**Giải thích chi tiết từng dòng code trong demo phát hiện website phishing bằng Học máy**

**1. Tải dữ liệu:**

Python

import pandas as pd

# Đọc dữ liệu từ file CSV

data = pd.read\_csv('urldata.csv')

* Dòng đầu tiên import thư viện pandas để thao tác với dữ liệu dạng bảng.
* Dòng thứ hai sử dụng hàm pd.read\_csv() để đọc dữ liệu từ file urldata.csv và lưu trữ vào biến data. Giả sử file này chứa các cột như Domain, URL, Features1, Features2, ..., Label.

**2. Mã hóa biến mục tiêu:**

Python

from sklearn.preprocessing import LabelEncoder

# Mã hóa biến mục tiêu 'Label' (ví dụ: phishing = 0, legitimate = 1)

label\_encoder = LabelEncoder()

data['Label'] = label\_encoder.fit\_transform(data['Label'])

* Dòng đầu tiên import thư viện sklearn.preprocessing để sử dụng bộ mã hóa LabelEncoder.
* Dòng thứ hai tạo biến label\_encoder để mã hóa nhãn "Label".
* Dòng thứ ba thay thế giá trị trong cột "Label" bằng mã số tương ứng (ví dụ: phishing = 0, legitimate = 1).

**3. Chia tách dữ liệu:**

Python

from sklearn.model\_selection import train\_test\_split

# Chia tách dữ liệu thành tập huấn luyện (X\_train, y\_train) và tập kiểm tra (X\_test, y\_test)

X = data.drop(['Domain', 'Label'], axis=1) # X là ma trận chứa các đặc trưng

y = data['Label'] # y là vector chứa nhãn (mã hóa)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

* Dòng đầu tiên import thư viện sklearn.model\_selection để chia tách dữ liệu.
* Dòng thứ hai tạo ma trận X chứa tất cả các cột trong data **trừ** Domain và Label. Ma trận này chứa các đặc trưng sẽ được sử dụng để huấn luyện mô hình.
* Dòng thứ ba tạo vector y chứa các giá trị nhãn (mã hóa) tương ứng từ cột Label.
* Dòng thứ tư sử dụng hàm train\_test\_split() để chia tách X và y thành tập huấn luyện (70%) và tập kiểm tra (30%). Hàm này cũng sử dụng random\_state=42 để đảm bảo tính ngẫu nhiên trong việc chia tách.

**4. Huấn luyện mô hình:**

Python

from sklearn.ensemble import RandomForestClassifier

# Huấn luyện mô hình Random Forest Classifier

rf\_classifier = RandomForestClassifier(n\_estimators=100, random\_state=42)

rf\_classifier.fit(X\_train, y\_train)

* Dòng đầu tiên import thư viện sklearn.ensemble để sử dụng mô hình RandomForestClassifier.
* Dòng thứ hai tạo mô hình rf\_classifier với 100 cây quyết định (n\_estimators=100) và random\_state=42 để đảm bảo tính ngẫu nhiên.
* Dòng thứ ba huấn luyện mô hình rf\_classifier sử dụng dữ liệu huấn luyện X\_train và y\_train.

**5. Đánh giá mô hình:**

Python

from sklearn.metrics import classification\_report, accuracy\_score

# Đánh giá mô hình trên tập kiểm tra

y\_pred = rf\_classifier.predict(X\_test)

print("Độ chính xác:", accuracy\_score(y\_test, y\_pred))

print("Báo cáo phân loại:")

print(classification\_report(y\_test, y\_pred))

* Dòng đầu tiên import thư viện sklearn.metrics để đánh giá mô hình.
* Dòng thứ hai sử dụng rf\_classifier.predict(X\_test) để dự đoán nhãn cho tập kiểm tra X\_test. Kết quả dự đoán được lưu trữ trong biến y\_pred.
* Dòng thứ ba tính toán độ chính xác (accuracy) của mô hình bằng cách so sánh y\_pred với y\_test.
* Dòng thứ tư và thứ năm in ra báo cáo phân loại chi tiết, bao gồm độ chính xác cho từng lớp (phishing và legitimate)

## Tiếp tục giải thích chi tiết từng dòng code trong demo phát hiện website phishing bằng Học máy

**6. Xác định tầm quan trọng của đặc trưng:**

Python

# Tính tầm quan trọng của từng đặc trưng

feature\_importance = pd.DataFrame(rf\_classifier.feature\_importances\_, index=X.columns, columns=['importance']).sort\_values('importance', ascending=False)

print("Tầm quan trọng của đặc trưng:")

print(feature\_importance)

* Dòng đầu tiên sử dụng thuộc tính feature\_importances\_ của mô hình rf\_classifier để lấy tầm quan trọng của từng đặc trưng.
* Dòng thứ hai tạo DataFrame feature\_importance với các cột:
  + index: Tên của từng đặc trưng.
  + importance: Giá trị tầm quan trọng tương ứng.
* Dòng thứ ba sắp xếp DataFrame theo giá trị importance (giảm dần) và in ra kết quả.

**7. Hàm dự đoán tính hợp pháp của domain:**

Python

def predict\_domain\_legality(domain):

if domain in data['Domain'].values:

# Lấy thông tin của domain từ data

domain\_row = data[data['Domain'] == domain].drop(['Domain', 'Label'], axis=1)

print("Thông tin domain:")

print(domain\_row)

# Dự đoán domain

prediction = rf\_classifier.predict(domain\_row)

print("Dự đoán:", prediction)

if prediction[0] == 1:

print(f"Domain '{domain}' có khả năng là website phishing.")

else:

print(f"Domain '{domain}' có khả năng là website hợp pháp.")

else:

print(f"Domain '{domain}' không tồn tại trong dữ liệu.")

# Hỏi người dùng nhập domain

domain\_to\_check = input("Nhập domain để kiểm tra tính hợp pháp: ")

# Dự đoán domain

predict\_domain\_legality(domain\_to\_check)

* Dòng đầu tiên định nghĩa hàm predict\_domain\_legality với tham số domain.
* Hàm này kiểm tra xem domain có tồn tại trong dữ liệu data hay không.
* Nếu có:
  + Lấy thông tin của domain từ data và in ra.
  + Sử dụng mô hình rf\_classifier để dự đoán domain là phishing hay legitimate.
  + In ra kết quả dự đoán và giải thích.
* Nếu không:
  + In thông báo domain không tồn tại.
* Cuối cùng, hàm hỏi người dùng nhập domain để kiểm tra và thực hiện dự đoán.

**Kết luận:**

Đoạn code Python này cung cấp một demo cơ bản về việc sử dụng Học máy để phát hiện website phishing. Nó thực hiện các bước như: tải dữ liệu, tiền xử lý dữ liệu, chia tách dữ liệu, huấn luyện mô hình, đánh giá mô hình, xác định tầm quan trọng của đặc trưng và dự đoán tính hợp pháp của domain mới.

**Lưu ý:**

* Hiệu quả của mô hình phụ thuộc vào chất lượng và độ cân bằng của dữ liệu.
* Cần cập nhật dữ liệu thường xuyên để mô hình thích ứng với các kỹ thuật phishing mới.
* Có thể kết hợp thêm các phương pháp học máy khác để nâng cao hiệu quả dự đoán.

**Demo này cung cấp ví dụ cơ bản về ứng dụng Học máy trong việc phát hiện website phishing. Để triển khai thực tế, cần có sự đầu tư vào việc thu thập dữ liệu, huấn luyện và tối ưu hóa mô hình.**