ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

BÁO CÁO ĐỒ ÁN MÔN HỌC

PYTHON CHO KHOA HỌC PHÂN LOẠI KHẢ NĂNG RỦI RO CỦA KHÁCH HÀNG KHI MUA BẢO HIỂM XE

SINH VIÊN THỰC HIỆN:

1. **Nguyễn Ngọc Minh** - MSSV: 23280029

Công việc: khai phá dữ liệu (EDA), viết báo cáo.

2. **Thái Ngọc Thanh Mai** - MSSV: 23280018

Công việc: xây dựng pipeline Tiền xử lý dữ liệu (Data processor)

3. **Nguyễn Đỗ Khánh Ngọc** -MSSV: 23280008

Công việc: xây dựng pipeline Mô hình học máy (Machine learning modeling)

TP Hồ Chí Minh, ngày 10 tháng 12 năm 2025

MỤC LỤC

MỞ ĐẦU

Trong bối cảnh dữ liệu ngày càng phong phú, việc chuyển đổi quy trình thẩm định bảo hiểm từ thủ công sang tự động hóa là một yêu cầu tất yếu để kiểm soát rủi ro và tối đa hóa hiệu suất. Đồ án này là một nghiên cứu Khoa học Dữ liệu (Data Science) chuyên sâu, tập trung vào việc xây dựng và đánh giá các mô hình phân loại nhằm hỗ trợ ra quyết định chấp nhận hay từ chối bảo hiểm xe cơ giới.

Mục tiêu cốt lõi là giải quyết bài toán phân loại nhị phân **OUTCOME**: 1 - Chấp nhận, 0 - Từ chối) bằng cách tận dụng các đặc trưng như hồ sơ rủi ro lái xe, đặc điểm nhân khẩu học và thông tin phương tiện.

Đồ án không chỉ dừng lại ở việc xây dựng một mô hình, mà còn thực hiện một phân tích so sánh toàn diện. Nhóm sẽ triển khai mô hình, huấn luyện và đánh giá nhiều thuật toán phân loại khác nhau (như **Logistic Regression, Decision Trees, Random Forests**,…) để xác định mô hình tối ưu nhất về mặt hiệu suất **Accuracy, Precision, Recall, F1-Score, AUC-ROC.**

Quan trọng hơn, dự án sẽ tập trung vào việc khai thác Insight (Data and Feature Insights):

1. Xác định các Biến Tác động mạnh nhất (Key Feature Importance): Khám phá những yếu tố **nào CREDIT_SCORE, PAST_ACCIDENTS, DRIVING_EXPERIENCE**,...) có ảnh hưởng quyết định nhất đến việc chấp nhận hay từ chối bảo hiểm.
2. Đưa ra Khuyến nghị Dựa trên Dữ liệu: Các insight này sẽ được chuyển hóa thành các khuyến nghị cụ thể, giúp công ty bảo hiểm điều chỉnh các quy tắc thẩm định, chính sách định giá và chiến lược kinh doanh.

Tóm lại: Đồ án là sự kết hợp giữa kỹ thuật xây dựng mô hình và phân tích dữ liệu, không chỉ tạo ra một công cụ hỗ trợ ra quyết định thẩm định hiệu quả mà còn cung cấp cái nhìn sâu sắc dựa trên dữ liệu để quản lý rủi ro chủ động.

GIỚI THIỆU VỀ CÁC MÔ HÌNH PHÂN LOẠI TRONG MACHINE LEARNING

1. Logistic Regression (Hồi quy Logistic)

- **Bản chất**: Là một thuật toán phân loại tuyến tính (Linear Classifier), mặc dù tên gọi có chữ "Regression".
- **Cách hoạt động:** Sử dụng một phương trình tuyến tính kết hợp với hàm Sigmoid (hàm logistic) để ánh xạ đầu ra về một giá trị xác suất nằm trong khoảng . Giá trị này được dùng để quyết định đối tượng thuộc về lớp nào (ví dụ: nếu xác suất > 0.5 thì là lớp 1, ngược lại là lớp 0).
- **Ưu điểm**: Dễ hiểu, dễ cài đặt, tính toán nhanh, cung cấp xác suất đầu ra cụ thể.
- **Nhược điểm:** Chỉ hiệu quả với các vấn đề có ranh giới phân loại tuyến tính (linearly separable).

2. Support Vector Machine (SVM)

- **Bản chất:** Là một mô hình có thể tuyến tính hoặc phi tuyến tính.
- **Cách hoạt động:** Tìm ra một siêu mặt phẳng (hyperplane) tốt nhất để phân chia các lớp dữ liệu. Siêu mặt phẳng này có khoảng cách (margin) lớn nhất đến các điểm dữ liệu gần nó nhất (gọi là Support Vectors).
- **Ưu điểm:** Rất mạnh mẽ với dữ liệu có số chiều cao (high-dimensional data) và cho kết quả chính xác cao, đặc biệt khi sử dụng các kỹ thuật Kernel để xử lý các bài toán phi tuyến tính.
- **Nhược điểm:** Tốn thời gian tính toán với tập dữ liệu lớn và khó diễn giải (interpretability) khi sử dụng Kernel phi tuyến.

3. K-Nearest Neighbors (KNN)

- **Bản chất**: Là thuật toán không tham số (Non-parametric) và "lười" (Lazy Learner – không học một mô hình cố định mà chỉ lưu trữ dữ liệu).
- **Cách hoạt động:** Để phân loại một điểm dữ liệu mới, thuật toán tìm ra điểm dữ liệu gần nhất (nearest neighbors) trong tập huấn luyện và gán nhãn cho điểm mới bằng cách lấy đa số phiếu (majority vote) từ điểm lân cận đó. Khoảng cách thường dùng là Euclidean.
- **Ưu điểm:** Đơn giản, dễ hiểu, hiệu quả cho dữ liệu có ranh giới phức tạp.
- **Nhược điểm:** Tốn bộ nhớ và thời gian tính toán khi tập dữ liệu lớn (do cần tính khoảng cách đến tất cả các điểm), nhạy cảm với dữ liệu nhiễu (outliers) và cần chuẩn hóa dữ liệu trước khi chạy.

4. Decision Tree (Cây Quyết định)

- **Bản chất**: Là một mô hình phân cấp chia dữ liệu thành các tập con đồng nhất.
- **Cách hoạt động:** Xây dựng một cấu trúc giống như cây. Mỗi nút bên trong (internal node) đại diện cho một phép kiểm tra trên một thuộc tính (feature), mỗi nhánh (branch) đại diện cho kết quả của phép kiểm tra, và mỗi nút lá (leaf node) đại diện cho nhãn lớp cuối cùng. Việc phân tách được thực hiện bằng cách tối ưu hóa các tiêu chí như Độ lợi thông tin (Information Gain) hoặc Gini Impurity.

- **Ưu điểm**: Rất dễ diễn giải (White-box model), không cần chuẩn hóa dữ liệu, xử lý tốt cả dữ liệu số và dữ liệu danh nghĩa (categorical).
- **Nhược điểm:** Rất dễ bị quá khớp (overfitting) với tập huấn luyện và có thể tạo ra các cây không cân bằng.

5. Random Forest (Rừng Ngẫu Nhiên)

- **Bản chất:** Là một kỹ thuật Học Hợp Thể (Ensemble Learning) sử dụng phương pháp Bagging (Bootstrap Aggregating).
- **Cách hoạt động:** Xây dựng một tập hợp lớn (hàng trăm, hàng nghìn) các Cây Quyết định độc lập. Mỗi cây được huấn luyện trên một mẫu con ngẫu nhiên của dữ liệu (Bootstrap Sample). Đối với bài toán phân loại, kết quả cuối cùng là đa số phiếu (majority vote) từ tất cả các cây trong rừng.
- **Ưu điểm:** Giảm thiểu đáng kể hiện tượng quá khớp của Decision Tree, độ chính xác cao, và có khả năng ước tính tầm quan trọng của các thuộc tính (feature importance).
- **Nhược điểm:** Tốc độ chậm hơn Decision Tree và khó diễn giải hơn (Black-box model).

6. XGBoost (Extreme Gradient Boosting)

- **Bản chất:** Là một kỹ thuật Học Hợp Thể (Ensemble Learning) sử dụng phương pháp Gradient Boosting.
- **Cách hoạt động:** Xây dựng các Cây Quyết định một cách tuần tự (sequential). Mỗi cây mới được thêm vào để sửa chữa những sai sót/lỗi (residuals) mà các cây trước đó mắc phải. XGBoost là phiên bản tối ưu hóa cao của Gradient Boosting.
- **Ưu điểm:** Độ chính xác vượt trội trong hầu hết các bài toán dữ liệu có cấu trúc (tabular data), rất hiệu quả và có khả năng xử lý song song.
- **Nhược điểm:** Khó hiểu và khó diễn giải, dễ bị quá khớp nếu các siêu tham số (hyperparameters) không được điều chỉnh (tuning) cẩn thận.

CHƯƠNG 1: KHAI PHÁ – TỔNG QUAN VỀ BỘ DỮ LIỆU (EDA)

I. Tổng quan bộ dữ liệu đầu vào:

Bộ dữ liệu gồm **105017** bản ghi với **18** features bao gồm được định nghĩa như sau:

- **ID**: Một mã định danh duy nhất cho mỗi bản ghi (cá nhân).
- **AGE**: Độ tuổi của cá nhân (ví dụ: 26-39, 40-64,...).
- **GENDER**: Giới tính của cá nhân (ví dụ: 'Nam', 'Nữ', 'Khác').
- **DRIVING_EXPERIENCE**: Kinh nghiệm lái xe (tính bằng năm hoặc theo nhóm như '0-9 năm', '10-19 năm',...).
- **EDUCATION**: Trình độ học vấn của cá nhân (ví dụ: 'Trung học', 'Đại học',...).
- **INCOME**: Mức thu nhập, có thể là số hoặc được phân loại:

  + **Poverty**: Cá nhân hoặc gia đình sống dưới mức sống tối thiểu; thu nhập rất thấp hoặc không có.

  + **Working Class**: Những người có thu nhập thấp đến trung bình, thường làm các công việc lao động chân tay hoặc dịch vụ.

  + **Middle Class**: Những người có thu nhập ổn định, thường là các chuyên gia được hưởng lương, đủ khả năng chi trả các nhu cầu cơ bản và có một khoản tiết kiệm.

  + **Upper Class**: Những cá nhân giàu có, thường sở hữu các tài sản hoặc doanh nghiệp đáng kể.
- **CREDIT_SCORE**: Điểm tín dụng tài chính – một chỉ số về độ tin cậy tín dụng.
- **VEHICLE_OWNERSHIP**: Cá nhân có sở hữu phương tiện hay không (1 = Có, 0 = Không).
- **VEHICLE_YEAR**: Năm chiếc xe được sản xuất, hoặc được nhóm lại (ví dụ: 'Trước 2015', 'Sau 2015').
- **MARRIED**: Tình trạng hôn nhân (1 = Đã kết hôn, 0 = Chưa kết hôn).
- **CHILDREN**: Tình trạng con cái (1 = Có, 0 = Không).
- **POSTAL_CODE**: Mã bưu chính nơi cư trú của cá nhân – có thể giúp suy ra khu vực.
- **ANNUAL_MILEAGE**: Số dặm/kilômét lái xe mỗi năm.
- **SPEEDING_VIOLATIONS**: Số lần vi phạm tốc độ đã được ghi nhận.

- **DUIS:** Số lần bị tai nạn do Lái xe khi bị Ảnh hưởng (DUI - Driving Under Influence) đã xảy ra.
- **PAST_ACCIDENTS**: Số vụ tai nạn giao thông trong quá khứ.
- **TYPE_OF_VEHICLE**: Loại phương tiện (ví dụ: 'Sedan', 'SUV', 'Truck', 'Van').
- **OUTCOME:** Biến mục tiêu cho việc phân loại. Nó có thể đại diện cho việc:

  + cá nhân đó được chấp nhận bảo hiểm,

  + bị coi là rủi ro cao, v.v.

II. Một số đặc điểm quan trọng của dữ liệu (Key data insights)

Để khai phá sâu hơn về dữ liệu, nhóm chia dataset thành 2 loại: **Numerical features và Categorical features**

1. Numerical features:

Có tổng cộng 11 numerical features bao gồm:

'ID',

'CREDIT_SCORE',

'VEHICLE_OWNERSHIP',

'MARRIED', 'CHILDREN',

'POSTAL_CODE',

'ANNUAL_MILEAGE', 'SPEEDING_VIOLATIONS', 'DUIS', 'PAST_ACCIDENTS',

'OUTCOME'.

a. Thống kê cơ bản:

	count	mean	std	min	25%	50%	75%	max
ID	105017.0	394919.044250	279693.639800	101.00000	156329.000000	354655.000000	598594.000000	999976.000000
CREDIT_SCORE	105016.0	0.602161	0.138046	0.06688	0.514855	0.601108	0.703213	0.954075
VEHICLE_OWNERSHIP	105016.0	0.827064	0.378193	0.00000	1.000000	1.000000	1.000000	1.000000
MARRIED	105016.0	0.584159	0.492869	0.00000	0.000000	1.000000	1.000000	1.000000
CHILDREN	105000.0	0.520076	0.499599	0.00000	0.000000	1.000000	1.000000	1.000000
POSTAL_CODE	105016.0	18045.234440	16708.232327	10238.00000	10238.000000	10238.000000	22957.750000	92101.000000
ANNUAL_MILEAGE	105017.0	11060.361656	2975.091433	-14000.00000	9000.000000	11000.000000	13000.000000	21000.000000
SPEEDING_VIOLATIONS	104973.0	0.675888	1.383816	0.00000	0.000000	0.000000	1.000000	20.000000
DUIS	104992.0	0.129772	0.589802	0.00000	0.000000	0.000000	0.000000	6.000000
PAST_ACCIDENTS	105016.0	0.549107	1.402784	-5.00000	0.000000	0.000000	0.000000	15.000000
OUTCOME	105017.0	0.551454	0.497348	0.00000	0.000000	1.000000	1.000000	1.000000

Dựa vào phân tích bảng thống kê mô tả cơ bản, một vấn đề đáng lưu ý về tính toàn vẹn dữ liệu đã được phát hiện ở hai đặc trưng số: **ANNUAL_MILEAGE (Số dặm đi hàng năm) và PAST_ACCIDENTS (Số vụ tai nạn trong quá khứ).** Cụ thể, giá trị nhỏ nhất (min) được ghi nhận của hai cột này lần lượt là -140.000 và -5. Về mặt ngữ nghĩa, đây là các giá trị âm không hợp lệ, vì số dặm bay tích lũy và số vụ tai nạn không thể nhận giá trị nhỏ hơn không.

Nhằm đánh giá toàn diện phạm vi ảnh hưởng của lỗi dữ liệu này trên toàn bộ tập dữ liệu, nhóm nghiên cứu đã tiến hành một bước kiểm tra thống kê bổ sung để xác định tần suất xuất hiện các giá trị âm ở tất cả các cột số. Kết quả của quá trình rà soát này được trình bày trong bảng dưới đây:

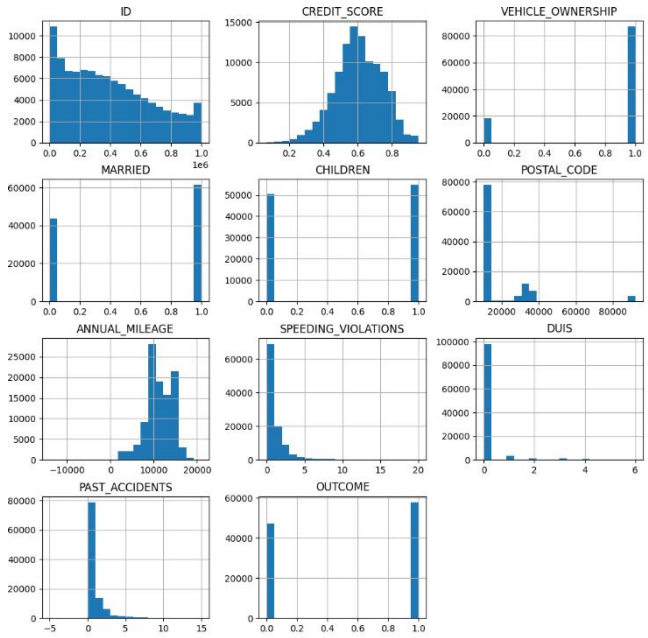
ID	0
CREDIT_SCORE	0
VEHICLE_OWNERSHIP	0
MARRIED	0
CHILDREN	0
POSTAL_CODE	0
ANNUAL_MILEAGE	3
SPEEDING_VIOLATIONS	0
DUIS	0
PAST_ACCIDENTS	3
OUTCOME	0

Các số liệu thống kê đã xác nhận rằng hiện tượng giá trị âm bất hợp lý chỉ tập trung tại hai cột **ANNUAL_MILEAGE và PAST_ACCIDENTS**. Phát hiện này là cơ sở quan trọng cho việc xác định các bước làm sạch và chuẩn hóa dữ liệu cần thiết trong giai đoạn tiền xử lý tiếp theo.

b. Đánh giá phân phối dữ liệu thông qua Skewness coefficient

- |Skewness| < 0.5: Phân phối tương đối đối xứng (gần chuẩn).
- 0.5 ≤ |Skewness| < 1: Phân phối có độ lệch trung bình.
- |Skewness| ≥ 1: Phân phối có độ lệch cao.

ID	0.448446
CREDIT_SCORE	-0.226257
VEHICLE_OWNERSHIP	-1.729648
MARRIED	-0.341512
CHILDREN	-0.080371
POSTAL_CODE	2.986917
ANNUAL_MILEAGE	-0.522199
SPEEDING_VIOLATIONS	3.899239
DUIS	5.607515
PAST_ACCIDENTS	4.344563
OUTCOME	-0.206916



Nhóm các cột có độ lệch dương cao (Positive Skewness)

Các biến liên quan đến lịch sử vi phạm (SPEEDING_VIOLATIONS, DUIS, PAST_ACCIDENTS):

- **Ý nghĩa thực tế:** Độ lệch dương mạnh ở các biến này phản ánh đúng thực tế của dữ liệu rủi ro. Điều này chỉ ra rằng phần lớn người lái xe trong tập dữ liệu không có hoặc có rất ít vi phạm/tai nạn được ghi nhận. Phần đuôi kéo dài về phía bên phải đại diện cho một nhóm nhỏ các cá nhân có rủi ro cao (High-Risk Outliers) với số lượng vi phạm/tai nạn cao bất thường.
- **Tác động đến mô hình:** Đây là dạng dữ liệu sự kiện hiếm (rare events). Sự lệch nghiêm trọng này có thể gây khó khăn cho quá trình mô hình hóa, vì mô hình sẽ có xu hướng học chủ yếu từ các trường hợp có giá trị 0.

POSTAL_CODE:

- **Ý nghĩa:** Mặc dù mã bưu chính là biến định danh (nominal) và giá trị Skewness không mang ý nghĩa thống kê sâu sắc như với biến định lượng, nhưng độ lệch dương mạnh cho thấy dữ liệu được thu thập tập trung cao độ tại một số ít khu vực địa lý nhất định.
- **Tác động đến mô hình:** Điều này cảnh báo về khả năng tồn tại thiên kiến địa lý (geographic bias) trong mẫu dữ liệu, có thể ảnh hưởng đến tính tổng quát hóa của mô hình.

Nhóm các cột có độ lệch âm (Negative Skewness)

ANNUAL_MILEAGE (Số dặm bay hàng năm):

- **Ý nghĩa:** Độ lệch âm mạnh chỉ ra rằng phần lớn người lái xe trong tập dữ liệu có quãng đường di chuyển hàng năm lớn. Phần đuôi bên trái đại diện cho một nhóm nhỏ đi quãng đường rất ít.
- **Lưu ý:** Cần lưu ý rằng độ lệch âm này cũng chịu ảnh hưởng bởi các giá trị bất thường rất nhỏ (bao gồm cả các giá trị âm không hợp lệ đã phát hiện trước đó).
- **Hành động khuyến nghị:** Trước khi mô hình hóa, cần xử lý triệt để các giá trị âm không hợp lệ. Sau đó, cân nhắc áp dụng các kỹ thuật biến đổi dữ liệu để làm cho phân phối đối xứng hơn, giúp cải thiện hiệu suất dự đoán của các mô hình (đặc biệt là mô hình hồi quy) đối với các giá trị ở phần đuôi thấp.

CREDIT_SCORE (Điểm tín dụng):

- **Ý nghĩa:** Phân phối của điểm tín dụng xấp xỉ đối xứng và rất gần với phân phối chuẩn (Normal Distribution). Điều này cho thấy tập dữ liệu bao phủ một phạm vi tín dụng cân bằng, không bị lệch quá nhiều về phía điểm tốt hay điểm kém.
- **Tác động đến mô hình:** Đây là một phân phối lý tưởng cho nhiều thuật toán máy học, đặc biệt là các mô hình tuyến tính. Do đó, đặc trưng này có thể

được sử dụng trực tiếp trong mô hình hóa mà không cần áp dụng các phép biến đổi phức tạp để chuẩn hóa phân phối.

Đánh giá về Hình dạng Phân phối (Distribution Shape):

- **Phân phối xấp xỉ chuẩn (Gaussian-like Distribution):** Chỉ có hai đặc trưng là **CREDIT_SCORE và ANNUAL_MILEAGE** thể hiện hình dạng phân phối tương đối giống phân phối Gaussian (hình chuông). Điều này nhất quán với phân tích độ lệch trước đó, cho thấy chúng có tính đối xứng cao hơn so với các biến khác.
- **Các phân phối khác:** Các đặc trưng định lượng còn lại không thể hiện bất kỳ hình dạng phân phối chuẩn rõ rệt nào. Điều này xác nhận lại kết quả phân tích Skewness: các biến đếm (count data) như **SPEEDING_VIOLATIONS, DUIS, PAST_ACCIDENTS** có độ lệch dương cực mạnh do chứa một lượng lớn giá trị 0, trong khi các biến bản chất là phân loại thì không thể tuân theo phân phối Gaussian.

c. Đánh giá số lượng giá trị duy nhất (Unique values):

	0
ID	98486
CREDIT_SCORE	104976
VEHICLE_OWNERSHIP	2
MARRIED	2
CHILDREN	2
POSTAL_CODE	10937
ANNUAL_MILEAGE	23
SPEEDING_VIOLATIONS	21
DUIS	7
PAST_ACCIDENTS	19
OUTCOME	2

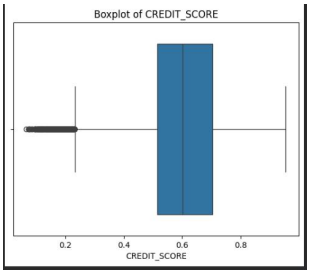
- **Biến có độ biến thiên cao:** Hai đặc trưng ID và CREDIT_SCORE chứa hầu hết các giá trị là duy nhất.
    - Đối với ID, điều này là hiển nhiên vì đây là khóa chính (Primary Key) dùng để định danh từng bản ghi.
    - Đối với CREDIT_SCORE, việc có nhiều giá trị duy nhất cho thấy đây là một biến liên tục (Continuous Variable) với độ phân giải cao. Đặc điểm này rất có lợi cho việc mô hình hóa, giúp phân biệt chi tiết giữa các mức độ rủi ro tín dụng khác nhau.

- **Biến có độ biến thiên thấp:** Các đặc trưng còn lại thể hiện ít biến thiên hơn đáng kể. Điều này chỉ ra rằng chúng có thể là các biến rời rạc (Discrete) với số lượng giá trị hữu hạn.
    - Các biến như **VEHICLE_OWNERSHIP, CHILDREN, và MARRIED** được xác định là các biến nhị phân (Binary Variables).
    - Đặc biệt, biến mục tiêu OUTCOME cũng là một biến nhị phân, xác nhận rằng bài toán đặt ra là **bài toán phân loại nhị phân (Binary Classification Task)**.

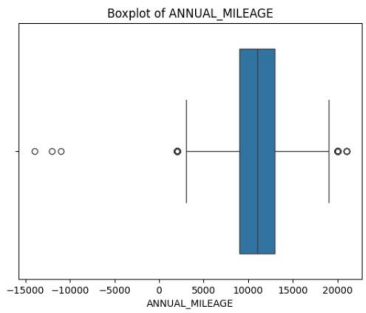
d. Gía trị ngoại lai (Outlier):

Trong bước tiếp theo của quá trình EDA, nhóm sử dụng phương pháp trực quan hóa bằng biểu đồ hộp (Boxplot) để đánh giá sự phân tán của dữ liệu và nhận diện các giá trị ngoại lai (outliers) tiềm ẩn trong các biến định lượng quan trọng. Việc xác định ngoại lai là rất quan trọng để phân biệt giữa nhiễu dữ liệu (cần xử lý) và các trường hợp hiếm mang thông tin giá trị thực tế.

- **Đối với biến CREDIT_SCORE:** Biểu đồ hộp cho thấy phần lớn dữ liệu tập trung quanh giá trị trung vị (khoảng 0.6). Tuy nhiên, đáng chú ý là sự xuất hiện của một dải điểm dày đặc nằm ngoài "râu" dưới (lower whisker), kéo dài từ khoảng giá trị 0.05 đến 0.2. Điều này chỉ ra sự tồn tại của một phân khúc khách hàng đáng kể có điểm tín dụng rất thấp. Trong ngữ cảnh đánh giá rủi ro, đây là nhóm đối tượng quan trọng cần được mô hình ghi nhận, không phải là nhiễu dữ liệu.



- **Đối với biến ANNUAL_MILEAGE:** Biểu đồ hộp cho thấy một dải phân tán rộng hơn đáng kể. Xuất hiện điểm dữ liệu nằm xa phía trên râu trên (upper whisker), chỉ ra sự tồn tại của một nhóm cá nhân có số dặm hàng năm cao bất thường. Kết hợp với phát hiện ở bước thống kê mô tả về việc tồn tại giá trị âm không hợp lệ (min = -140000), có thể kết luận đặc trưng này chứa cả lỗi dữ liệu cần loại bỏ và các giá trị ngoại lai tự nhiên ở ngưỡng cao.



2. Categorical features:

Có tổng cộng 7 categorical features bao gồm:

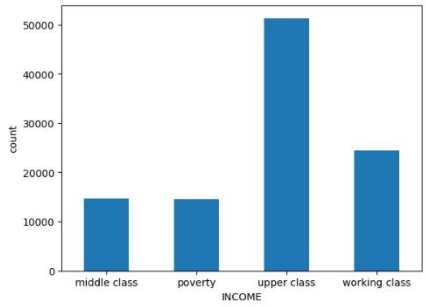
'AGE', 'GENDER',

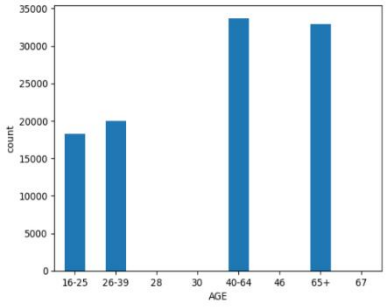
'DRIVING_EXPERIENCE',

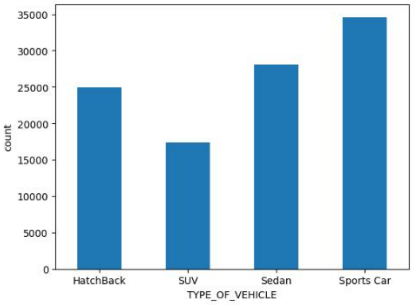
'EDUCATION', 'INCOME',

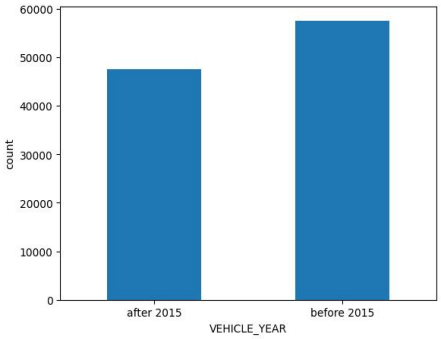
'VEHICLE_YEAR', 'TYPE_OF_VEHICLE'.

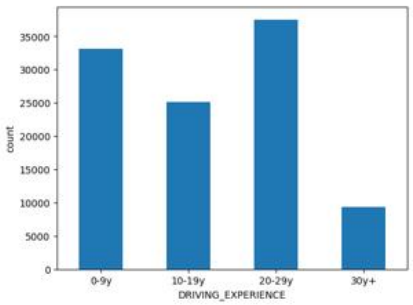
a. Thống kê cơ bản

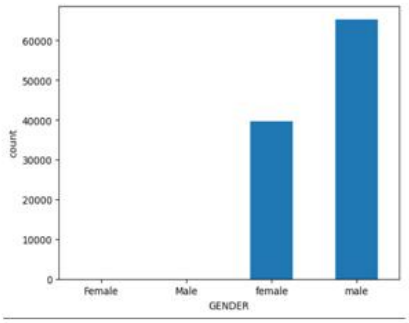








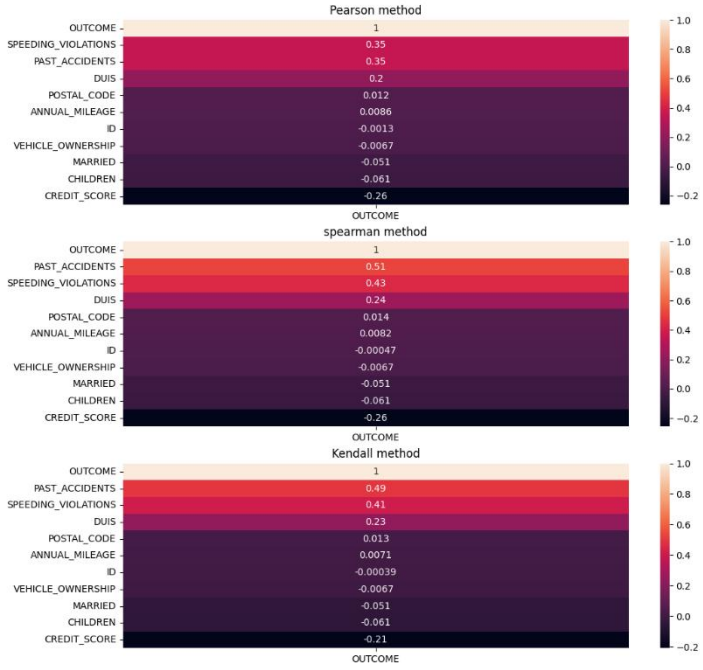




Tên Đặc trưng (Feature)	Các nhóm chính & Số lượng quan sát (Main Categories & Counts)	Vấn đề Dữ liệu cần xử lý (Data Issues to Address)
AGE	40-64 (33,721) 65+ (32,968) 26-39 (20,017) 16-25 (18,307)	Chứa một số ít giá trị số lẻ chưa được phân nhóm (ví dụ: 46, 67, 28, 30).
GENDER	male (65,318) female (39,686)	Lỗi không nhất quán chữ hoa/thường (xuất hiện 'Male', 'Female' bên cạnh 'male', 'female'). Có 1 giá trị thiếu (NaN).
DRIVING_EXPERIENCE	20-29y (37,499) 0-9y (33,117) 10-19y (25,104) 30y+ (9,296)	Có 1 giá trị thiếu (NaN).
EDUCATION	high school (46,598) university (31,214) none (27,170)	Lỗi không nhất quán chữ hoa/thường (xuất hiện 'University' bên cạnh 'university'). Có 26 giá trị thiếu (NaN).
INCOME	upper class (51,281) working class	(Không phát hiện vấn đề bất thường).

	(24,458) middle class (14,739) poverty (14,539)	
VEHICLE_YEAR	before 2015 (57,521) after 2015 (47,495)	Có 1 giá trị thiếu (NaN).
TYPE_OF_VEHICLE	Sports Car (34,596) Sedan (28,126) HatchBack (24,904) SUV (17,391)	(Không phát hiện vấn đề bất thường).

3. Phân tích mối quan hệ giữa Numerical features/Categorical features và OUTCOME



Mối tương quan giữa Numerical features và biến mục tiêu OUTCOME dựa trên 3 phương pháp Pearson, Spearman và Kendall

\

KẾT QUẢ KIỂM ĐỊNH CHI BÌNH PHƯƠNG (CHI-SQUARE TEST)

Biến mục tiêu: OUTCOME
Mức ý nghĩa (alpha): 0.05

	Feature	Chi2 Statistic	P-value	Kết luận
0	AGE	5,016.620	0.000	Có mối liên hệ (Dependent)
1	VEHICLE_YEAR	4.280	0.038	Có mối liên hệ (Dependent)
2	EDUCATION	6.140	0.105	Không có mối liên hệ đáng kể (Independent)
3	GENDER	3.010	0.390	Không có mối liên hệ đáng kể (Independent)
4	INCOME	2.980	0.394	Không có mối liên hệ đáng kể (Independent)
5	DRIVING_EXPERIENCE	1.720	0.632	Không có mối liên hệ đáng kể (Independent)
6	TYPE_OF_VEHICLE	1.650	0.647	Không có mối liên hệ đáng kể (Independent)

Nhận xét:
- Các biến có P-value < 0.05 (thường hiển thị là 0.0 nếu rất nhỏ) là những biến có ảnh hưởng có ý nghĩa thống kê
- P-value càng nhỏ, khả năng hai biến này độc lập với nhau càng thấp (tức là mối quan hệ càng rõ ràng).

Thống kê kiểm định Chi bình phương về mối tương quan giữa Categorical features và biến mục tiêu OUTCOME

Phân tích mối tương quan giữa Biến số (Numerical Features) và Biến mục tiêu (OUTCOME)

Các đặc trưng có mối tương quan DƯƠNG mạnh nhất với Rủi ro:

- Ba đặc trưng liên quan đến lịch sử lái xe là **PAST_ACCIDENTS**, **SPEEDING_VIOLATIONS**, và **DUIS** luôn thể hiện hệ số tương quan dương cao nhất với OUTCOME trên cả ba phương pháp.
- **Ý nghĩa:** Điều này chỉ ra mối quan hệ đồng biến rõ rệt: khi số lượng tai nạn hoặc vi phạm trong quá khứ của một cá nhân tăng lên, khả năng họ được phân loại vào nhóm rủi ro cao (OUTCOME=1) cũng tăng theo.

Đặc trưng có mối tương quan ÂM mạnh nhất với Rủi ro:

- Đặc trưng **CREDIT_SCORE** (Điểm tín dụng) thể hiện hệ số tương quan âm đáng kể nhất và nhất quán trên mọi phương pháp đo lường.
- **Ý nghĩa:** Kết quả này cho thấy mối quan hệ nghịch biến: điểm tín dụng càng cao, xác suất rủi ro của cá nhân càng thấp. Đây là một chỉ báo tài chính quan trọng.

Các đặc trưng có mối tương quan yếu:

- Các biến số còn lại như **ID, ANNUAL_MILEAGE, POSTAL_CODE, VEHICLE_OWNERSHIP, MARRIED,** và **CHILDREN** có hệ số tương quan rất thấp (gần bằng 0) với OUTCOME. Điều này cho thấy chúng có ít mối liên hệ tuyến tính hoặc đơn điệu trực tiếp với việc xác định rủi ro khi xét đơn lẻ.

Kết luận và Định hướng Xử lý

Kết quả phân tích tương quan xác nhận rằng nhóm biến lịch sử vi phạm là những chỉ báo rủi ro mạnh mẽ nhất. Tuy nhiên, khi kết hợp với kết quả phân tích độ lệch (skewness) trước đó, chúng ta nhận thấy đây là các dữ liệu sự kiện hiếm. Việc tương quan mạnh trên dữ liệu sự kiện hiếm cảnh báo nguy cơ cao về **quá khớp (overfitting)** nếu đưa trực tiếp vào mô hình học máy chính.

Do đó, kết quả này **củng cố chiến lược tách biệt dữ liệu** đã đề ra:

- **Nhóm biến lịch sử (tương quan dương mạnh nhưng là sự kiện hiếm):** Sẽ được tách ra để xây dựng hệ thống quy tắc sàng lọc rủi ro độc lập (sau này).
- **Nhóm biến như CREDIT_SCORE (tương quan âm mạnh, phân phối tốt):** Là ứng viên hàng đầu để làm đặc trưng chính cho mô hình dự đoán.
- **Nhóm biến tương quan yếu:** Cần cân nhắc kỹ lưỡng trong giai đoạn lựa chọn đặc trưng (Feature Selection) tiếp theo.

Phân tích Kết quả Kiểm định Chi-bình phương giữa Categorical features và biến mục tiêu OUTCOME.

- **Các biến có mối liên hệ đáng kể với OUTCOME (P-value < 0.05):**
    - **AGE (Tuổi):** Với Chi2 Statistic rất cao (5,016.62) và P-value = 0.000, đây là biến phân loại có ảnh hưởng mạnh mẽ nhất đến biến mục tiêu. Điều này hoàn toàn hợp lý vì độ tuổi thường liên quan mật thiết đến kinh nghiệm và hành vi lái xe.
    - **VEHICLE_YEAR (Năm sản xuất xe):** P-value = 0.038 (nhỏ hơn 0.05), cho thấy có mối liên hệ có ý nghĩa thống kê giữa đời xe và rủi ro.
- **Các biến KHÔNG có mối liên hệ đáng kể với OUTCOME (P-value > 0.05):**
    - **EDUCATION (P=0.105), GENDER (P=0.390), INCOME (P=0.394), DRIVING_EXPERIENCE (P=0.632), TYPE_OF_VEHICLE (P=0.647):** Kết quả kiểm định cho thấy không đủ bằng chứng thống kê để bác bỏ giả thuyết rằng các biến này độc lập với OUTCOME. Điều này có nghĩa là việc biết thông tin về giới tính, thu nhập, hay loại xe, v.v., không giúp ích đáng kể trong việc dự đoán rủi ro một cách trực tiếp khi xét đơn lẻ.

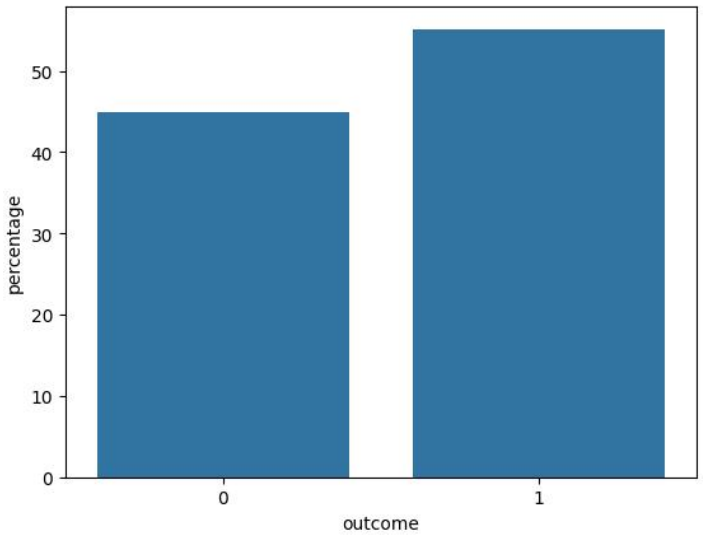
Định hướng Xử lý Tiếp theo (Quan trọng)

Dựa trên kết quả này, nhóm có hai hướng xử lý chính cho bước Feature Selection (Lựa chọn đặc trưng) trong pipeline:

- **Nhận định:** Mặc dù kiểm định Chi-square đơn biến cho kết quả không đáng kể, nhưng các yếu tố như DRIVING_EXPERIENCE (kinh nghiệm lái xe) hay INCOME (liên quan đến khả năng bảo dưỡng xe) về mặt logic nghiệp vụ vẫn được xem là quan trọng đối với rủi ro bảo hiểm.
- **Hành động:**
    - Chỉ loại bỏ ID, POSTAL_CODE để giảm độ phức tạp của mô hình.
    - Để cho các thuật toán máy học mạnh mẽ (như Random Forest, XGBoost) trong giai đoạn modeling.py tự quyết định tầm quan trọng của chúng. Các thuật toán này có khả năng nắm bắt các mối quan hệ phi tuyến và tương tác phức tạp giữa các biến mà kiểm định Chi-square đơn biến không phát hiện được.

Tóm lại: Nhóm ghi nhận rằng AGE và VEHICLE_YEAR có mối liên hệ thống kê mạnh nhất. Tuy nhiên, để tận dụng tối đa khả năng của các mô hình học máy hiện đại trong việc phát hiện các tương tác phức tạp, nhóm quyết định **giữ lại tất cả các đặc trưng phân loại** để đưa vào giai đoạn huấn luyện mô hình tiếp theo.

4. Đánh giá độ cân bằng dữ liệu (biến mục tiêu OUTCOME)



Thống kê số lượng biến OUTCOME

Dữ liệu cho thấy sự phân bố lý tưởng của biến mục tiêu với **45% nhãn An toàn (OUTCOME = 0) và 55% (OUTCOME = 1)** nhãn Rủi ro. Về mặt kỹ thuật, đặc điểm này cho phép nhóm áp dụng trực tiếp các thước đo tiêu chuẩn như Accuracy mà không cần can thiệp bằng các kỹ thuật xử lý mất cân bằng phức tạp. Xét trên khía cạnh bài toán, sự phân phối này—tuy cao hơn tỷ lệ tai nạn thực tế—là một bước chuẩn bị dữ liệu thiết yếu. Nó đảm bảo mô hình được huấn luyện trong môi trường tối ưu nhất để học sâu các tín hiệu rủi ro, thay vì bị lấn át bởi số lượng lớn các trường hợp an toàn.

5. Tổng kết, đưa ra hướng xử lý tiếp theo

a. Đảm bảo chất lượng dữ liệu

- **Xử lý Lỗi Ngữ nghĩa (Semantic Error Correction):** Đối với các giá trị âm bất hợp lý trong cột ANNUAL_MILEAGE và PAST_ACCIDENTS , sử dụng phương pháp Lấy giá trị tuyệt đối (Absolute Value Transformation). Giả định rằng đây là lỗi nhập liệu về dấu (sign error) thay vì dữ liệu rác.
- **Xử lý Lỗi Chính tả (Fuzzy Matching):** Để khắc phục sự không nhất quán trong các biến phân loại (ví dụ: 'University' vs 'university'), áp dụng thuật toán Fuzzy String Matching (sử dụng thư viện fuzzywuzzy) với ngưỡng tương đồng (fuzzy_threshold) là 90. Các giá trị không đạt ngưỡng này sẽ được chuyển về NaN để xử lý như dữ liệu thiếu.
- **Chiến lược Điền khuyết (Imputation Strategy):** Áp dụng chiến lược điền khuyết phân tầng: sử dụng Median cho biến số (để tránh ảnh hưởng bởi ngoại lai) và Mode cho biến phân loại.

Ngoài ra đảm bảo dữ liệu không bị trùng lặp, hợp lệ và đáp ứng đủ 7 tiêu chí chất lượng dữ liệu.

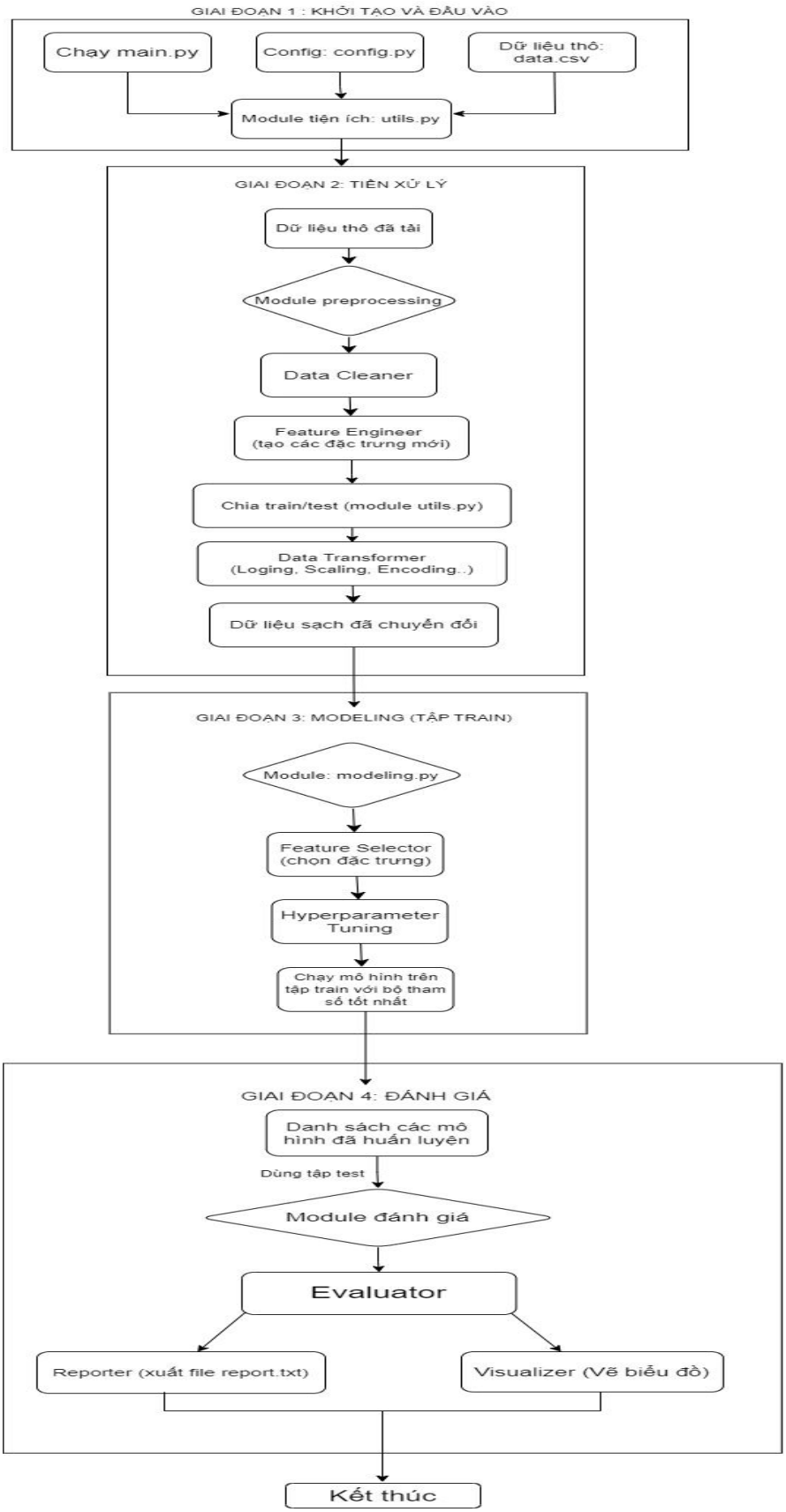
b. Gợi ý biến đổi và chuẩn hóa dữ liệu

- **Xử lý Ngoại lai (Outlier Handling):** Với các biến có phân phối lệch và nhiều ngoại lai như ANNUAL_MILEAGE , áp dụng kỹ thuật Capping (Winsorization) dựa trên phân vị (ví dụ: cắt tại ngưỡng 1% và 99%) thay vì xóa bỏ, nhằm giữ lại thông tin của các nhóm khách hàng đặc biệt.
- **Chuẩn hóa Dữ liệu (Scaling):** Do dữ liệu chứa ngoại lai, ưu tiên sử dụng RobustScaler (dựa trên Trung vị và IQR) cho các biến có độ lệch trung bình , và PowerTransformer (Yeo-Johnson) cho các biến có độ lệch cao để đưa phân phối về dạng gần chuẩn (Gaussian-like), hỗ trợ tốt hơn cho các thuật toán như Logistic Regression và SVM.
- **Mã hóa Biến phân loại (Encoding):**
    - Sử dụng Ordinal Encoding cho các biến có thứ tự như EDUCATION và INCOME.
    - Sử dụng One-Hot Encoding cho các biến định danh như GENDER và TYPE_OF_VEHICLE, giúp mô hình học được đặc trưng mà không tạo ra thứ tự giả

c. Lựu chọn / tạo đặc trưng mới

- **Tạo đặc trưng tương tác (Interaction Features):** Xây dựng các biến tổng hợp như Family Stability bằng cách kết hợp MARRIED, CHILDREN và VEHICLE_OWNERSHIP. Kỹ thuật này giúp mô hình bắt được các mẫu hành vi phức tạp hơn là xét từng biến đơn lẻ.
- **Loại bỏ biến Rò rỉ (Leakage Prevention):** Kiên quyết loại bỏ nhóm biến hành vi (DUIS, SPEEDING) khỏi tập đặc trưng huấn luyện, chuyển chúng sang hệ thống lọc Rule-based độc lập.
- **Lựa chọn đặc trưng (Feature Selection):** Sử dụng kỹ thuật RFE (Recursive Feature Elimination) hoặc Sequential Forward Selection để tự động chọn lọc ra tập hợp các biến tối ưu nhất, giúp giảm chiều dữ liệu và ngăn ngừa Overfitting.

CHƯƠNG 2: THIẾT KẾ VÀ CÀI ĐẶT



DATA PROCESSOR

I. Tiện ích kỹ thuật (Utility Classes)

Nhóm này bao gồm các class chịu trách nhiệm về các tác vụ nền tảng (nạp dữ liệu, ghi log, chia tập) giúp đảm bảo tính ổn định và nhất quán của toàn bộ Pipeline xử lý dữ liệu.

1. Class SystemLogger

a. Tổng Quan Chức Năng:

- **Mục tiêu:** Cấu hình hệ thống ghi log toàn cục (Global Logging Configuration).
- **Tính năng chính:** Ghi log đồng thời ra file và màn hình console với định dạng chuẩn.

b. Tham số khởi tạo (__init__)

Tham số	Kiểu dữ liệu	Mặc định	Mô tả
log_file	Str	"training_process.log"	Tên file log để lưu trữ các thông báo.

c. Hàm chính (setup_logging)

- **Chức năng:** Thiết lập cấu hình log, xóa cấu hình cũ, đặt Level là `INFO`.
- **Log Format:** `%(asctime)s - %(name)s - %(levelname)s - %(message)s`.
- **Handlers:** Cấu hình `logging.FileHandler` (ghi vào file) và `logging.StreamHandler` (in ra màn hình).

2. Class Dataloader

a. Tổng quan chức năng

- **Mục tiêu:** Quản lý việc nạp dữ liệu đa định dạng (CSV, Excel, JSON) vào Pandas DataFrame.
- **Tính năng chính:** Hỗ trợ **Lazy Loading** (tải dữ liệu khi cần), **Factory Pattern** (tạo bộ đọc chuyên biệt qua @classmethod), và truyền tham số linh hoạt (**kwargs).

b. Tham số khởi tạo (__init__)

Tham số	Kiểu dữ liệu	Mô tả
file_path	Str	Đường dẫn tuyệt đối hoặc tương đối tới file dữ liệu.
** **kwargs	Dict	Các tham số tùy chọn khác (như encoding , header) sẽ được truyền trực tiếp vào hàm đọc của Pandas.

c. Phương thức Class (@classmethod)

Các phương thức này giúp khởi tạo đối tượng DataLoader với cấu hình chuyên biệt cho từng loại file.

Hàm	Tham số đặc trưng	Mô tả chức năng
from_csv	sep=', '	Khởi tạo bộ đọc cho file **.csv**. Tự động cảnh báo nếu đuôi file sai.
from_excel	sheet_name=0	Khởi tạo bộ đọc cho file **.xlsx/.xls**. Cho phép chọn sheet cần đọc.
from_json	orient='records'	Khởi tạo bộ đọc file **.json**. Định cấu hình hướng dẫn dữ liệu

d. Thuộc tính và hàm chính (core properties)

Thuộc tính	Kiểu	Mô tả Chức năng	Cơ chế Kỹ thuật
data	@property	Trả về DataFrame. Thực hiện **Lazy Loading**: chỉ đọc file từ ổ cứng khi thuộc tính này được gọi lần đầu tiên.	Kiểm tra self._dataframe is None. Nếu đúng thì gọi _load_data(), ngược lại trả về cache
shape	@property	Trả về kích thước (dòng, cột) của dữ liệu một cách an toàn.	Trả về (0, 0) nếu chưa load hoặc lỗi, tránh crash chương trình
_load_data	Internal	Hàm nội bộ thực hiện việc đọc file vật lý và xử lý lỗi (FileNotFound)	Tự động phát hiện đuôi file qua _get_extension để chọn hàm pd.read_... phù hợp
_get_extension	@staticmethod	Hàm tiện ích tĩnh: trích xuất phần mở rộng (đuôi file) để xác định định dạng	Sử dụng thư viện os.path.splitext(). Độc lập với instance

3. Class DataSplitter

a. Tổng quan chức năng

- **Mục tiêu:** Quản lý việc chia DataFrame thành các tập huấn luyện/kiểm thử hoặc các fold cho Cross-Validation.
- **Tính năng chính:** Hỗ trợ chia đơn giản (train_test_split) với **Stratification** (cân bằng) và **K- Fold**.

b. Tham số khởi tạo (__init__)

Tham số	Kiểu dữ liệu	Mô tả
dataframe	pd.DataFrame DataFrame	dataframe

Tham số	Kiểu dữ liệu	Mô tả
	đã được làm sạch để chia.	
target_col	Str	Tên cột mục tiêu được sử dụng để cân bằng tỷ lệ phân phối khi chia tập (stratify).

c. Hàm chính (Core Methods)

simple_split(self, test_size=0.2)

- **Chức năng**: Chia tập Huấn luyện/Kiểm thử.
- **Cơ chế:** Sử dụng train_test_split . Tự động áp dụng Stratification nếu target_col hợp lệ (để cân bằng tỷ lệ phân phối của biến mục tiêu).
- **Giá trị trả về:** Tuple (train_df, test_df) .

kfold_split_data(self, n_splits)

- **Chức năng:** Tạo ra các chỉ số (indices) cho K-Fold Cross-Validation.
- **Cơ chế:** Sử dụng KFold(shuffle=True, random_state=42) .
- **Giá trị trả về:** Danh sách các Tuple [(train_index, test_index), ...]

II. Tiền xử lí dữ liệu

Nhóm này bao gồm các Class chịu trách nhiệm thực thi trình tự tiền xử lý dữ liệu nghiêm ngặt: Làm sạch, Tạo đặc trưng, Chuẩn hóa và Điều phối luồng công việc.

1. Class DataCleaner

a. Tổng quan chức năng:

- **Kế thừa**: BaseEstimator , TransformerMixin (Tuân thủ giao thức Scikit-learn)
- **Mục tiêu:** Chuẩn hóa, làm sạch, và tiền xử lý dữ liệu thô (Raw Data) bao gồm: xử lý giá trị ngoại lai (Outliers), lỗi chính tả (Typos), giá trị thiếu (Missing Values), và logic nghiệp vụ.
- **Phạm vi:** Dữ liệu dạng bảng (Tabular Data) - Pandas DataFrame.

b. Tham số khởi tạo (__init__): Đây là các tham số cấu hình chính, thiết lập các luật lệ làm sạch dữ liệu

Tham số	Kiểu dữ liệu	Mặc định	Mô tả
golden specs	Dict	{}	Các quy tắc ánh xạ chuẩn cho cột phân loại. Key là tên cột, Value là danh sách các giá trị chuẩn hợp lệ.

Tham số	Kiểu dữ liệu	Mặc định	Mô tả
range_rules	Dict	{}	Các luật giới hạn Min/Max cho các cột số. Key là tên cột, Value là Tuple (min, max).
cols_to_drop	List	[]	Danh sách tên cột cần loại bỏ vĩnh viễn khỏi DataFrame.
fuzzy_threshold	Int	90	Ngưỡng độ chính xác (0-100) của thuật toán Fuzzy Matching. Các giá trị phân loại không đạt ngưỡng này sẽ bị coi là lỗi và chuyển về NaN.
max_drop_ratio	Float	0.05	Ngưỡng tỷ lệ hàng bị lỗi tối đa (≤ 5%). Nếu tổng số hàng vi phạm luật range_rules vượt ngưỡng này, chúng sẽ được GIỮ LẠI và gán NaN (để Impute). Ngược lại, chúng sẽ bị XÓA BỎ (Drop Row).
imputation_strategy	String	auto	Chiến lược điền dữ liệu thiếu 'auto': chỉ dùng Median (số)/ Mode, 'ffill': ưu tiên Forward-fill trước sau đó vét lại bằng Auto, 'bfill': ưu tiên Backward-fill trước sau đó vét lại bằng Auto

c. Hàm chính (Core methods)

fit(self, X, y=None)

- **Chức năng:** Học các tham số cần thiết từ tập dữ liệu huấn luyện (Training Data) X.
- **Mô tả:**
  - Reset bộ nhớ (self.medians_, self.modes_).
  - Phân loại cột: xác định self.numeric_cols và self.categorical_cols.
  - Học tham số điền thiếu (Imputation Parameters): Tính và lưu trữ Median cho cột số (self.medians_) và Mode cho cột phân loại (self.modes_).
- **Giá trị trả về:** self (đã được fit).

transform(self, X)

- **Chức năng:** Áp dụng các luật làm sạch đã cấu hình và các tham số đã học từ fit lên dữ liệu X.
- **Trình tự thực hiện:**
  - _drop_cols(): Loại bỏ cột.
  - _remove_duplicates(): Xóa trùng lặp.
  - _abs_numerical_col(): Lấy trị tuyệt đối.
  - _standardize_categories(): Chuẩn hóa, áp dụng Fuzzy Matching.
  - _fix_logic_age(): Xử lý logic đặc thù (cột AGE).

- _enforce_numerical_ranges(): Xử lý Outliers dựa trên Range Rules và max_drop_ratio.
- _impute_missing_values(): Điền giá trị thiếu bằng Median/Mode đã học hoặc dùng bfill(), ffill().

- **Giá trị trả về:** DataFrame đã được làm sạch

d. Hàm hỗ trợ

Hàm	**Kiểu**	**Mô tả Chức năng**	**Cơ chế Kỹ thuật**
_drop_cols	Internal	Loại bỏ các cột được chỉ định trong self.cols_to_drop.	df.drop(columns=...).
_remove_duplicates	Internal	Xóa các hàng trùng lặp hoàn toàn.	df.drop_duplicates().
_abs_numerical_col	Internal	Chuyển đổi giá trị âm thành dương cho cột số.	df[col].abs().
_standardize_categories	Internal	Chuẩn hóa format (lowercase, strip) và sửa lỗi chính tả cho cột phân loại.	str.lower(), str.strip(). Gọi _fuzzy_helper.
_fix_logic_age	Internal	Xử lý logic riêng cho cột AGE.	Gọi _age_logic_helper.
_enforce_numerical_ranges	Internal	Kiểm tra và xử lý Outliers vi phạm Range Rules.	Ép kiểu số (pd.to_numeric), tạo Global Mask các dòng lỗi, ra quyết định dựa trên tỷ lệ lỗi so với self.max_drop_ratio.
_impute_missing_values	Internal	Điền giá trị thiếu (NaN).	df.fillna() với giá trị từ self.medians_ hoặc

Hàm	Kiểu	Mô tả Chức năng	Cơ chế Kỹ thuật
			self.modes_ hoặc bfill(), ffill() và vét lại bằng self.medians_, self.modes_
_fuzzy_helper	Internal	Logic Sửa Lỗi Chính Tả: Dùng Fuzzy Matching để map giá trị lỗi về giá trị chuẩn nếu độ giống ≥ fuzzy_threshold.	Thư viện fuzzywuzzy.process.extract.
_age_logic_helper	Static	Logic Mapping AGE: Trích xuất số từ chuỗi và ánh xạ vào các khoảng nhóm tuổi (ví dụ: '16-25').	Thư viện re (Regex) và logic điều kiện.

2. Class FeaturesEngineer:

a. Tổng quan chức năng:

- **Kế thừa**: BaseEstimator , TransformerMixin (Tuân thủ giao thức Scikit-learn)
- **Mục tiêu:** Tạo các đặc trưng mới (Derived Features) từ các đặc trưng hiện có (Existing Features) thông qua các phép toán logic hoặc tương tác (Interaction Features).
- **Đặc điểm: Stateless (Không trạng thái) -** Quá trình chuyển đổi là cố định

b. Hàm chính (Core methods)

- **fit(self, X, y=None):** Không học tham số do tính chất stateless. Trả về self.
- **transform(self, X):** Áp dụng các quy tắc kỹ thuật đặc trưng đã định nghĩa. Gọi tuần tự các hàm phụ trợ như self._create_family_features(). Trả về DataFrame X đã bổ sung cột mới.

c. Hàm hỗ trợ:

- **create_family_features:** Tính toán chỉ số tổng hợp "Sự ổn định gia đình" (Family Stability).
- **Công thức:** Tổng giá trị số của các cột MARRIED, CHILDREN, và VEHICLE_OWNERSHIP. Giá trị thiếu được coi là 0.

3. Class DataTransformer

a. Tổng quan chức năng:

- **Kế Thừa:** BaseEstimator, TransformerMixin.

- **Mục Tiêu:** Chuẩn hóa, biến đổi (transform), và mã hóa (encode) dữ liệu. Tập trung xử lý Outliers, Binning, Encoding và Scaling.
- **Đặc điểm:** Stateful - Học các tham số (ngưỡng cắt, Mean/Std/Min/Max) từ tập huấn luyện trong hàm fit.

b. Tham số khởi tạo (__init__)

Tham số	**Mặc định**	**Mô tả**
scaling_strategy	'auto'	Chiến lược chuẩn hóa : − ‘auto’: tự động chọn dựa trên độ lệch − ‘Standard’,‘minmax’, ‘robust’: chỉ rõ thuật toán sử dụng cụ thể − ‘none’ : tắt chuẩn hóa
default_outlier_strategy	'capping'	Chiến lược xử lý Outlier mặc định: 'capping', 'log', hoặc ‘none’.
capping_quantiles	(0.01, 0.99)	Cặp Quantile dùng để tính ngưỡng cắt dưới/trên khi strategy='capping'.
outlier_strategies	{}	Chiến lược xử lý Outlier riêng cho từng cột.
binning_cols	[]	Danh sách cột số sẽ được chia thành nhóm (Binning).
ordinal_mappings	{}	Quy tắc ánh xạ cho cột thứ tự (Ordinal Encoding).
nominal_cols	[]	Danh sách cột định danh sẽ được mã hóa bằng One-Hot Encoding.
ignore_cols	[]	Danh sách cột cần bỏ qua.

c. Hàm chính (Core methods)

- **fit(self, X, y=None)**

· **Chức năng:** Học tất cả các tham số biến đổi cần thiết từ dữ liệu đầu vào.

- **Các bước học:** Học các biên độ chia nhóm, ngưỡng cắt Capping, One-Hot Encoder, và cuối cùng là các Scaler (trên dữ liệu đã được làm sạch tạm thời).

- **transform(self, X)**
- **Chức năng: Áp dụng tuần tự các tham số đã học lên dữ liệu mới X.**
- **Trình tự biến đổi: Chia nhóm (Bining) → Xử lý outlier (Capping/Log) →** Mã hóa (Ordinal/One-hot) → Chuẩn hóa (Scaling).

d. Hàm hỗ trợ và logic kỹ thuật

Giai đoạn	Hàm	Mô tả Chức năng	Cơ chế Kỹ thuật
Binning	fit_binning	Học 5 ngưỡng phân vị (Quantile) để chia nhóm.	KBinsDiscretizer(strategy='quantile').
Outlier	fit_outliers	Học ngưỡng cắt Q_{min} và Q_{max} cho Capping.	Tính df[col].quantile(q).
Encoding	fit_encoding	Học OneHotEncoder cho các cột định danh.	OneHotEncoder(handle_unknown='ignore').
Scaling	fit_scaling	Tự động chọn Scaler dựa trên độ lệch (Skewness).	Xem chi tiết bên dưới.

Chi tiết Logic _fit_scaling (Tự động chọn Scaler): chế độ 'auto'

- Skewness |s| > 1.0 (Cực lệch): Sử dụng PowerTransformer (Yeo-Johnson) để chuyển phân phối về dạng chuẩn.
- Skewness 0.5 < |s| < 1.0 (Lệch trung bình): Sử dụng RobustScaler (dựa trên Median và IQR) để giảm nhạy cảm với ngoại lai.
- Skewness |s| < 0.5 (Ít lệch): Sử dụng MinMaxScaler để chuẩn hóa về khoảng [0, 1].
- Cột Thứ Tự/Chia Nhóm: Luôn sử dụng MinMaxScaler.

Giai đoạn	**Hàm**	**Mô tả Chức năng**	**Cơ chế Kỹ thuật**
Binning	_transform_binning	Áp dụng quy tắc chia nhóm đã học để chuyển đổi số liên tục thành chỉ số nhóm (0, 1, 2, 3, 4).	Gọi binner.transform() từ KBinsDiscretizer đã fit.
Outlier	_transform_outliers	Xử lý giá trị ngoại lai: Log hóa (nếu lệch dương) hoặc Cắt ngọn	Sử dụng np.log1p() hoặc df.clip(lower, upper).

		(Capping) dựa trên ngưỡng đã học.	
Encoding	_transform_encoding	Mã hóa biến phân loại: Ánh xạ từ điển cho Ordinal và biến đổi vector nhị phân cho Nominal.	df.map() (Ordinal) và encoder.transform() (One-Hot) kết hợp pd.concat.
Scaling	_transform_scaling	Chuẩn hóa tất cả các cột số (gồm cả cột gốc, cột sau binning/encoding) về cùng miền giá trị.	Duyệt từng cột và gọi scaler.transform().

2. Class DataPreparationPipeline:

a. Tổng quan chức năng:

- **Mục Tiêu:** Quản lý và thực thi toàn bộ luồng tiền xử lý dữ liệu từ khâu nạp dữ liệu thô, làm sạch, tạo đặc trưng, đến tách và chuẩn hóa tập Huấn luyện/Kiểm thử.
- **Nguyên tắc cốt lõi:** Tránh Data Leakage (Rò rỉ dữ liệu) - Đảm bảo các tham số học chỉ được tính toán trên tập huấn luyện.

b. Tham số khởi tạo (__init_)

Tham số	**Kiểu dữ liệu**	**Mô tả**
file_path	Str	Đường dẫn tới file dữ liệu thô.
cleaner	DataCleaner	Instance đã cấu hình cho bước Làm sạch cơ bản.
featuring	FeatureEngineer	Instance đã cấu hình cho bước Tạo đặc trưng.
transformer	DataTransformer	Instance đã cấu hình cho bước Chuẩn hóa/Mã hóa.
target_col	Str	Tên cột mục tiêu dùng để chia tập hoặc cân bằng dữ liệu.

c. Quy trình thực thi

Hàm run (self, test_size = 0.2) -> Trả về tupple(train_data, test_data)

STT	**Giai đoạn**	**Hành động Kỹ thuật**	**Lý do & Nguyên tắc**

STT	Giai đoạn	Hành động Kỹ thuật	Lý do & Nguyên tắc
1	Nạp Dữ liệu	Nạp dữ liệu thô từ file_path.	Khởi tạo dữ liệu
2	Làm Sạch Dữ liệu và Tạo Đặc Trưng Mới	Áp dụng cleaner.fit_transform() và featuring.fit_transform() trên toàn bộ dữ liệu.	Các luật làm sạch cơ bản (xóa trùng lặp, điền Median/Mode thô) và luật tạo đặc trưng là global nên được áp dụng trước khi tách tập.
3	Tách Tập (Split)	Chia dữ liệu đã sơ chế (df_clean) thành train_df và test_df (sử dụng Stratification nếu target_col tồn tại).	Nguyên tắc Vàng: Đây là điểm phân tách để tránh rò rỉ dữ liệu (Data Leakage).
4	Chuẩn hóa, Mã hóa	Học: transformer.fit(train_df) (chỉ học từ tập Train). Áp dụng: transformer.transform(train_df) và transformer.transform(test_df).	Tính nhất quán: Các tham số thống kê được học từ tập huấn luyện được áp dụng đồng nhất lên tập kiểm thử.

MÔ HÌNH HỌC MÁY (MACHINE LEARNING MODELING)

Phần này chịu trách nhiệm xây dựng, huấn luyện, tinh chỉnh và đánh giá các thuật toán học máy. Các mô hình được hỗ trợ bao gồm: **Logistic Regression, SVM, Decision Tree, Random Forest, XGBoost, KNN.**

I. Module modeling.py

1. Class ModelTrainer

a. Tổng Quan Chức Năng

- **Mục tiêu:** Quản lý việc khởi tạo, huấn luyện, dự đoán và lưu trữ mô hình.
- **Tính năng chính:** Hỗ trợ đa dạng thuật toán thông qua giao diện thống nhất và tự động lưu file .pkl.

b. Tham số khởi tạo (__init__)

Tham số	Kiểu dữ liệu	Mô tả
model_name	str	Tên định danh của mô hình (ví dụ: 'xgboost', 'svm').
**kwargs	dict	Các tham số tùy chọn khác truyền vào mô hình (hyperparameters).

c. Hàm chính

- **build_model:** Khởi tạo đối tượng mô hình từ thư viện Scikit-learn/XGBoost dựa trên tên. Sử dụng cấu trúc if-elif để trả về instance tương ứng.
- **train_model(X_train, y_train):** Huấn luyện mô hình bằng cách gọi phương thức .fit()
- **predict_y(X_test):** Thực hiện dự đoán trên tập kiểm thử bằng .predict().
- **save_model:** Lưu trạng thái mô hình đã huấn luyện ra file vật lý .pkl sử dụng joblib.
- **get_supported_models (Static):** Trả về danh sách các mô hình được hệ thống hỗ trợ.

2. Class HyperparameterTuning

a. Tổng quan chức năng

- **Mục tiêu:** Tự động tìm kiếm bộ siêu tham số (Hyperparameters) tối ưu nhất cho từng loại mô hình.

- **Phương pháp:** Hỗ trợ cả **Grid Search** và **Random Search**.

b. Tham số khởi tạo (__init__)

Tham số	Kiểu dữ liệu	Mô tả
tuning_method	str	Phương pháp tìm kiếm: 'grid_search', 'random_search' hoặc 'default'.
scoring	str	Chỉ số đánh giá để tối ưu hóa (ví dụ: 'f1', 'accuracy').

c. Hàm chính (Core methods)

- **_get_model_config (Internal):** Định nghĩa không gian tìm kiếm (Search Space) chứa các tham số cần thử nghiệm cho từng loại model.
- **tune_hyperparameters:** Thực thi quá trình tìm kiếm tham số tốt nhất sử dụng Cross-Validation. Trả về bộ tham số tối ưu (Best Params).

3. Class FeatureWrapperSelector

a. Tổng quan chức năng:

- **Mục tiêu:** Giảm chiều dữ liệu bằng cách chọn ra các đặc trưng quan trọng nhất, loại bỏ nhiễu.
- **Kỹ thuật:** Sử dụng phương pháp Wrapper: **RFE** (Recursive Feature Elimination) hoặc **Sequential Selection** (Forward/Backward).

b. Hàm chính:

- **Fit_transform(X, y):** Học từ dữ liệu và trả về DataFrame mới chỉ chứa các cột đặc trưng quan trọng nhất đã được chọn lọc.

II. Module evaluation.py (Đánh giá & Trực quan hóa)

1. Class ModelEvaluator:

- **Mục tiêu:** Tính toán các chỉ số hiệu suất từ kết quả dự đoán.
- **Thuộc tính @property metrics:** Tự động tính toán và trả về Dictionary chứa các chỉ số trung bình: **Accuracy, Precision, Recall, F1-Score.**

2. Class ReportModel

- **Mục tiêu:** Tổng hợp kết quả so sánh giữa các mô hình và xuất báo cáo dạng văn bản.
- **Hàm chính save_comparision**: Tạo bảng xếp hạng hiệu suất (sắp xếp theo F1-Score) và lưu vào file .txt.

3. Class Visualizer

- **Mục tiêu:** Vẽ biểu đồ để phân tích trực quan kết quả (Confusion Matrix, ROC).
- **Tham số khởi tạo:**
  - cv_results: Danh sách kết quả từ Cross-validation.
  - model_name_prefix: Tên mô hình (dùng làm tiêu đề/tên file).
  - save_dir: Đường dẫn thư mục lưu ảnh (mặc định 'RESULT').
- **Hàm chính:** plot_confusion_matrix lưu ảnh png và plot_ROC_curve (lưu ảnh png).

III. Module pipeline.py

Đây là lớp quản lý trung tâm, kết nối tất cả các module trên thành một quy trình tự động khép kín.

1. Tham số khởi tạo (__init__)

Tham số	Mô tả
train_file, test_file	Đường dẫn file dữ liệu đã qua xử lý (Train/Test).
tuning_method	Phương pháp tinh chỉnh tham số (Grid/Random).
feature_method	Phương pháp chọn đặc trưng (RFE/Forward/Backward).
n_features	Số lượng đặc trưng mong muốn giữ lại.

2. Quy trình thực thi hàm

Giai đoạn	Hành động Kỹ thuật
1. Load & Split	Nạp dữ liệu Train/Test. Tách biệt biến đặc trưng (X) và nhãn (y).
2. Feature Selection	Gọi FeatureWrapperSelector. Fit trên tập Train để tìm ra tập đặc trưng tốt nhất, sau đó áp dụng bộ lọc này lên tập Test (Tránh rò rỉ dữ liệu).
3. Model Loop	Vòng lặp duyệt qua từng mô hình: 3.1. Tuning: Gọi HyperparameterTuning. Sử dụng K-Fold trên tập Train để tìm tham số tối ưu. 3.2. Training: Gọi ModelTrainer. Huấn luyện lại (Retrain) trên toàn bộ tập Train với tham số tốt nhất. 3.3. Prediction: Dự đoán nhãn trên tập Test. 3.4. Eval & Viz: Tính toán chỉ số, vẽ biểu đồ và lưu Model (.pkl).

Giai đoạn	Hành động Kỹ thuật
4. Reporting	Tổng hợp kết quả tất cả mô hình và xuất báo cáo cuối cùng.

IV. Kỹ Thuật K-Fold Cross Validation

Để đảm bảo mô hình có độ ổn định cao và tránh hiện tượng quá khớp (Overfitting), dự án áp dụng kỹ thuật **K-Fold Cross Validation** (với $K = 3$) tại hai giai đoạn quan trọng:

Giai đoạn 1: Lựa chọn Đặc trưng (Feature Selection)

Cơ chế: Khi sử dụng SequentialFeatureSelector (Forward/Backward), thuật toán chia tập Train thành 3 phần. Một tập đặc trưng chỉ được chọn nếu nó mang lại hiệu quả tốt trung bình trên cả 3 lần kiểm thử chéo này.

Giai đoạn 2: Tinh chỉnh Siêu tham số (Hyperparameter Tuning)

Cơ chế: Trong GridSearchCV hoặc RandomizedSearchCV :

- Tập Train được chia làm 3 folds.
- Mỗi bộ tham số ứng viên (ví dụ: max_depth=5) được huấn luyện và đánh giá 3 lần.
- Điểm số cuối cùng là trung bình cộng của 3 lần chạy.

Ý nghĩa: Đảm bảo bộ tham số được chọn là tối ưu thực sự cho tổng thể dữ liệu.

Lưu ý: Sau khi tìm được tham số tốt nhất nhờ K-Fold, mô hình cuối cùng (Final Model) sẽ được **học thật 100%** trên toàn bộ tập Train (không chia K- Fold nữa) để tận dụng tối đa dữ liệu trước khi dự đoán trên tập Test.

GIAO DIỆN DÒNG LỆNH (CLI & MAIN EXECUTION)

1. **Cấu hình Tham số (parse_arguments):** người dùng điều khiên quy trình chạy thông qua các "cờ" (flags) sau:

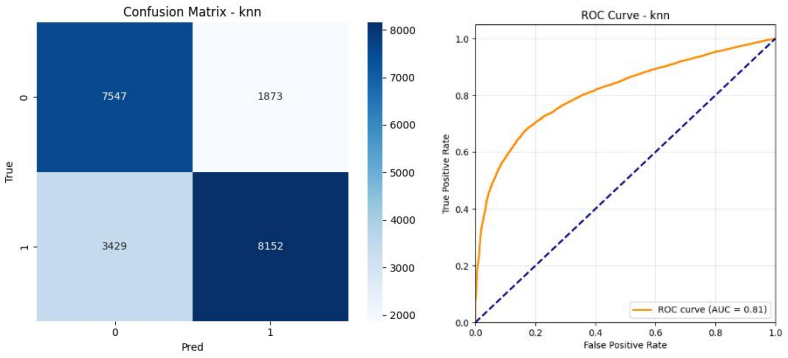
Tham số (Flag)	Mặc định	Mô tả Chức năng
--file	DATA/...	Đường dẫn đến file dữ liệu gốc cần huấn luyện.
--test_size	0.2	Tỷ lệ chia tập kiểm thử (Test set).
--tuning	'random'	Phương pháp tinh chỉnh tham số (grid_search, random_search, default).
--feature_method	'rfe'	Phương pháp chọn đặc trưng (rfe, forward, backward).
--n_features	15	Số lượng đặc trưng quan trọng muốn giữ lại.

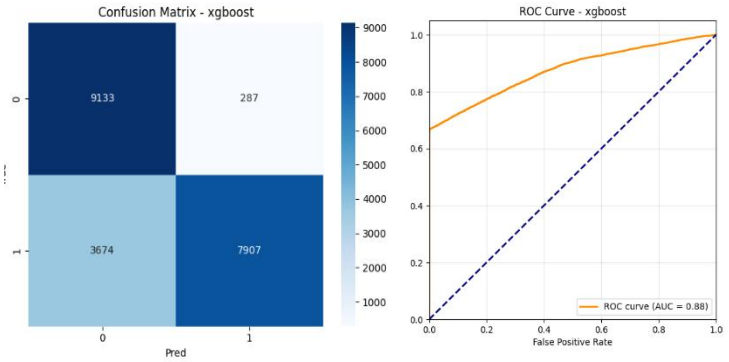
2. **Luồng thực thi chính (main):** Hàm main() đóng vai trò là điểm khởi đầu (Entry Point), thực hiện tuần tự:

- **Parsing:** Đọc các tham số cấu hình từ dòng lệnh.
- **Preprocessing:** Gọi DataPreparationPipeline để làm sạch và chia dữ liệu -> Lưu 2 file sạch (final_train_data.csv, final_test_data.csv) vào thư mục DATA/.
- **Modeling:** Khởi tạo AutoMLPipeline với file dữ liệu vừa tạo để bắt đầu huấn luyện.
- **Reporting:** Kết quả cuối cùng (Report, Model, Image) được lưu tập trung vào thư mục RESULT/.

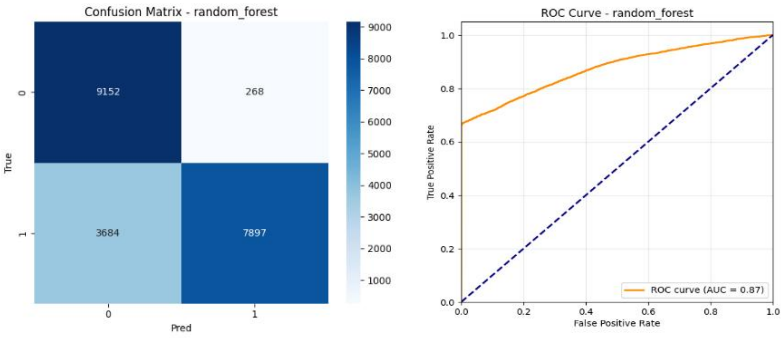
CHƯƠNG 3: THỰC NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ

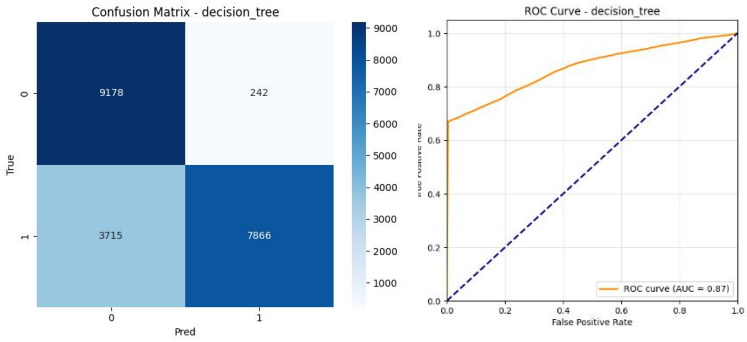
Sau khi cài đặt chương trình, nhóm đã chạy thực hiện chạy thực nghiệm và cho kết quả như sau:



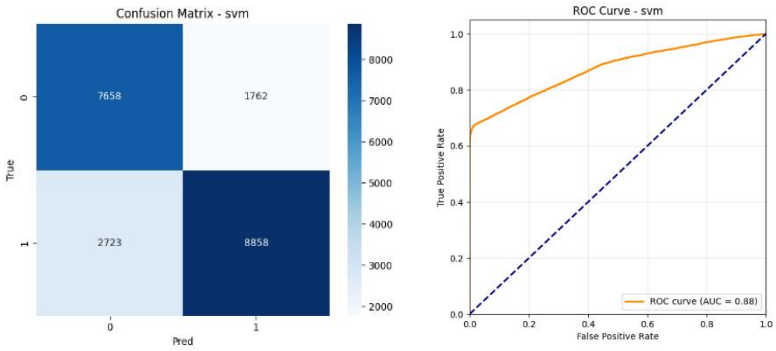


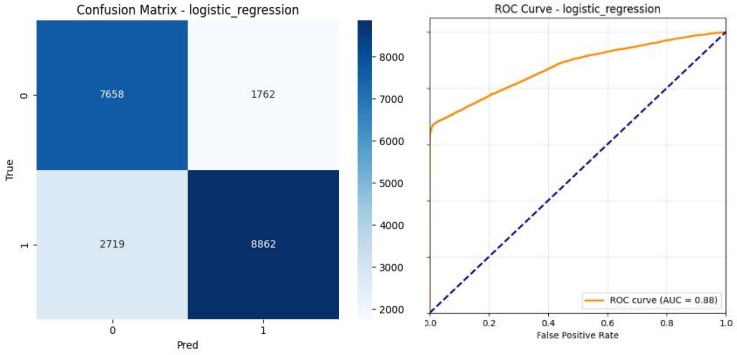
KNN ***XG-boost***





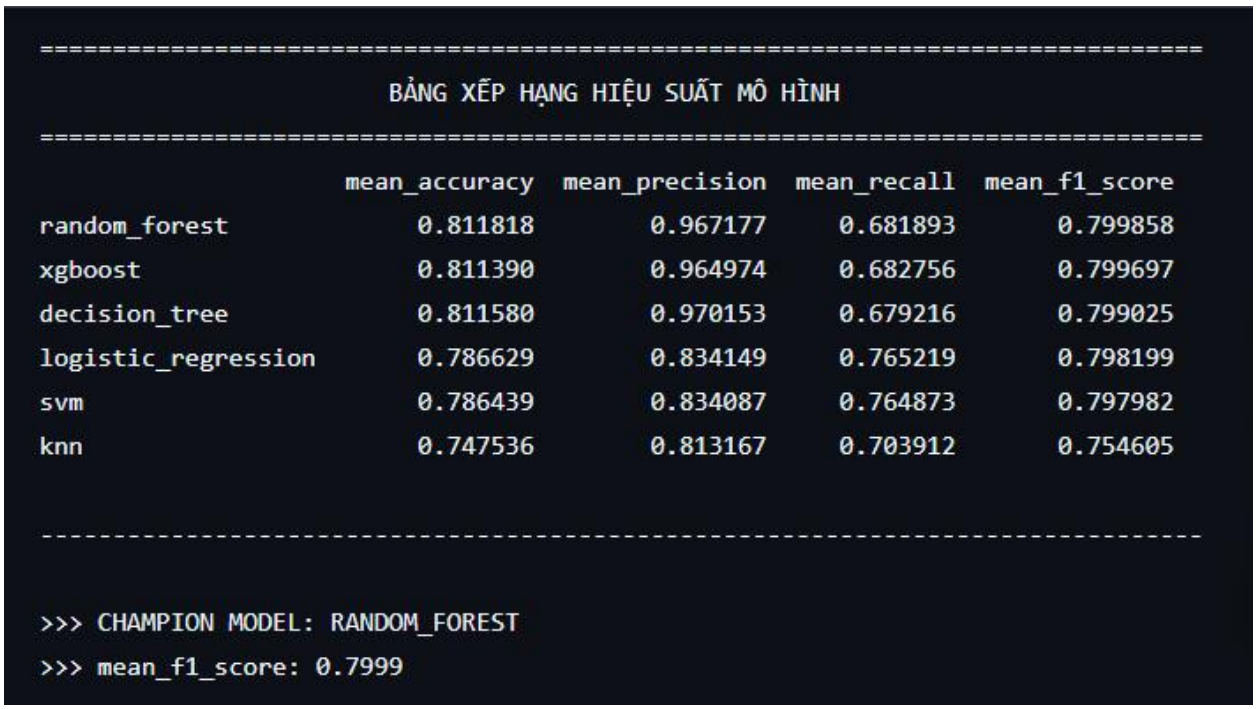
Random_forest ***Decision_tree***





SVM ***Logistic_regression***

Confusion Matrix và ROC Curve của 6 mô hình sau khi chạy thực nghiệm



```
================================================================================
                        BẢNG XẾP HẠNG HIỆU SUẤT MÔ HÌNH
================================================================================
                     mean_accuracy  mean_precision  mean_recall  mean_f1_score
random_forest             0.811818        0.967177     0.681893       0.799858
xgboost                   0.811390        0.964974     0.682756       0.799697
decision_tree             0.811580        0.970153     0.679216       0.799025
logistic_regression       0.786629        0.834149     0.765219       0.798199
svm                       0.786439        0.834087     0.764873       0.797982
knn                       0.747536        0.813167     0.703912       0.754605

--------------------------------------------------------------------------------

>>> CHAMPION MODEL: RANDOM_FOREST
>>> mean_f1_score: 0.7999
```

Bảng đánh giá hiệu suất mô hình

1. Tổng quan:

Mô hình đã đạt được độ ổn định cao với F1-Score trung bình xấp xỉ 0.80 ở hầu hết các thuật toán hàng đầu. Điều này chứng tỏ chiến lược xử lý dữ liệu đã phát huy hiệu quả, giúp mô hình học được các đặc trưng cốt lõi mà không bị hiện tượng "học vẹt" (overfitting) hay rò rỉ dữ liệu.

2. Phân tích chi tiết từng nhóm mô hình

a. Nhóm Mô hình Cây (Tree-Based Models: Random Forest, XGBoost, Decision Tree)

Đây là nhóm hoạt động hiệu quả nhất, chiếm trọn 3 vị trí đầu bảng xếp hạng.

- Điểm mạnh vượt trội - Độ chính xác (Precision) cực cao (~96-97%):
    - Cả 3 mô hình này đều đạt Precision trên 96%. Điều này có nghĩa là: Khi mô hình cảnh báo một khách hàng là "Có rủi ro", thì xác suất dự báo đó chính xác lên tới 97%.
    - Tỷ lệ báo động giả (False Positive) cực thấp. Đây là ưu điểm lớn nếu doanh nghiệp muốn tránh làm phiền khách hàng tốt.
- Điểm hạn chế - Độ nhạy (Recall) trung bình (~68%):
    - Mô hình có xu hướng "thận trọng", chỉ bắt những trường hợp rủi ro rất rõ ràng, do đó bỏ sót khoảng 32% số ca rủi ro tiềm ẩn khó phát hiện hơn.

b. Nhóm Mô hình Tuyến tính & Vector (Logistic Regression, SVM)

- Độ phủ (Recall) tốt hơn (~76%):
    - Tuy xếp hạng thấp hơn về F1-Score, nhưng Logistic Regression và SVM lại có khả năng phát hiện rủi ro rộng hơn (bắt được 76% số ca rủi ro).
    - Đổi lại, độ chính xác (Precision) giảm xuống còn ~83%, tức là chấp nhận tỷ lệ bắt nhầm cao hơn để không bỏ sót.

Tóm lại: Dựa trên bảng xếp hạng hiệu suất, Random Forest là lựa chọn chính thức cho bài toán này. Mô hình này phù hợp với chiến lược quản trị rủi ro tập trung vào chất lượng dự báo (thà bỏ sót còn hơn bắt nhầm), đảm bảo các quyết định từ chối hoặc tăng phí bảo hiểm/tín dụng đều có cơ sở dữ liệu cực kỳ vững chắc.

CHƯƠNG 4: KẾT LUẬN

Đồ án đã thiết kế và triển khai thành công một quy trình xử lý dữ liệu (Data Pipeline) hoàn chỉnh từ khâu tiền xử lý (Preprocessing) đến mô hình hóa (Modeling) sử dụng ngôn ngữ Python cùng hệ sinh thái các thư viện phân tích dữ liệu mạnh mẽ (Pandas, Numpy, Scikit-learn).

Khác với các phương pháp tiếp cận máy móc thông thường, điểm sáng tạo cốt lõi của đồ án nằm ở việc phát hiện và xử lý triệt để vấn đề Rò rỉ dữ liệu (Data Leakage). Thông qua quá trình Phân tích Khám phá (EDA), nhóm nhận định rằng việc đưa trực tiếp các biến hành vi hiếm vào mô hình học máy sẽ dẫn đến hiện tượng "học vẹt" (Overfitting), khiến mô hình đạt điểm số ảo nhưng mất khả năng dự báo thực tế.

Đồ án đã xây dựng được một khung làm việc (Framework) chuẩn bằng Python, bao gồm các bước xử lý dữ liệu nhiễu, chuẩn hóa danh mục, và đặc biệt là kỹ thuật tạo đặc trưng mới (Feature Engineering). Quy trình này có tính tái sử dụng cao và dễ dàng mở rộng cho các bài toán tương tự.

Sau quá trình thực nghiệm so sánh các thuật toán (Logistic Regression, SVM, Decision Tree, XGBoost), mô hình **Random Forest** đã được lựa chọn làm mô hình tối ưu (Champion Model) với các chỉ số hiệu suất vượt trội:

- **F1-Score đạt ~0.80:** Cho thấy sự cân bằng hài hòa giữa độ chính xác (Precision) và độ phủ (Recall), khắc phục được điểm yếu của các bộ dữ liệu mất cân bằng.
- **Precision đạt ~97%:** Đây là chỉ số quan trọng nhất về mặt độ tin cậy. Khi mô hình đưa ra cảnh báo rủi ro, xác suất dự báo đúng lên tới 97%. Điều này giúp giảm thiểu tối đa tỷ lệ báo động giả (False Positive), đảm bảo tính khả thi khi triển khai vào quy trình nghiệp vụ.
- **Khả năng Tổng quát hóa (Generalization):** Kết quả này chứng minh mô hình đã học được các mối quan hệ phi tuyến tính phức tạp giữa các đặc trưng nhân khẩu học (Tuổi, Tín dụng, Thu nhập...) thay vì chỉ ghi nhớ các quy tắc đơn giản từ dữ liệu lịch