**RANDOM FOREST**

**Các bước làm:**

1. **Chia dữ liệu:** Sử dụng train\_test\_split để chia dữ liệu thành tập huấn luyện (80%) và tập kiểm tra (20%).

A screen shot of a computer program

Description automatically generated

**Giải thích:**

Lưu các giá trị

* **x**: Chứa giá trị các cột quyền của ứng dụng (các thuộc tính), mà mô hình học máy sẽ sử dụng để phân loại ứng dụng là độc hại hay không độc hại
* **y**: Chứa giá trị cột Label

Chia dữ liệu theo tỷ lệ 80-20 và lưu vào các biến

* **X\_train**: Chứa dữ liệu từ X, loại bỏ cột mục tiêu (Label). Nếu X có 1000 mẫu, thì X\_train sẽ có khoảng 800 mẫu (80% của 1000). Đây là tập hợp các biến đầu vào (features) được sử dụng để huấn luyện mô hình.
* **X\_test**: Chứa dữ liệu từ X, nhưng chiếm khoảng 20% tổng số mẫu. Nếu X có 1000 mẫu, thì X\_test sẽ có khoảng 200 mẫu. Đây là tập hợp các biến đầu vào (features) được sử dụng để kiểm tra mô hình sau khi đã huấn luyện.
* **y\_train**: Chứa giá trị từ y, là cột mục tiêu (Label). Nếu y có 1000 mẫu, thì y\_train sẽ có khoảng 800 giá trị (80% của 1000). Đây là tập hợp các giá trị mục tiêu (labels) tương ứng với các mẫu trong X\_train, dùng để huấn luyện mô hình.
* **y\_test**: Chứa giá trị từ y, nhưng chiếm khoảng 20% tổng số mẫu. Nếu y có 1000 mẫu, thì y\_test sẽ có khoảng 200 giá trị. Đây là tập hợp các giá trị mục tiêu (labels) tương ứng với các mẫu trong X\_test, dùng để đánh giá mô hình.

1. **Khởi tạo và huấn luyện mô hình:** Sử dụng RandomForestClassifier từ sklearn.ensemble để khởi tạo mô hình và huấn luyện với dữ liệu huấn luyện.

A screenshot of a computer program

Description automatically generated

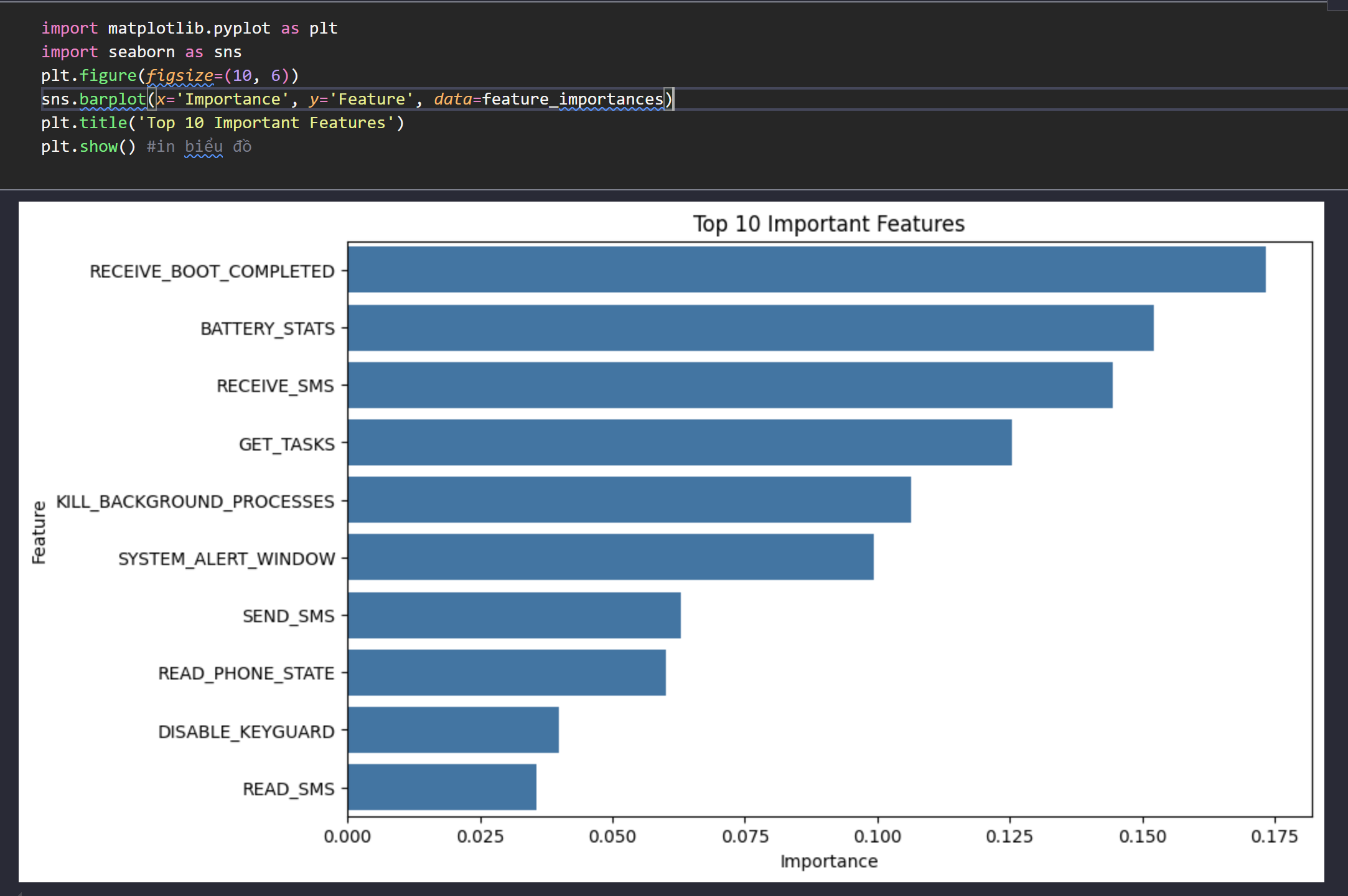
**Giải thích:**

Tạo một đối tượng mô hình Random Forest (model)

* **n\_estimators=100:** Chỉ định số lượng cây quyết định trong mô hình. 100 cây sẽ được sử dụng trong quá trình huấn luyện.
* **random\_state=42**: Thiết lập giá trị ngẫu nhiên để đảm bảo kết quả mô hình có thể tái tạo được

1. **Biểu đồ cột ngang**

Mô hình tính toán **mức độ quan trọng** của các đặc trưng (features) để xác định những đặc trưng nào có ảnh hưởng lớn nhất đến kết quả dự đoán sau khi huấn luyện



**sns.barplot(x='Importance',y='Feature',data=feature\_importances):**Dòng này tạo biểu đồ cột ngang sử dụng hàm barplot của Seaborn

* **x='Importance'**: Trục x đại diện cho "Mức độ quan trọng" của các đặc trưng (thường là các giá trị số cho biết mức độ đóng góp của đặc trưng vào dự đoán của mô hình).
* **y='Feature'**: Trục y đại diện cho tên của các đặc trưng.
* **data=feature\_importances**: DataFrame feature\_importances chứa dữ liệu, trong đó một cột là "Importance" và một cột khác là "Feature."

1. **Dự đoán:** Dự đoán trên tập kiểm tra và sử dụng các metric như độ chính xác (accuracy) và báo cáo phân loại (classification\_report) để đánh giá mô hình.

A computer screen with text

Description automatically generated

**Giải thích:**

* **model.predict(X\_test)**:Dùng để dự đoán nhãn (label) cho các mẫu trong tập kiểm tra (X\_test). Kết quả sẽ là một mảng chứa các giá trị dự đoán cho mỗi mẫu trong tập kiểm tra (y\_pred).
* **accuracy\_score(y\_test, y\_pred)**: Tính toán độ chính xác bằng cách so sánh nhãn thực tế (y\_test) với nhãn dự đoán (y\_pred). Giá trị trả về sẽ là một số giữa 0 và 1, cho biết tỷ lệ dự đoán đúng (accuracy).
* **print(f'Accuracy: {accuracy \* 100:.2f}%')**: In ra độ chính xác dưới dạng phần trăm (sử dụng \* 100 để chuyển đổi từ tỷ lệ sang phần trăm và :.2f để giới hạn số thập phân)
* **print(classification\_report(y\_test, y\_pred))**: classification\_report(y\_test, y\_pred): In ra một báo cáo chi tiết về hiệu suất của mô hình phân loại

**Kết quả**

A screenshot of a computer screen

Description automatically generated

1. **Biểu đồ:**

**1. Biểu đồ cột nhóm** **(grouped bar chart):** được sử dụng để so sánh ba chỉ số phân loại quan trọng—**Precision** (độ chính xác), **Recall** (độ nhạy), và **F1-Score**—cho từng lớp trong một bài toán phân loại.

A graph of different colored bars

Description automatically generated

**2. Biểu đồ ma trận nhầm lẫn (confusion matrix heatmap):** Biểu đồ giúp hiểu rõ mô hình phân loại hoạt động tốt như thế nào, dễ dàng quan sát số lượng dự đoán đúng và sai cho

từng lớp thông qua các ô trong ma trận. Các số liệu này giúp đánh giá mức độ chính xác, nhạy và lỗi của mô hình, từ đó có cái nhìn trực quan về hiệu suất tổng thể.

A yellow and purple squares

Description automatically generated

* **True Positive (TP):** Dự đoán là 1 và thực tế cũng là 1 (mô hình dự đoán đúng nhãn 1) (hàng dưới bên phải).
* **True Negative (TN):** Dự đoán là 0 và thực tế cũng là 0 (mô hình dự đoán đúng nhãn 0). (hàng trên bên trái).
* **False Positive (FP):** Dự đoán là 1, nhưng thực tế là 0 (mô hình dự đoán nhầm nhãn 1). (hàng trên bên phải).
* **False Negative (FN):** Dự đoán là 0, nhưng thực tế là 1 (mô hình dự đoán nhầm nhãn 0). (hàng dưới bên trái)

3. **Biểu đồ** **đường cong ROC (Receiver Operating Characteristic curve)** kết hợp với giá trị **AUC (Area Under the Curve)**. Nó giúp đánh giá hiệu suất của mô hình phân loại, trong trường hợp này là **RandomForestClassifier**, bằng cách so sánh tỷ lệ dương tính thật (True Positive Rate - TPR) với tỷ lệ dương tính giả (False Positive Rate - FPR). AUC càng cao thì hiệu suất của mô hình càng tốt trong việc phân biệt giữa các lớp.

A graph with a line and a blue line

Description automatically generated

**FPR (False Positive Rate - Tỷ lệ dương tính giả):**

* Đây là tỷ lệ mà mô hình dự đoán **dương tính** (positive) khi thực tế là **âm tính** (negative).
* Trục x biểu diễn FPR.

**TPR (True Positive Rate - Tỷ lệ dương tính thật hay độ nhạy):**

* Đây là tỷ lệ mà mô hình dự đoán **dương tính** (positive) khi thực tế cũng là **dương tính**.
* Trục y biểu diễn TPR.

**Đường cong ROC:**

* Đường cong này biểu diễn sự đánh đổi giữa **FPR** và **TPR** ở các ngưỡng khác nhau.
* Một mô hình tốt sẽ có đường ROC đi gần tới góc trên bên trái, biểu thị TPR cao và FPR thấp.

**AUC (Area Under the Curve):**

* **AUC** là diện tích dưới đường ROC. Giá trị **AUC** dao động từ 0.5 đến 1:
* **AUC = 1**: Mô hình hoàn hảo.
* **AUC = 0.5**: Mô hình ngẫu nhiên, không tốt hơn so với đoán mò.
* **AUC càng gần 1**: Mô hình càng tốt.

Trong biểu đồ, **AUC** được hiển thị trong nhãn chú thích (legend).

**Đường màu xanh (đường chéo [0, 1]):**

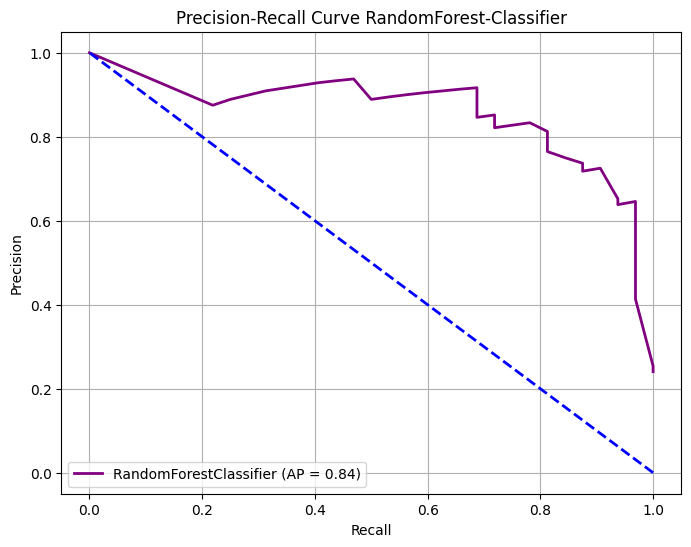
* Đường này đại diện cho mô hình dự đoán ngẫu nhiên, có AUC = 0.5. Nếu đường ROC của mô hình ở phía trên đường này, nghĩa là mô hình tốt hơn ngẫu nhiên.

**4. Biểu đồ đường cong Precision-Recall (Precision-Recall curve)**

**-** Đánh giá mối quan hệ giữa **Precision** và **Recall**. Khi điều chỉnh ngưỡng dự đoán của mô hình, **Precision** và **Recall** sẽ thay đổi, và đường cong này giúp thấy được sự đánh đổi giữa hai chỉ số này.

- Đặc biệt, trong các trường hợp dữ liệu mất cân bằng, Precision-Recall curve thường cung cấp cái nhìn chi tiết và hữu ích hơn so với đường ROC.

Một mô hình lý tưởng sẽ có cả Precision và Recall đều cao, được biểu diễn bởi một đường cong tiến gần tới góc trên bên phải của biểu đồ.



**Precision (Độ chính xác):**

* Tỷ lệ các mẫu dự đoán là dương tính (positive) mà thực sự đúng.
* Precision cao nghĩa là trong các dự đoán dương tính của mô hình, phần lớn là đúng.
* Precision được hiển thị trên **trục y**.

**Recall (Độ nhạy):**

* Tỷ lệ các mẫu thực sự dương tính mà mô hình dự đoán đúng.
* Recall cao nghĩa là mô hình đã bắt được phần lớn các trường hợp dương tính.
* Recall được hiển thị trên **trục x**.

**Đường cong Precision-Recall:**

* Đường này biểu thị mối quan hệ giữa Precision và Recall tại các ngưỡng dự đoán khác nhau.
* Một mô hình tốt sẽ có đường cong nằm gần góc trên bên phải của biểu đồ, thể hiện mức độ Precision và Recall đều cao.

**Average Precision (AP):**

* **AP** là một giá trị tóm tắt hiệu suất của mô hình trên toàn bộ các ngưỡng dự đoán, tính toán dựa trên diện tích dưới đường Precision-Recall.
* **AP càng cao** (gần 1) thì mô hình càng tốt. Trong biểu đồ này, AP được hiển thị trong chú thích (legend).

**Đường màu xanh chéo ([1, 0] đến [0, 1]):**

* Đây là đường biểu diễn cho mô hình phân loại ngẫu nhiên, không tốt hơn việc đoán mò.

**6. Đánh giấ hiệu năng**

**1. Độ chính xác (Accuracy)**

Accuracy: 89.47% của 133 mẫu trong tập kiểm thử. Cho thấy mô hình đã dự đoán chính xác khoảng 89.47% tổng số mẫu trong tập kiểm tra. Đây là một chỉ số tổng quát cho thấy mô hình hoạt động tốt, nhưng không đủ để đánh giá toàn diện hiệu suất.

**2. Precision, Recall, F1-Score**

Ba chỉ số này giúp đánh giá hiệu suất mô hình một cách chi tiết hơn đối với từng lớp (class). Lớp 0 đại diện cho "không độc hại" và lớp 1 đại diện cho "độc hại".

**Lớp 0.0 (Ứng dụng không độc hại)**:

* **Precision** (Độ chính xác): 0.91

Trong tất cả các mẫu mà mô hình dự đoán là lớp 0 (không độc hại), có 91% là chính xác (thực sự thuộc lớp 0).

* **Recall** (Độ nhạy): 0.96

Trong tất cả các mẫu thực sự thuộc lớp 0, mô hình đã nhận diện đúng 96% số mẫu.

* **F1-Score**: 0.93

F1-score là sự kết hợp giữa precision và recall, với giá trị 0.93 cho thấy mô hình hoạt động tốt trong việc phân loại các mẫu thuộc lớp 0.

* **Support**: 101

Trong tập dữ liệu kiểm thử, có 101 mẫu thực sự thuộc lớp 0 (không độc hại).

**Lớp 1.0 (Ứng dụng độc hại)**:

* **Precision**: 0.85

Trong tất cả các mẫu mà mô hình dự đoán là lớp 1 (độc hại), có 85% là chính xác.

* **Recall**: 0.69

Trong tất cả các mẫu thực sự thuộc lớp 1, mô hình nhận diện đúng 69% số mẫu. Giá trị này tương đối thấp, cho thấy mô hình bỏ lỡ một số mẫu thuộc lớp 1.

* **F1-Score**: 0.76

F1-score là sự kết hợp của precision và recall, và giá trị 0.76 cho thấy mô hình có thể chưa tối ưu trong việc phát hiện các ứng dụng độc hại.

* **Support**: 32

Có 32 mẫu trong tập kiểm thử thực sự thuộc lớp 1 (độc hại).

3. **Macro Average (macro avg):**

Macro average tính toán giá trị trung bình của precision, recall và F1-score của 2 lớp 0(không độc hại) và 1(độc hại) mà không xét đến số lượng mẫu trong mỗi lớp.

* **Precision**: 0.88
* **Recall**: 0.82
* **F1-score**: 0.85

Điều này thể hiện hiệu suất tổng quan của mô hình đối với cả hai lớp 0 và 1.

4. **Weighted Average (weighted avg)**

Tính toán các chỉ số (precision, recall, F1-score) dựa trên tỷ lệ số lượng mẫu trong mỗi lớp

* **Precision** (0.89): Được tính bằng cách nhân precision của mỗi lớp với số lượng mẫu tương ứng của lớp đó, rồi chia cho tổng số mẫu. Do lớp 0 có nhiều mẫu hơn, nên precision của lớp 0 có ảnh hưởng lớn hơn đến weighted average.
* **Recall** (0.89): Tương tự, recall của lớp 0 và lớp 1 được nhân với số lượng mẫu của từng lớp rồi chia cho tổng số mẫu.
* **F1-score** (0.89): Cũng được tính tương tự dựa trên số lượng mẫu của từng lớp.

Vì lớp 0 có nhiều mẫu hơn, nên các chỉ số của lớp này đóng góp nhiều hơn vào weighted avg. Nhờ đó, weighted avg phản ánh tốt hơn hiệu suất tổng thể của mô hình trên cả hai lớp, không để mô hình chỉ đạt kết quả tốt vì tập trung quá nhiều vào lớp có nhiều mẫu (lớp 0), mà còn được đánh giá đúng trên cả hai lớp, dù số lượng mẫu chênh lệch (chỉ giỏi phân loại một lớp mà kém phân loại lớp kia).

**Tóm lại:**

* Mô hình có độ chính xác tổng thể cao (**89.47%**).
* Mô hình phân loại tốt các ứng dụng không độc hại (lớp 0) với **precision** và **recall** cao.
* Tuy nhiên, đối với các ứng dụng độc hại (lớp 1), mô hình có thể bỏ lỡ một số mẫu, do **recall** chỉ đạt 69%. Điều này có thể yêu cầu điều chỉnh mô hình để cải thiện khả năng nhận diện các ứng dụng độc hại.

**7. Tinh chỉnh mô hình :** Sử dụng GridSearchCV để tìm kiếm các tham số tốt nhất cho mô hình.

A screenshot of a computer program

Description automatically generated

**Giải thích:**

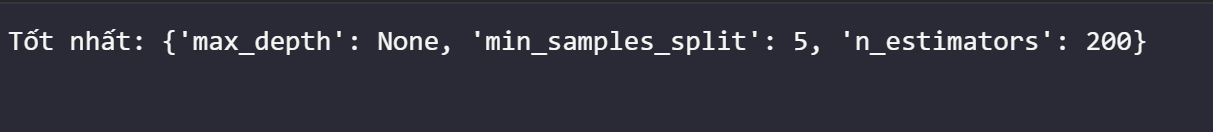
**param\_grid:**

* **n\_estimators**: Số lượng cây trong mô hình Random Forest (100 và 200 cây).
* **max\_depth**: Độ sâu tối đa của mỗi cây (không giới hạn (None), 10, 20 và 30).
* **min\_samples\_split**: Số mẫu tối thiểu cần thiết để chia một nút (2, 5 và 10).

**Grid\_search**:

* **param\_grid=param\_grid**: Tham số này chỉ định cho GridSearchCV biết các tham số nào sẽ được tối ưu hóa và các giá trị nào sẽ được thử nghiệm cho mỗi tham số.
* **cv = 3**: Tham số này xác định số lần chia dữ liệu huấn luyện (sử dụng phương pháp k-fold cross-validation với 3 folds ) để kiểm tra mô hình. Mỗi fold sẽ được sử dụng để kiểm tra mô hình trong khi các fold còn lại được sử dụng để huấn luyện.

**Kết quả**



Kết quả này cho thấy rằng việc tối ưu hóa các tham số thông qua GridSearchCV đã giúp tìm ra sự kết hợp tốt nhất cho mô hình:

* Không giới hạn độ sâu của cây.
* Yêu cầu tối thiểu 5 mẫu để chia một nút.
* Sử dụng 200 cây trong mô hình Random Forest.

1. **Lưu mô hình:** Sử dụng joblib để lưu mô hình đã huấn luyện vào file để sử dụng sau này.

A screenshot of a computer program

Description automatically generated