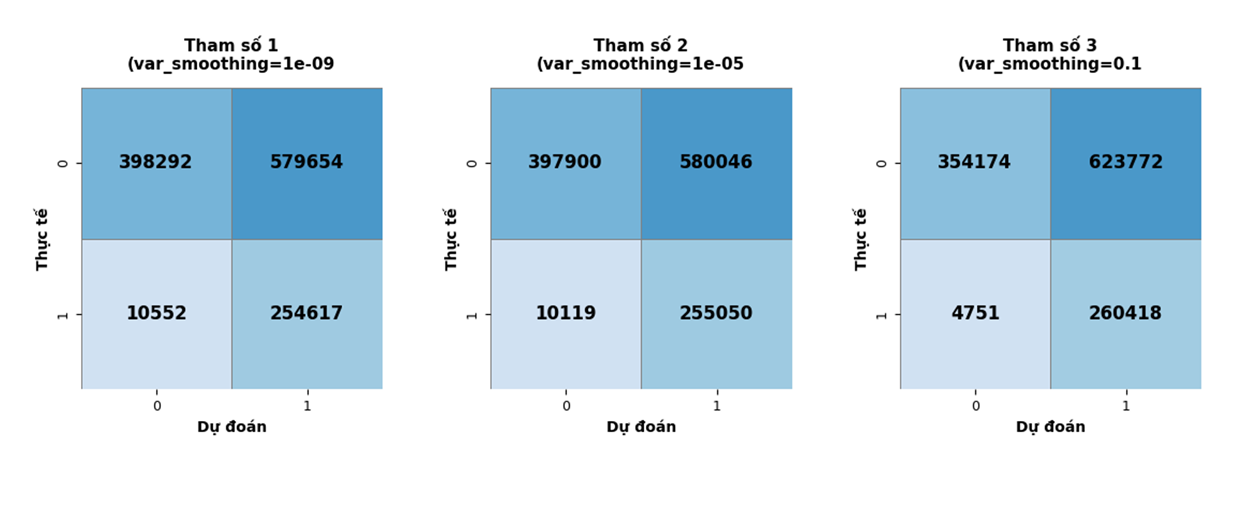
Hình 30. Biểu đồ so sánh hiệu suất Random Forest trên tập CV và Test

**

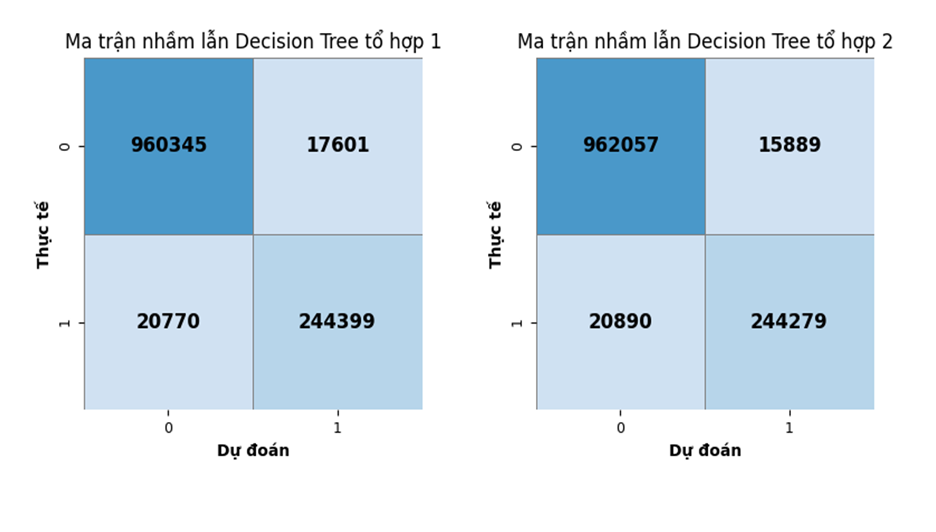
Hình 31. Biểu đồ so sánh hiệu suất XGBoost trên tập CV và Test

**

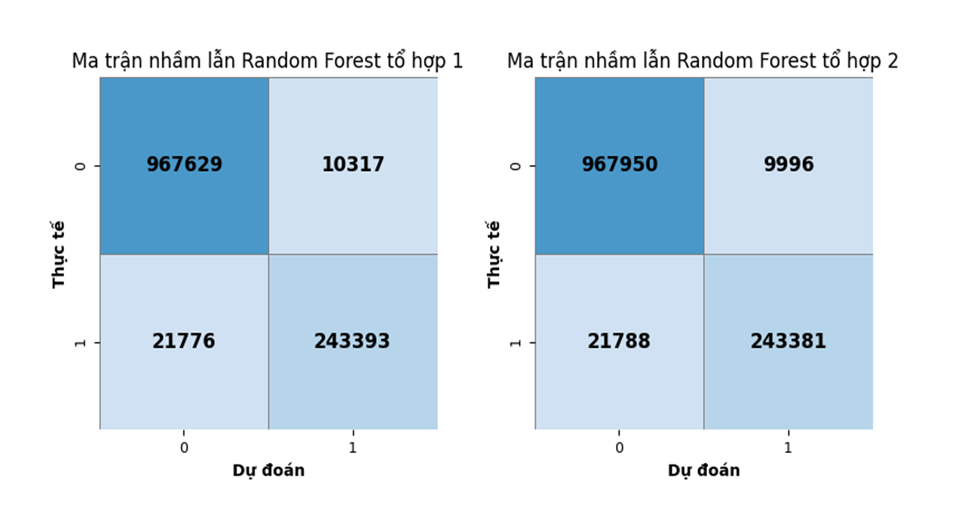
Hình 32. Ma trận nhầm lẫn của mô hình KNN

**

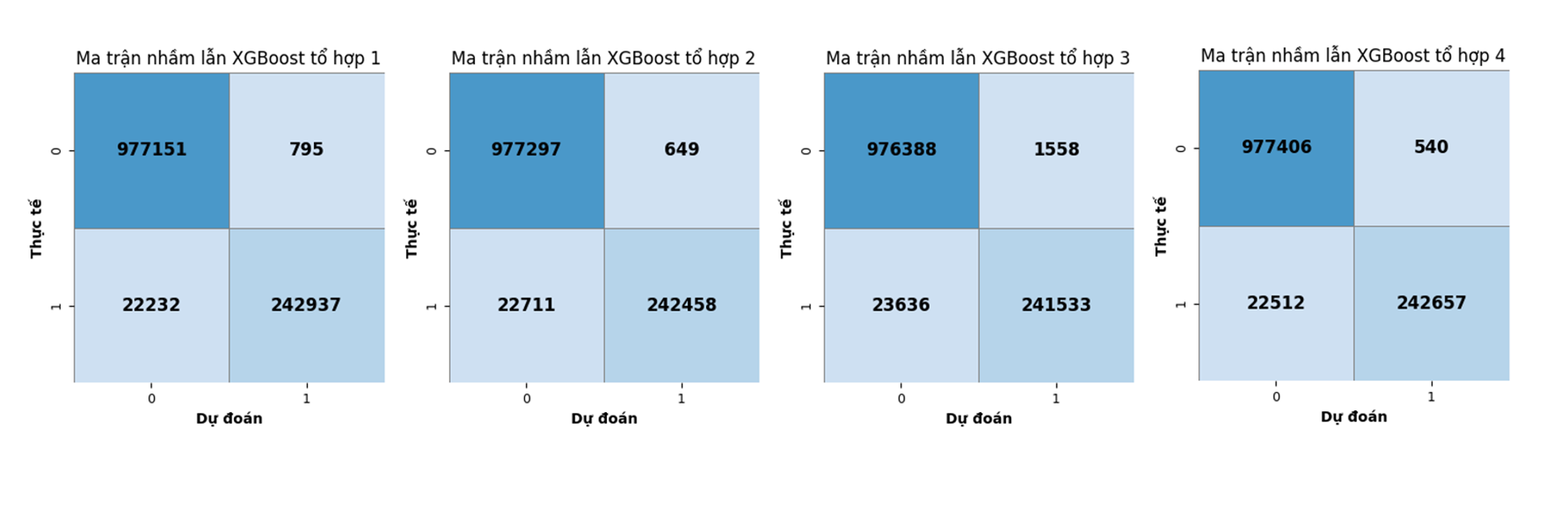
Hình 33. Ma trận nhầm lẫn của mô hình GaussianNB

**

Hình 34. Ma trận nhầm lẫn của mô hình DecisionTree

**

Hình 35. Ma trận nhầm lẫn của mô hình Random Forest

**

Hình 36. Ma trận nhầm lẫn của mô hình XGBoost

Sau khi huấn luyện trên dữ liệu mất cân bằng, các mô hình KNN, Decision Tree, Random Forest và XGBoost đều cho kết quả tốt, với Accuracy và các chỉ số Precision, Recall, F1-score đều vượt 90%.

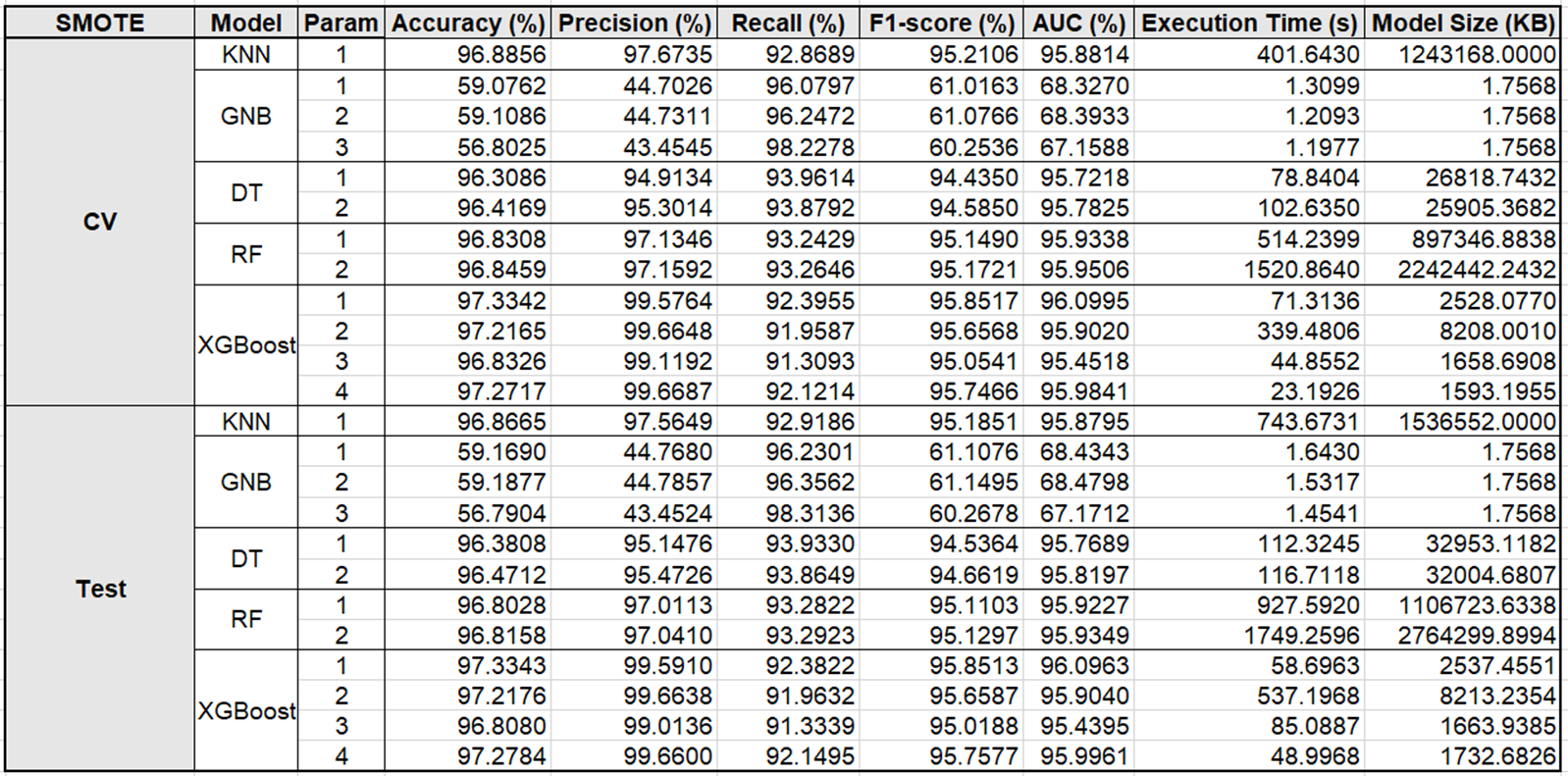
KNN dùng tham số mặc định đạt Accuracy trên 98%, AUC ~95.7%, mô hình chiếm khá nhiều bộ nhớ (~1GB) và thực thi chậm hơn XGBoost.

Random Forest và Decision Tree ổn định, riêng RF có độ chính xác cao nhưng chiếm dung lượng lớn (trên 1.5GB) do tham số sử dụng. XGBoost nổi bật với hiệu năng cao và mô hình khá gọn (~2.5MB ở cấu hình tốt nhất).

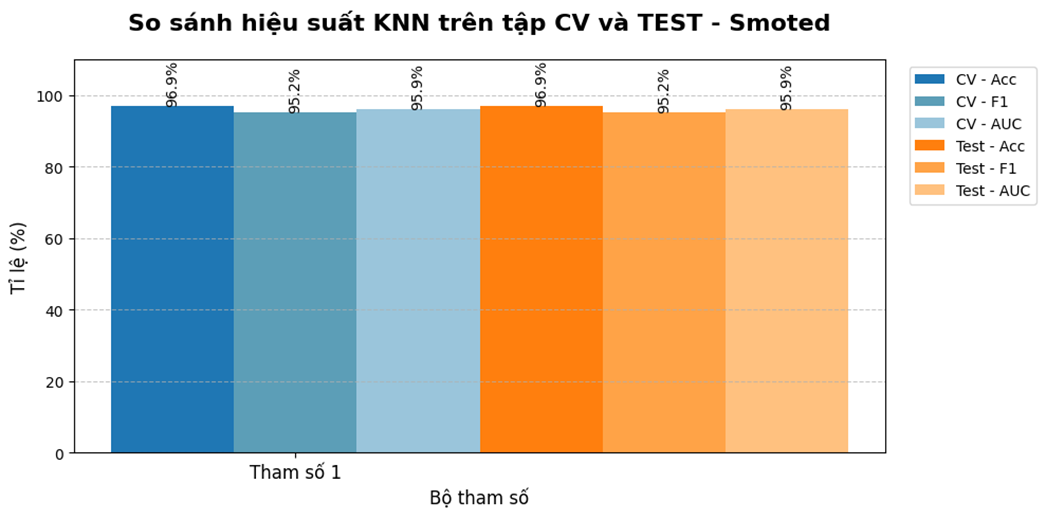
Ngược lại, GaussianNB có kết quả thấp nhất. Dù Recall cao (~96%), Precision chỉ ~30% khiến F1-score và AUC bị kéo xuống (~68%). Ma trận nhầm lẫn cho thấy GNB dự đoán quá nhiều nhãn tấn công, gây lệch phân loại. Dù xử lý rất nhanh và nhẹ, GNB không phù hợp với dữ liệu IDS vốn có phân phối không đều và phức tạp.

Tóm lại, XGBoost và Random Forest là hai mô hình phù hợp nhất cho bài toán phát hiện xâm nhập.

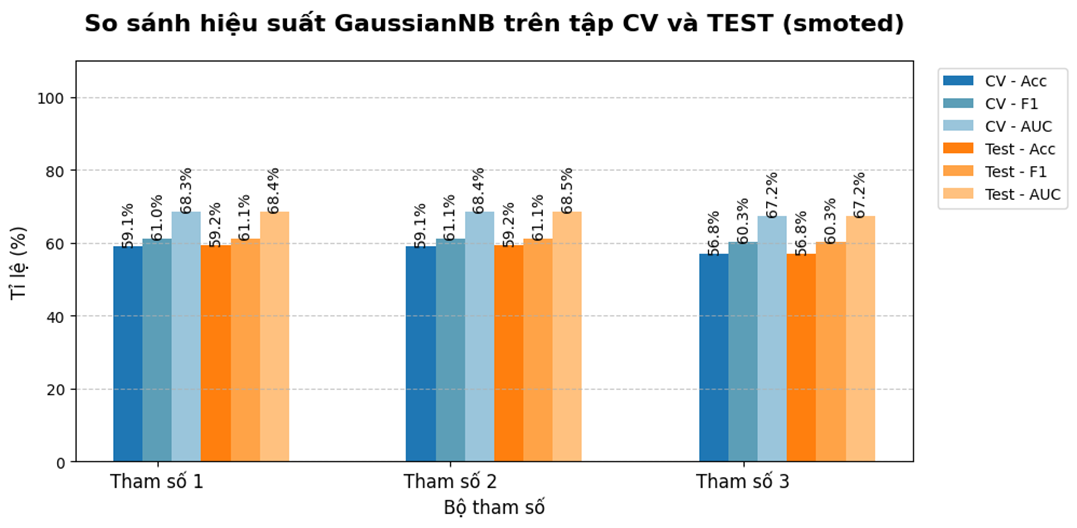
* Đối với tập dữ liệu đã cân bằng bằng phương pháp SMOTE:



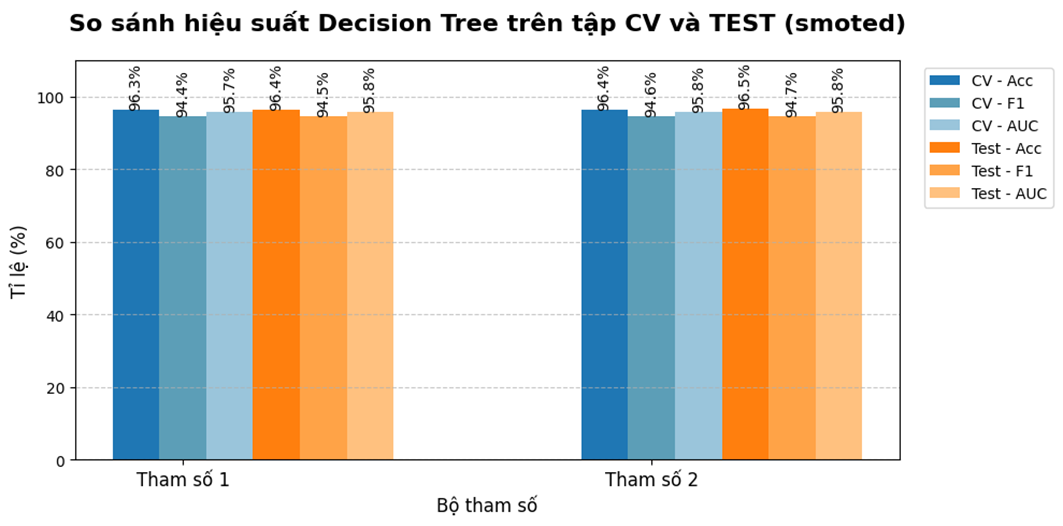
Hình 37. Bảng thống kê kết quả các mô hình trên tập đã SMOTE

**

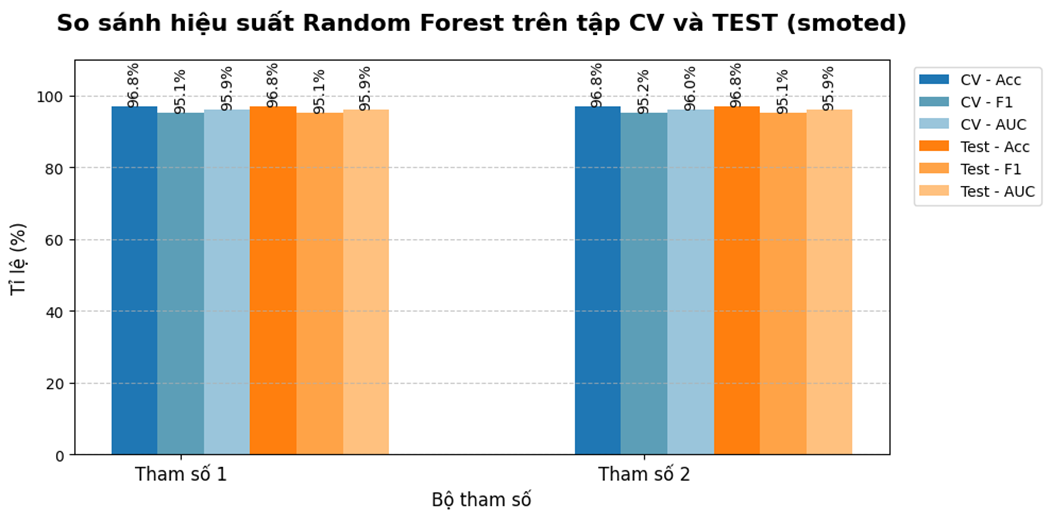
Hình 38. Biểu đồ so sánh hiệu suất KNN trên tập CV và Test (SMOTE)

**

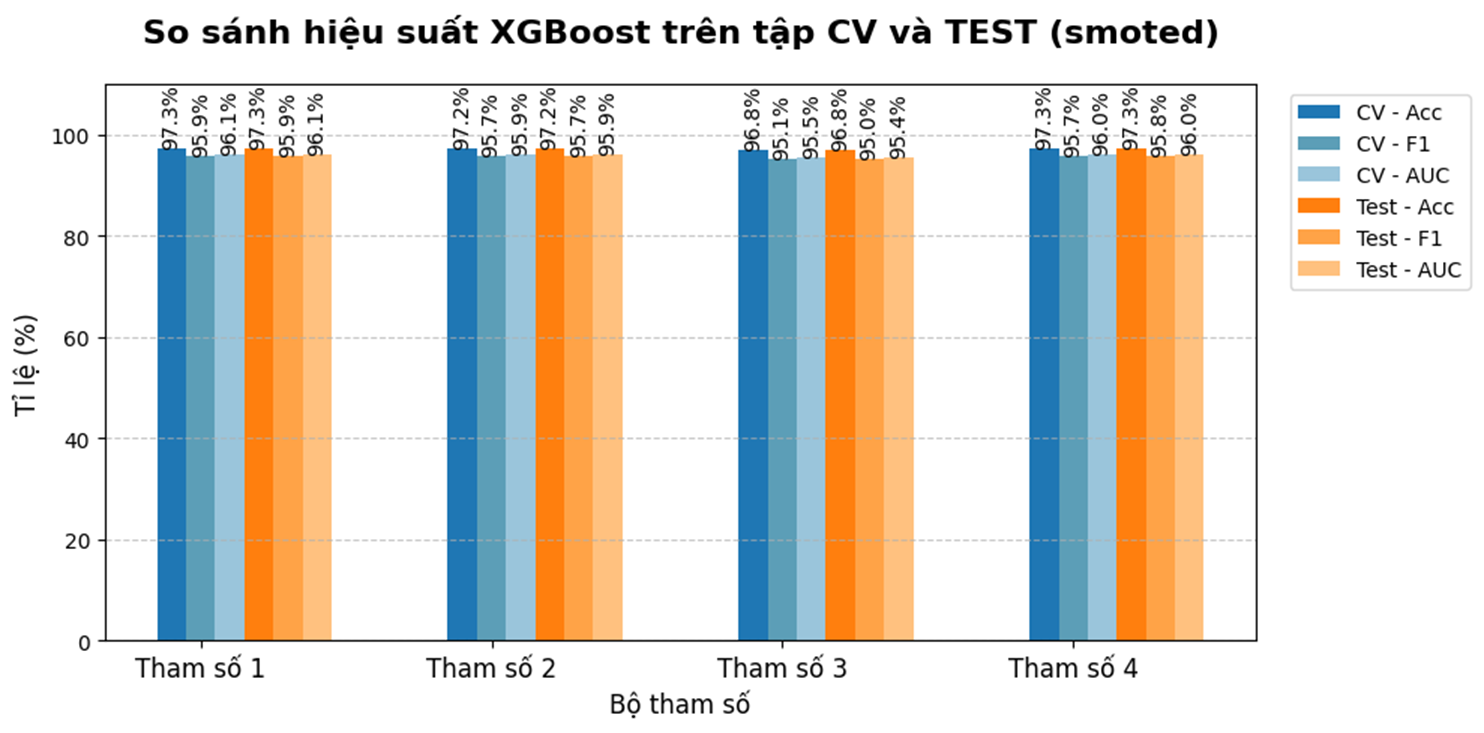
Hình 39. Biểu đồ so sánh hiệu suất GaussianNB trên tập CV và Test (SMOTE)

**

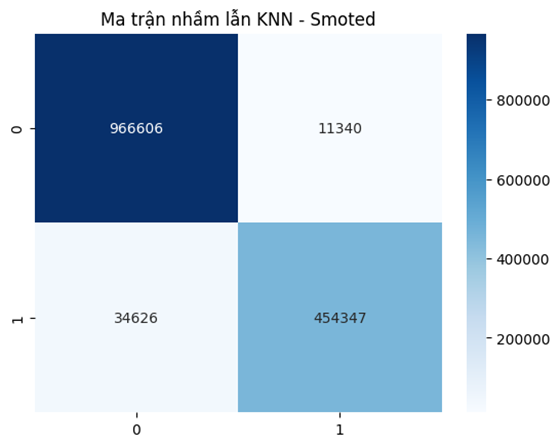
Hình 40. Biểu đồ so sánh hiệu suất DecisionTree trên tập CV và Test (SMOTE)

**

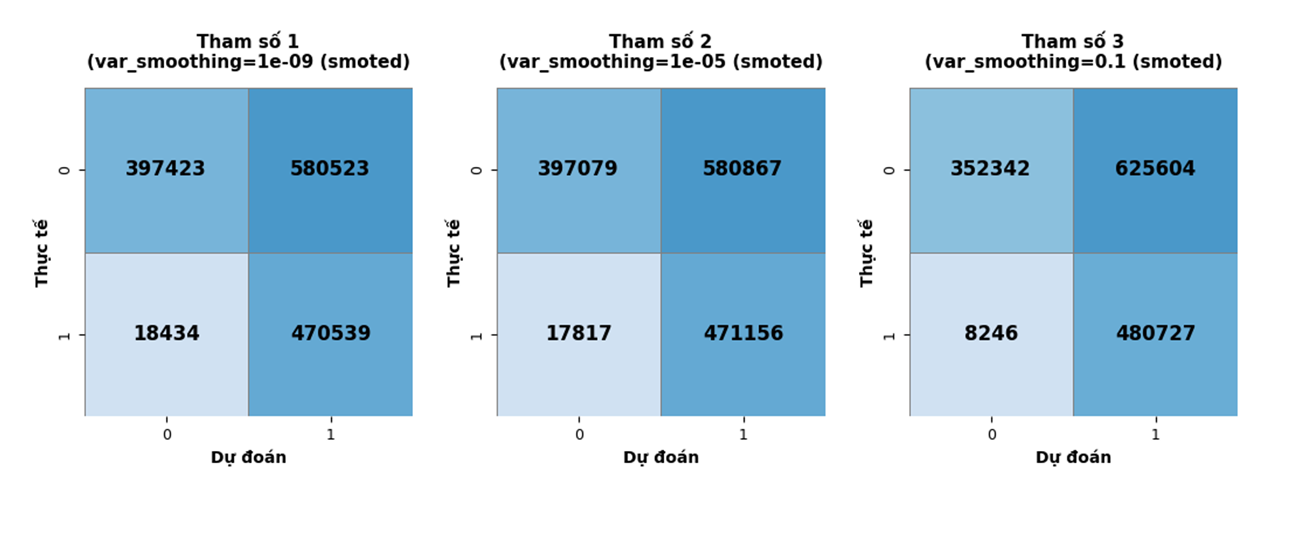
Hình 41. Biểu đồ so sánh hiệu suất Random Forest trên tập CV và Test (SMOTE)

**

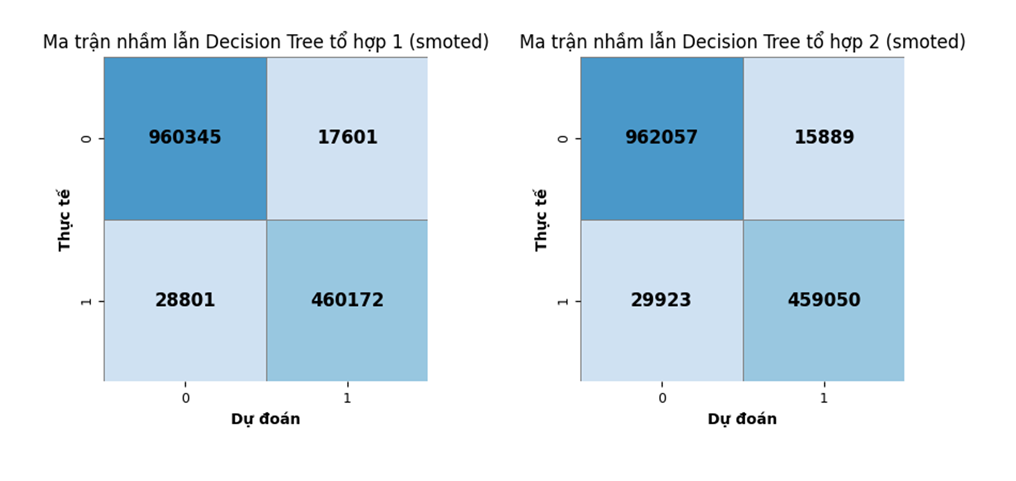
Hình 42. Biểu đồ so sánh hiệu suất XGBoost trên tập CV và Test (SMOTE)

**

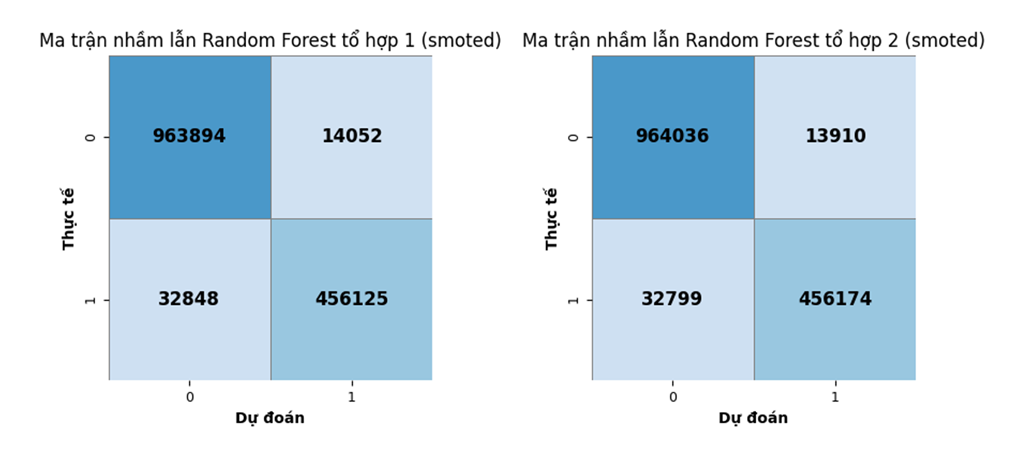
Hình 43. Ma trận nhầm lẫn KNN(SMOTE)

**

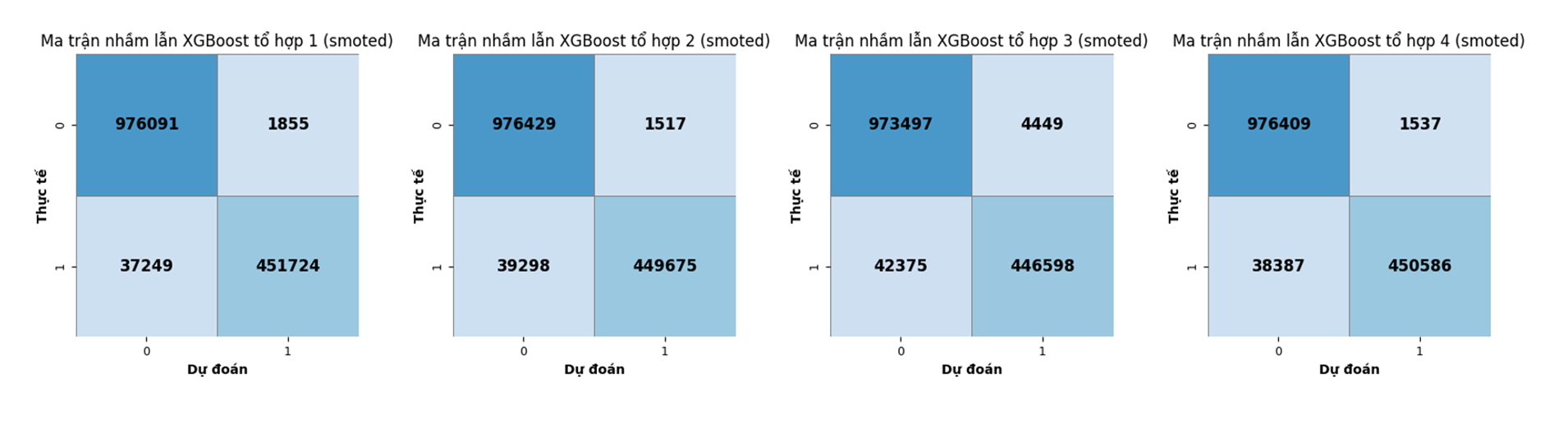
Hình 44. Ma trận nhầm lẫn GaussianNB (SMOTE)

**

Hình 45. Ma trận nhầm lẫn DecisionTree (SMOTE)

**

Hình 46. Ma trận nhầm lẫn Random Forest (SMOTE)

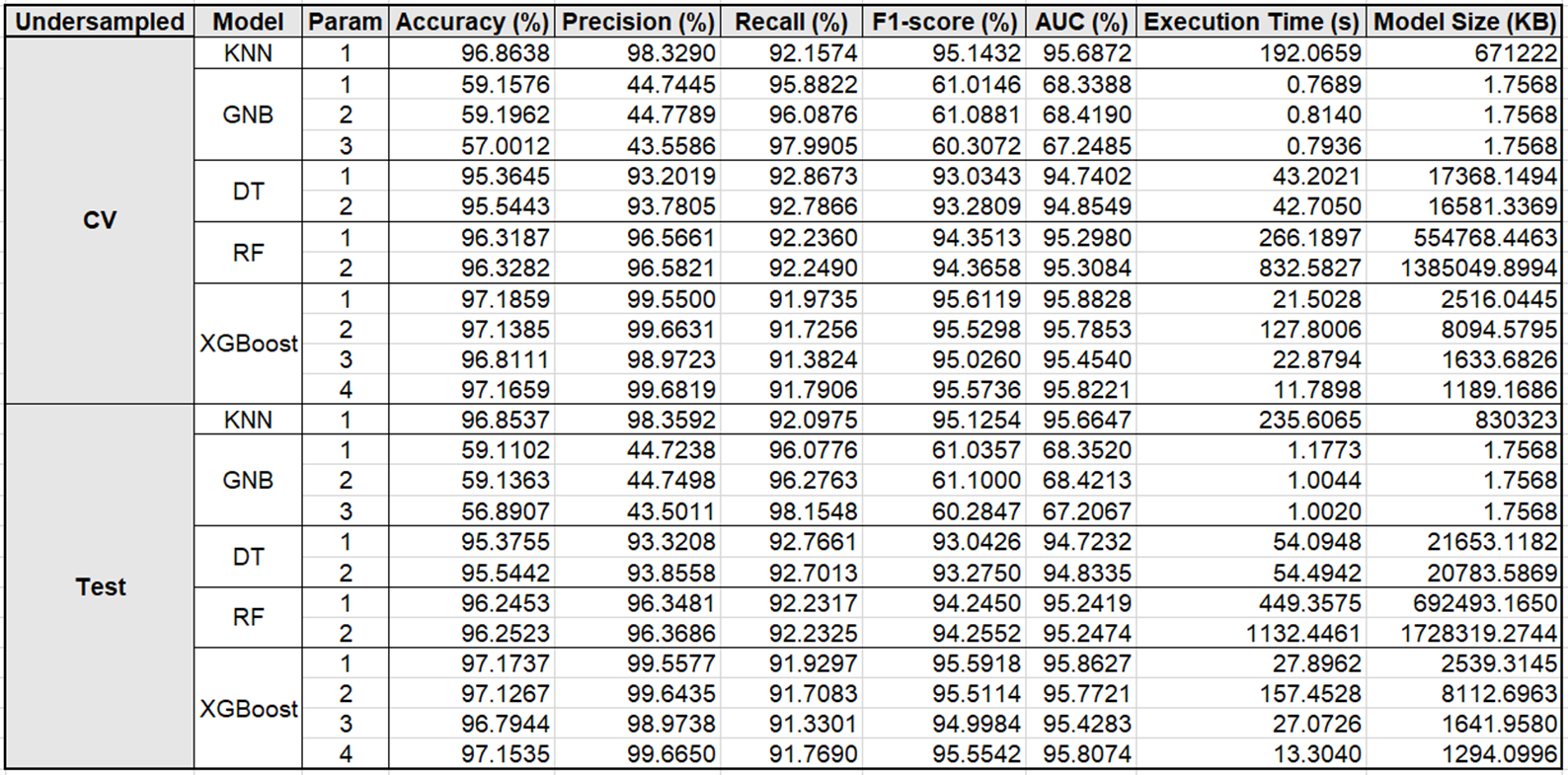
**

Hình 47. Ma trận nhầm lẫn XGBoost (SMOTE)

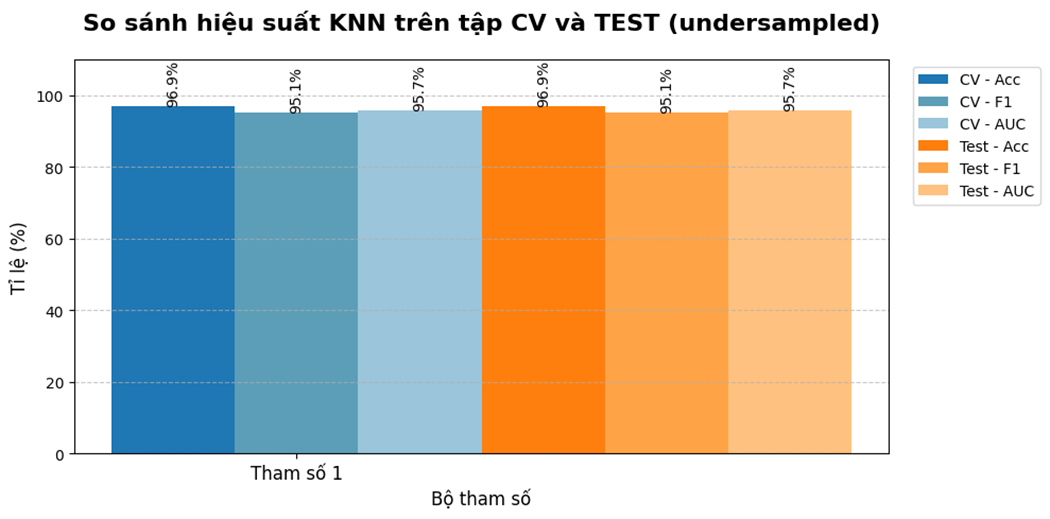
Sau khi áp dụng SMOTE để cân bằng, phần lớn các mô hình vẫn giữ được hiệu suất ổn định. Cụ thể, các mô hình như KNN, Decision Tree, Random Forest và XGBoost đều duy trì độ chính xác cao (trên 96%), với các chỉ số Precision và Recall tương đối cân đối – cho thấy khả năng phân loại giữa hai lớp không bị lệch nhiều. Đối với Gaussian Naive Bayes, Precision và F1-score có cải thiện nhẹ so với trước, nhưng nhìn chung kết quả vẫn kém hơn đáng kể so với các mô hình còn lại.

Tuy nhiên, xem xét ma trận nhầm lẫn, đặc biệt là ở DT, RF và XGBoost, có thể nhận thấy một điểm hạn chế: SMOTE không thực sự cải thiện khả năng nhận diện nhãn tấn công (label 1). Ngược lại, số lượng mẫu tấn công bị dự đoán sai thành benign (False Negative) lại tăng lên. Điều này cho thấy khả năng phát hiện tấn công của mô hình có xu hướng giảm, dù các chỉ số tổng thể vẫn cao do các mẫu tổng hợp từ SMOTE không phản ánh đúng bản chất dữ liệu thực tế, khiến mô hình học sai ranh giới phân lớp. Do đó, khi áp dụng SMOTE, cần kết hợp đánh giá bằng ma trận nhầm lẫn để có cái nhìn đầy đủ, thay vì chỉ dựa vào Accuracy hay F1-score.

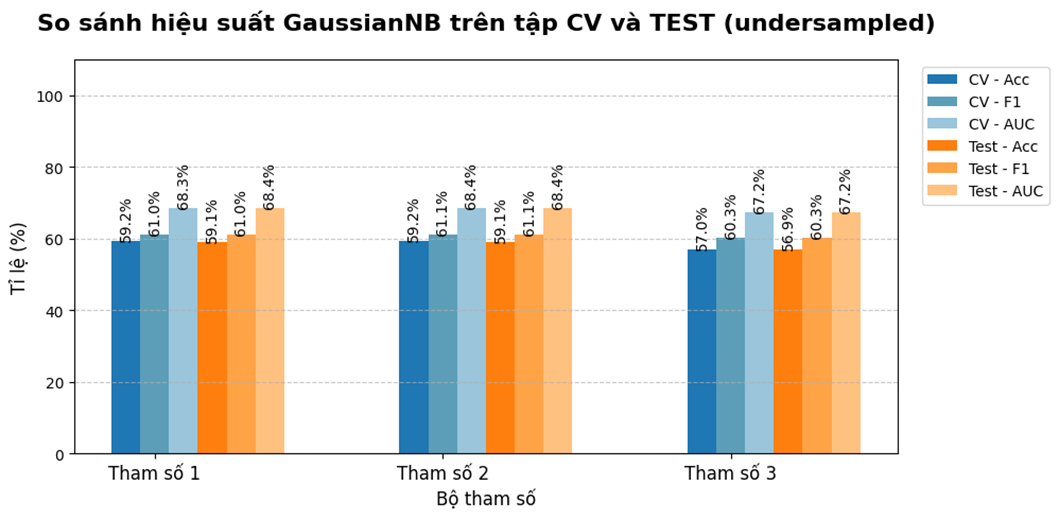
* Đối với tập dữ liệu cân bằng bằng RandomUnderSampler:



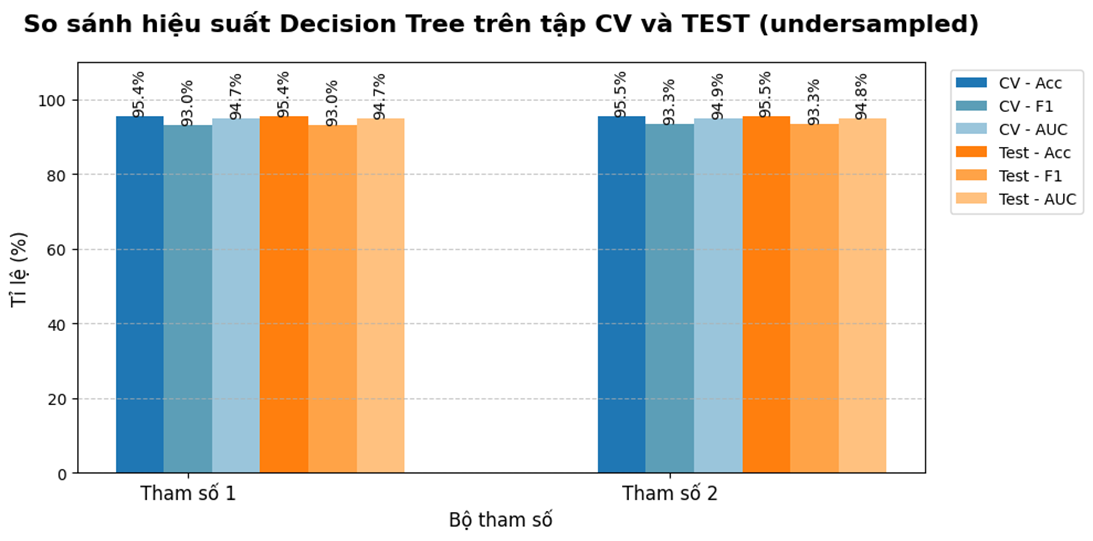
Hình 48. Bảng thống kê kết quả các mô hình trên tập đã RandomUnderSampler

**

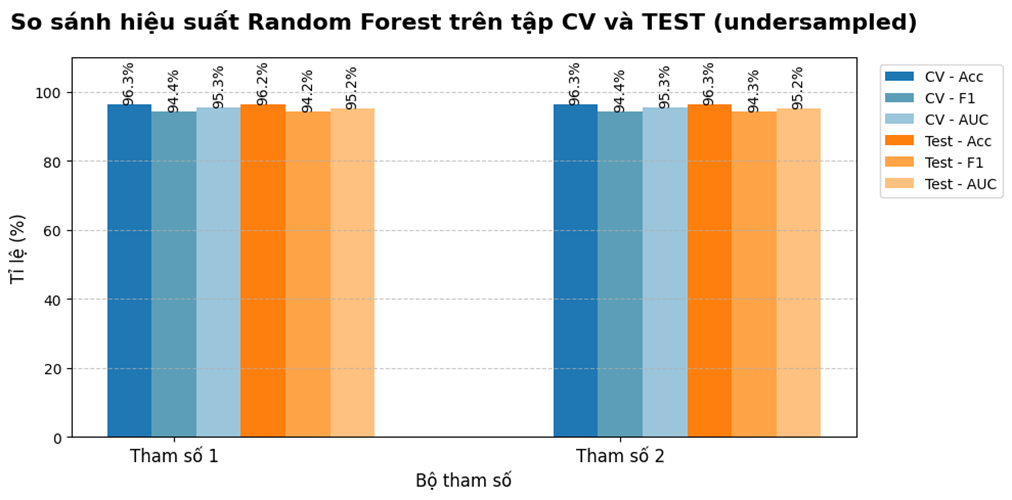
Hình 49. Biểu đồ so sánh hiệu suất KNN trên tập CV và Test (RandomUnderSampler)

**

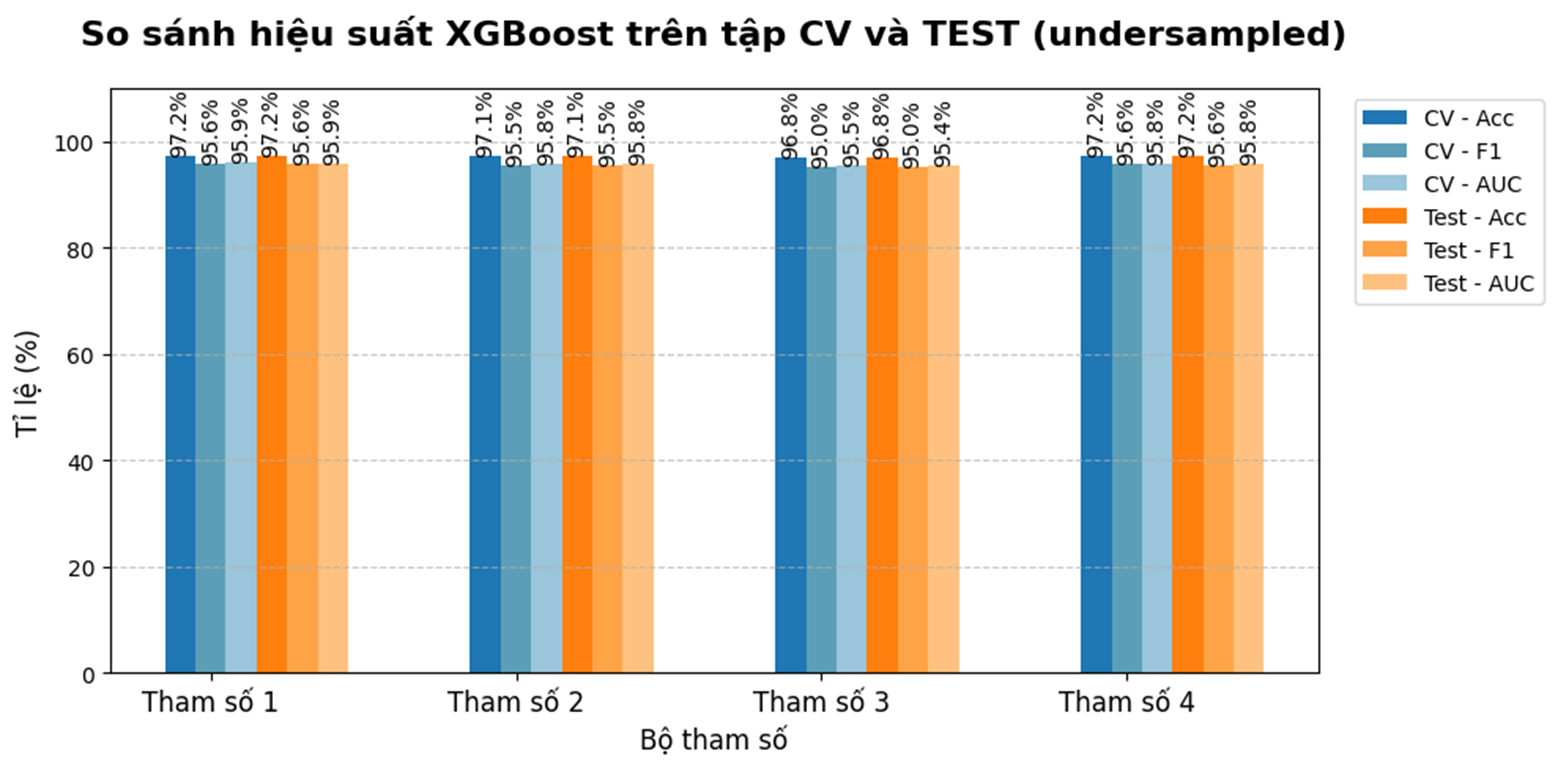
Hình 50. Biểu đồ so sánh hiệu suất GaussianNB trên tập CV và Test (RandomUnderSampler)



Hình 51. Biểu đồ so sánh hiệu suất DecisionTree trên tập CV và Test (RandomUnderSampler)

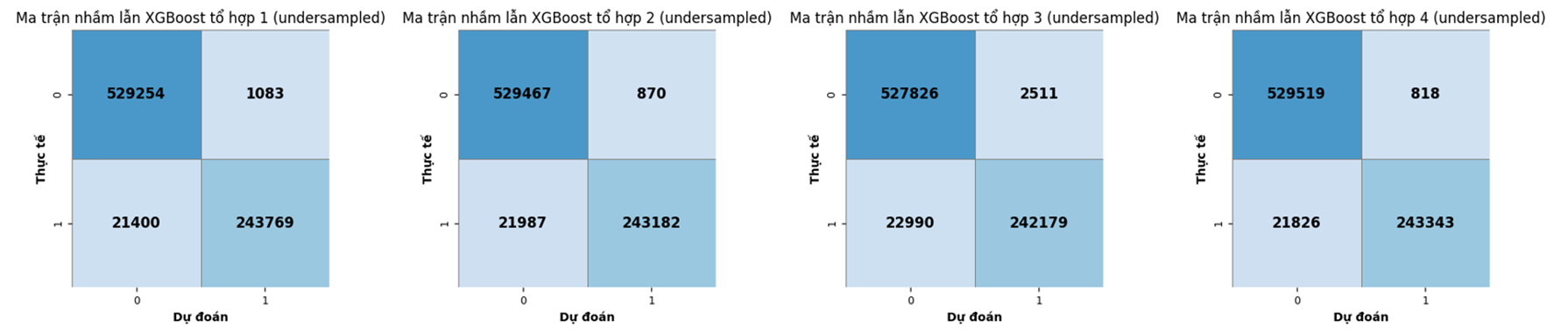
**

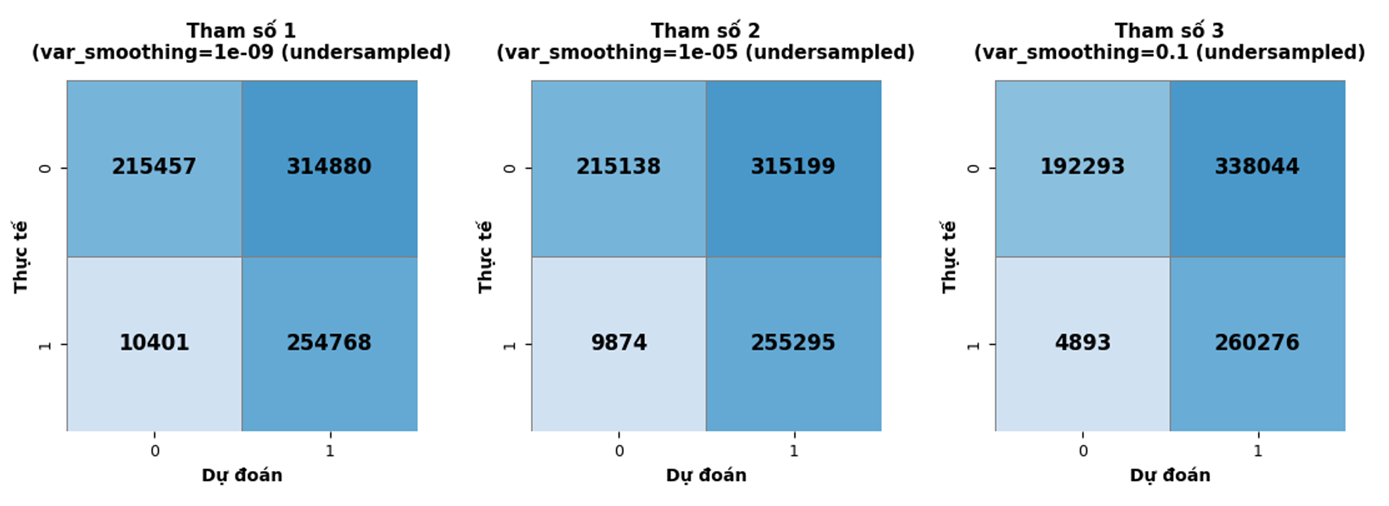
Hình 52. Biểu đồ so sánh hiệu suất RandomForest trên tập CV và Test (RandomUnderSampler)

**

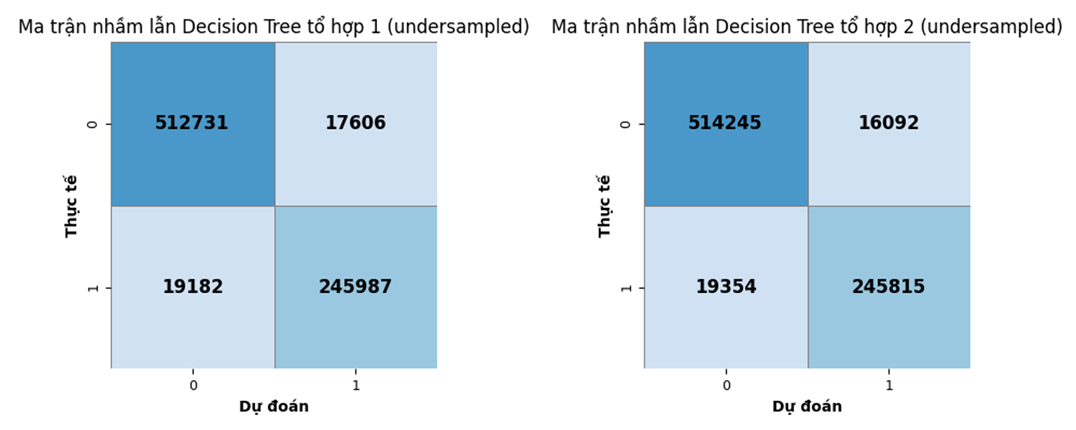
Hình 53. Biểu đồ so sánh hiệu suất XGBoost trên tập CV và Test (RandomUnderSampler)

**

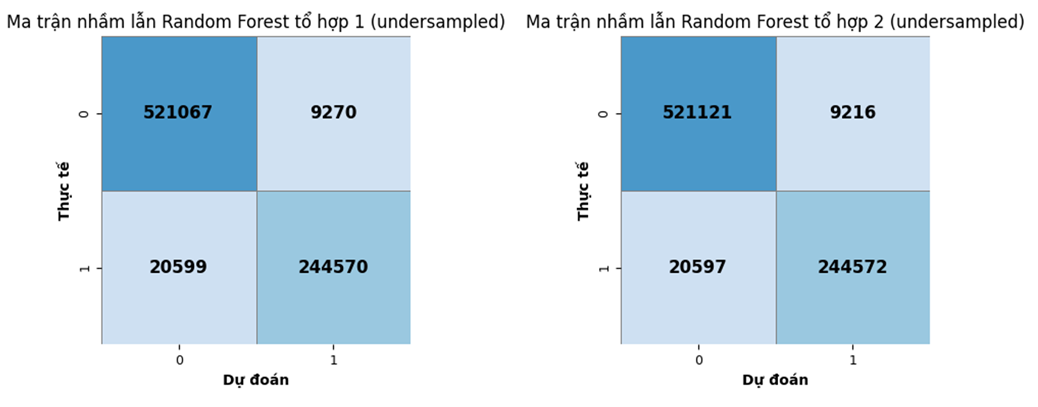
Hình 54. Ma trận nhầm lẫn KNN (RandomUnderSampler)



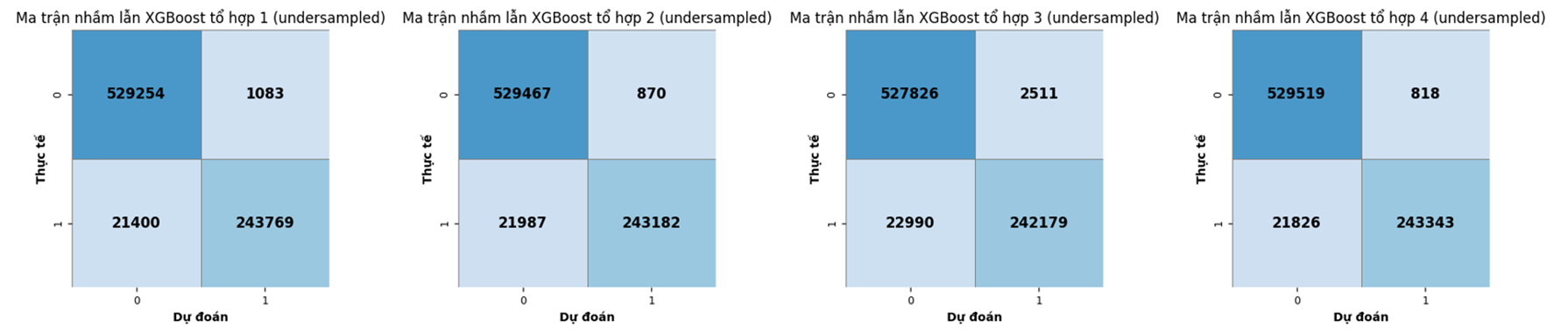
Hình 55. Ma trận nhầm lẫn GaussianNB (RandomUnderSampler)

**

Hình 56. Ma trận nhầm lẫn DecisionTree (RandomUnderSampler)

**

Hình 57. Ma trận nhầm lẫn RandomForest (RandomUnderSampler)

**

Hình 58. Ma trận nhầm lẫn XGBoost (RandomUnderSampler)

Sau khi áp dụng Random Undersampling để cân bằng số lượng giữa hai lớp, các mô hình nhìn chung vẫn giữ hiệu suất ổn định với Accuracy và AUC cao – đặc biệt là các mô hình mạnh như XGBoost, Random Forest. Hiệu quả của các mô hình không giảm so với tập chưa cân bằng. Các mô hình vẫn duy trì khả năng phân loại tốt ngay cả khi bị giảm lượng dữ liệu.

Tuy nhiên, GaussianNB vẫn là ngoại lệ, mặc dù Recall cao, nhưng Precision vẫn thấp, tương tự như các tập trước. Điều này cho thấy mô hình này không phù hợp với dữ liệu IDS dù đã cân bằng bằng bất kỳ phương pháp nào.

So với SMOTE, Random Undersampling mang lại kết quả khá tương đương ở các mô hình mạnh, nhưng có thể dễ triển khai và ít rủi ro hơn về việc sinh mẫu giả không sát thực tế. Ngoài ra, thời gian huấn luyện giảm đáng kể do kích thước dữ liệu nhỏ lại.

Tóm lại, undersampling vẫn giữ được hiệu quả tổng thể của mô hình và có thể xem là một lựa chọn đơn giản nhưng hiệu quả, miễn là không quá làm mất đi dữ liệu quan trọng. Trong khi đó, các mô hình như XGBoost và RF tiếp tục chứng minh sự ổn định và hiệu suất cao nhất qua mọi tình huống.

**Tổng kết chương 4:** Trong chương này, đề tài đã trình bày chi tiết kết quả thực nghiệm khi áp dụng các mô hình học máy truyền thống để phát hiện tấn công mạng sử dụng tập dữ liệu CSE-CIC-IDS2018. Các mô hình được thử nghiệm trong ba trường hợp khác nhau: dữ liệu mất cân bằng, sau khi cân bằng bằng SMOTE và sau khi giảm mẫu bằng undersampling. Việc đánh giá được thực hiện bằng cả Cross-Validation và trên tập test, giúp so sánh rõ ràng hiệu quả học và khả năng dự đoán thực tế của từng mô hình.

Từ những kết quả trên, em có thêm cơ sở để đưa ra đánh giá tổng quan và đề xuất hướng cải tiến phù hợp trong chương sau.

# CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Sau khi thực hiện đầy đủ các bước từ thu thập, xử lý dữ liệu đến xây dựng và đánh giá các mô hình học máy, đề tài đã cho cái nhìn tổng quan về hiệu quả của từng mô hình trong bài toán phát hiện xâm nhập mạng trên tập dữ liệu CSE-CIC-IDS2018. Trong chương này, đề tài sẽ tổng kết lại những kết quả đạt được, rút ra một số nhận xét chung và đề xuất các hướng phát triển tiềm năng để hoàn thiện hơn trong tương lai.

## 5.1 Kết luận

Sau khi thử nghiệm các mô hình trên cả ba phiên bản dữ liệu (mất cân bằng, SMOTE và RandomUnderSampler), đề tài rút ra một số nhận xét tổng quát như sau:

* Các mô hình mạnh như Random Forest và XGBoost cho kết quả ổn định và tốt nhất. Dù xử lý dữ liệu theo cách nào, hai mô hình này đều đạt Accuracy cao, các chỉ số Precision, Recall, F1-score và AUC đều rất ấn tượng. Tuy nhiên, mô hình XGBoost tối ưu hơn rất nhiều nếu không muốn nói là vượt trội về thời gian thực thi cũng như kích cỡ mô hình so với RF.
* KNN cũng hoạt động khá tốt, nhất là khi dùng mặc định, nhưng kích cỡ mô hình và thời gian thực không tối ưu bằng so với mặt bằng nên cũng đáng cân nhắc khi sử dụng.
* Decision Tree thể hiện tốt với độ chính xác và tốc độ xử lý vô cùng ổn định, mô hình khá nhẹ nên là một lựa chọn cân bằng giữa hiệu năng và độ đơn giản.
* Trong khi đó, Gaussian Naive Bayes là mô hình cho kết quả thấp nhất. Dù Recall cao nhưng Precision rất thấp, dẫn đến nhiều cảnh báo sai. Nhìn vào ma trận nhầm lẫn, có thể thấy GNB thường dự đoán lệch về phía tấn công.

Về phía xử lý dữ liệu mất cân bằng:

* Tập gốc (Imbalanced) tuy mất cân bằng thật, nhưng các mô hình mạnh vẫn học được khá tốt.
* SMOTE giúp tăng Accuracy, Precision và AUC ở một số mô hình, nhưng cũng làm tăng sai sót ở lớp tấn công trong vài trường hợp.
* Undersampling giúp giảm thời gian huấn luyện, vẫn duy trì hiệu quả cao, nhưng có nguy cơ làm mất dữ liệu quan trọng.

Tóm lại, XGBoost là lựa chọn tốt nhất, RF cũng tương đối tối ưu, DT cũng là một lựa chọn tốt, còn GaussianNB có vẻ không phù hợp. Việc xử lý mất cân bằng cần được lựa chọn tùy bài toán cụ thể, và nên đánh giá kỹ bằng cả chỉ số lẫn ma trận nhầm lẫn để đảm bảo không bỏ sót các nhãn quan trọng như tấn công.

## 5.2 Hạn chế và hướng phát triển

## 5.2.1 Hạn chế

Mặc dù đề tài đã đạt được một số kết quả khả quan, nhưng vẫn còn một vài hạn chế nhất định.

* Do giới hạn thời gian nên chỉ sử dụng một tập dữ liệu là CSE-CIC-IDS2018
* Tiền xử lý và huấn luyện mô hình mỗi bước dùng 1 file riêng biệt để dễ bảo trì và linh hoạt, nhưng phải khai báo lặp lại thư viện ở nhiều nơi, và khó thực hiện huấn luyện tự động toàn bộ pipeline.
* Tuy có đánh giá tốc độ và dung lượng mô hình, nhưng đề tài chưa triển khai thử nghiệm trong môi trường IDS thực tế nên chưa thể kết luận rõ về hiệu quả khi ứng dụng thật.

## 5.2.2 Hướng phát triển

Trong tương lai, đề tài có thể được mở rộng theo một số hướng như:

* Ứng dụng mô hình học sâu (Deep Learning) như LSTM, CNN để cải thiện độ chính xác và phát hiện các tấn công phức tạp hơn.
* Tối ưu quy trình huấn luyện, xây dựng pipeline tự động thay vì nhiều file rời, giúp dễ bảo trì và mở rộng.
* Thử nghiệm trên tập dữ liệu khác như CIC-IDS2017 hoặc NSL-KDD để kiểm tra tính ổn định của mô hình.
* Kết hợp thêm kỹ thuật xử lý mất cân bằng chẳng hạn như ADASYN hoặc học có trọng số để nâng cao hiệu quả phân loại.
* Triển khai thử nghiệm trong môi trường thực tế, đánh giá khả năng phát hiện tấn công trực tiếp trên mạng giả lập.

# TÀI LIỆU THAM KHẢO

Songma, S.; Sathuphan, T.; Pamutha, T. Optimizing Intrusion Detection Systems in Three Phases on the CSE-CIC-IDS-2018 Dataset. Computers 2023, 12, 245.

https://[doi.org/10.3390/computers12120245](http://doi.org/10.3390/computers12120245)

Leevy, J.L., Khoshgoftaar, T.M. A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data. *J Big Data* 7, 104 (2020). <https://doi.org/10.1186/s40537-020-00382-x>

G. Karatas, O. Demir and O. K. Sahingoz, "Increasing the Performance of Machine Learning-Based IDSs on an Imbalanced and Up-to-Date Dataset", in *IEEE Access*, vol. 8, pp. 32150-32162, 2020, doi: 10.1109/ACCESS.2020.2973219.

Chimphlee, W., & Chimphlee, S. (2024), “Hyperparameters optimization XGBoost for network intrusion detection using CSE-CIC-IDS 2018 dataset”, in *IAES International Journal of Artificial Intelligence (IJ-AI)*, \*13\*(1), 817–826. <https://doi.org/10.11591/ijai.v13.i1.pp817-826>.

K. Bong and J. Kim, "Analysis of Intrusion Detection Performance by Smoothing Factor of Gaussian NB Model Using Modified NSL-KDD Dataset," *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*, Jeju Island, Korea, Republic of, 2022, pp. 1471-1476, doi: 10.1109/ICTC55196.2022.9952381.

Huang, J., Wei, Y., Yi, J., & Liu, M. (2018). An improved kNN based on class contribution and feature weighting. *2018 10th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*. <https://doi.org/10.1109/ICMTMA.2018.00083>