TRƯỜNG ĐẠI HỌC CÔNG NGHỆ
ĐẠI HỌC QUỐC GIA HÀ NỘI - VNU
VIỆN TRÍ TUỆ NHÂN TẠO

Dự Án

PHÁT HIỆN GIAN LẬN THẺ TÍN DỤNG SỬ DỤNG CÁC THUẬT TOÁN HỌC MÁY

Đáp ứng một phần yêu cầu đối với môn Học máy (I_AIT2004#_37)
Viện Trí tuệ Nhân tạo
QH-2023-I-CQ-AI2

Được viết bởi:
23020384 Nguyễn Đình Khải
23020363 Vi Minh Hiển

Tháng 12, 2024

MỤC LỤC

Danh sách hình vẽ

Danh sách bảng

1. Giới Thiệu

1.1. Tổng quan về vấn đề

Với sự phát triển nhanh chóng của công nghệ tài chính, các giao dịch thẻ tín dụng ngày càng trở nên phổ biến. Tuy nhiên, điều này cũng đi kèm với những rủi ro liên quan đến gian lận thẻ tín dụng, gây thiệt hại lớn cho các tổ chức tài chính và khách hàng. Phát hiện gian lận thẻ tín dụng đã trở thành một bài toán quan trọng trong lĩnh vực ngân hàng và thanh toán trực tuyến, yêu cầu các hệ thống có khả năng phân tích và phát hiện các giao dịch gian lận một cách chính xác và hiệu quả.

Trong khi những phương pháp truyền thống chủ yếu dựa vào các quy tắc và các mẫu được xác định trước, các phương pháp học máy đã mang lại bước đột phá mới trong việc phát hiện gian lận thẻ tín dụng. Học máy giúp xây dựng các mô hình tự động học từ dữ liệu và phát hiện các giao dịch bất thường mà không cần phải dựa vào các quy tắc cứng nhắc.

1.2. Tầm quan trọng của việc phát hiện gian lận thẻ tín dụng

Gian lận thẻ tín dụng có thể gây ra thiệt hại lớn cho các tổ chức tài chính và người tiêu dùng. Theo báo cáo của các cơ quan giám sát tài chính, gian lận thẻ tín dụng chiếm tỷ lệ đáng kể trong tổng số các hành vi gian lận tài chính. Các tổ chức tài chính cần có các biện pháp để giảm thiểu các rủi ro này, bảo vệ tài sản của khách hàng và nâng cao uy tín thương hiệu. Việc phát hiện gian lận một cách chính xác và kịp thời có thể giúp giảm thiểu thiệt hại và ngừng các hành vi gian lận trước khi chúng lan rộng.

Việc áp dụng học máy vào bài toán này giúp hệ thống có khả năng nhận diện được những giao dịch gian lận mới mà không cần phải dựa vào các mẫu cố định, từ đó tăng tính linh hoạt và hiệu quả của hệ thống.

1.3. Mục tiêu nghiên cứu

Mục tiêu của nghiên cứu này là phát triển một hệ thống học máy để phát hiện gian lận thẻ tín dụng. Cụ thể, nghiên cứu sẽ tập trung vào các thuật toán học máy cơ bản như Logistic Regression, Random Forest, SVM, và XGBoost để xây dựng mô hình phân loại cho bài toán này. Bằng cách sử dụng các thuật toán này, chúng ta sẽ phân tích và so sánh hiệu quả của chúng trong việc phát hiện gian lận, từ đó tìm ra phương pháp tối ưu nhất. [**?**].

1.4. Lý do lựa chọn đề tài

Lý do chọn đề tài này là vì gian lận thẻ tín dụng không chỉ gây thiệt hại về kinh tế mà còn ảnh hưởng đến sự tin cậy của khách hàng đối với các tổ chức tài chính.
Ngoài ra, đề tài này cũng mang lại cơ hội áp dụng kiến thức học máy đã học vào thực tế, đồng thời có phần xử lý dữ liệu phức tạp, giúp nâng cao kỹ năng về xử lý và phân tích dữ liệu. [**?**].

2. Tổng Quan Về Các Phương Pháp Học Máy Sử Dụng

2.1. Khái quát về học máy

Học máy (Machine Learning) là lĩnh vực nghiên cứu cho phép máy tính học hỏi, suy luận và hành động dựa trên dữ liệu. Điều này được thực hiện thông qua việc xây dựng các chương trình máy tính có khả năng xử lý dữ liệu, rút ra thông tin hữu ích, đưa ra dự đoán về các thuộc tính chưa biết, và gợi ý các hành động hoặc quyết định cần thực hiện.
Học máy tự động hóa quy trình phân tích dữ liệu và cho phép máy tính học từ dữ liệu mà không cần lập trình chi tiết cho từng trường hợp cụ thể.

2.2. Baseline cho mô hình

Classification Models:

- `Logistic Regression`
- `Decision Tree`
- `Random Forest`
- `XGBoost`

Class Imbalance Solutions:

- `Imbalance`
- `SMOTE`
- `ADASYN`

Metrics:

- `Accuracy Score`
- `Confusion Matrix`
- `Precision Score`

- `Recall Score`
- `ROC_AUC`
- `F1 Score`

2.3. Các thuật toán học máy được sử dụng

2.3.1. Logistic Regression

Logistic Regression (Hồi quy Logistic) là thuật toán cơ bản trong học máy được sử dụng cho bài toán phân loại nhị phân. Phương pháp này dựa trên mô hình tuyến tính để ước tính xác suất một mẫu thuộc về một lớp cụ thể.
Cơ chế hoạt động: Logistic Regression sử dụng hàm sigmoid để chuyển đổi đầu ra của một phương trình tuyến tính thành xác suất nằm trong khoảng [0, 1]. Mẫu được gán nhãn "1" nếu xác suất lớn hơn ngưỡng (thông thường là 0.5), ngược lại là nhãn "0".
Hàm sigmoid:

$$\sigma(z) = \frac{1}{1+\exp^{-z}}$$

trong đó $z = w^T x + b$ với w là trọng số, x là vector đặc trưng, và b là hằng số bù.
Ứng dụng là để phân loại các giao dịch gian lận dựa trên xác suất.
Ưu điểm của mô hình là đơn giản, dễ triển khai, tính toán nhanh chóng. Khả năng diễn giải kết quả tốt nhờ trọng số của các đặc trưng.
Tuy nhiên mô hình còn hạn chế hiệu suất kém khi dữ liệu không tuyến tính.Nhạy cảm với mất cân bằng dữ liệu, cần các phương pháp bổ trợ để điều chỉnh.

2.3.2. Decision Tree

Decision Tree (Cây quyết định) là thuật toán phân loại dựa trên việc chia nhỏ dữ liệu thành các nhánh tương ứng với các điều kiện khác nhau.
Cơ chế hoạt động: Decision Tree sử dụng các tiêu chí như Information Gain hoặc Gini Impurity để xác định điểm phân chia tối ưu tại mỗi nút. Mỗi nút lá (leaf node) đại diện cho một lớp hoặc xác suất của một lớp.
Các chỉ số phân chia chính:
Gini Impurity: Đo lường mức độ lẫn lộn của các lớp tại một nút.

$$G = 1 - \sum_{i=1}^{C} p_i^2$$

trong đó p_i là xác suất của lớp i.
Information Gain: Đo lường mức độ giảm entropy sau một phân chia.

$$IG = H(parent) - \sum_k \frac{|D_k|}{|D|} H(D_k)$$

trong đó H là entropy và D_k là tập con sau khi phân chia.
Ưu điểm là dễ hiểu, trực quan, đặc biệt với dữ liệu ít nhiễu. Có thể xử lý cả đặc trưng định tính và định lượng.
Hạn chế, dễ bị overfitting nếu cây quá sâu hoặc dữ liệu chứa nhiễu.Nhạy cảm với thay đổi nhỏ trong dữ liệu, làm thay đổi cấu trúc cây.

2.3.3. Random Forest

Random Forest là một thuật toán mạnh mẽ thuộc nhóm ensemble, được tạo thành từ nhiều cây quyết định độc lập.
Cơ chế hoạt động:
Dữ liệu được lấy mẫu ngẫu nhiên (bootstrap sampling) để xây dựng mỗi cây.
Mỗi cây sử dụng tập con các đặc trưng tại mỗi nút để tăng tính đa dạng.
Kết quả cuối cùng được xác định qua bỏ phiếu (voting) hoặc trung bình.
Ưu điểm, giảm overfitting so với cây quyết định đơn lẻ. Hiệu quả với dữ liệu phi tuyến tính hoặc mất cân bằng. Xử lý tốt các đặc trưng tương quan cao.
Hạn chế, tốn nhiều tài nguyên tính toán với dữ liệu lớn.Mất khả năng diễn giải trực quan khi số lượng cây tăng cao.

2.3.4. XGBoost (Extreme Gradient Boosting)

XGBoost là một thuật toán tăng cường độ dốc (Gradient Boosting) hiệu quả cao, tối ưu hóa cả tốc độ lẫn độ chính xác.
Cơ chế hoạt động: Mỗi cây mới được thêm vào nhằm giảm lỗi của các cây trước đó. Sử dụng Regularized Loss Function để giảm overfitting.
Hàm lỗi của XGBoost:

$$L = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

trong đó $\Omega(f_k)$ là hàm điều chỉnh để tránh overfitting.

Các tính năng chính:
Hỗ trợ xử lý dữ liệu thiếu và không yêu cầu xử lý trước kỹ lưỡng.
Tối ưu hóa gradient nhanh chóng bằng cách song song hóa.
Tự động điều chỉnh trọng số cho các lớp dữ liệu mất cân bằng.
Ưu điểm, khả năng dự đoán cao trong bài toán phức tạp. Linh hoạt với nhiều tham số tinh chỉnh. Hạn chế, tốn thời gian tinh chỉnh tham số. Cần nhiều tài nguyên tính toán nếu tập dữ liệu lớn.

Thuật toán	Độ chính xác	Khả năng giải thích	Nhạy cảm với dữ liệu mất cân bằng	Tính phức tạp
Logistic Regression	Trung bình	Cao	Cao	Thấp
Decision Tree	Trung bình	Cao	Trung bình	Thấp
Random Forest	Cao	Trung bình	Thấp	Trung bình
XGBoost	Cao nhất	Thấp	Thấp	Cao

Hình 2.1: So sánh tổng quan giữa các thuật toán

2.4. Giải quyết vấn đề mất cân bằng dữ liệu

2.4.1. Xử Lý Dữ Liệu Không Cân Bằng (Imbalance Handling)

Phương pháp cơ bản để xử lý mất cân bằng lớp bao gồm:

- `Random Under-Sampling`: Loại bỏ một phần dữ liệu từ lớp chiếm ưu thế để cân bằng với lớp thiểu số.
- `Random Over-Sampling`: Nhân bản các mẫu từ lớp thiểu số để cân bằng với lớp chiếm ưu thế.

2.4.2. SMOTE (Synthetic Minority Oversampling Technique)

SMOTE là một phương pháp hiện đại để xử lý mất cân bằng lớp bằng cách tạo ra các mẫu mới từ lớp thiểu số thay vì nhân bản trực tiếp.

Nguyên lý hoạt động:

- Chọn ngẫu nhiên một mẫu từ lớp thiểu số và xác định các hàng xóm gần nhất (k-nearest neighbors).

- Tạo ra các điểm dữ liệu mới bằng cách nội suy tuyến tính giữa mẫu được chọn và một hàng xóm gần nhất.

Ưu điểm là có thể giảm nguy cơ overfitting so với Random Over-Sampling.Tăng cường dữ liệu lớp thiểu số một cách tinh tế và hiệu quả.
Tuy nhiên vẫn còn hạn chế không hoạt động tốt nếu dữ liệu có nhiều nhiễu hoặc phân bố không đồng nhất. Tạo ra các mẫu mới có thể không phản ánh đúng bản chất của dữ liệu thực.

2.4.3. ADASYN (Adaptive Synthetic Sampling)

ADASYN là một mở rộng của SMOTE, tập trung tạo nhiều mẫu giả lập hơn cho các điểm khó phân loại. Mục tiêu chính là giảm độ lệch phân phối giữa các lớp bằng cách ưu tiên tăng cường những vùng dữ liệu bị thiểu số nghiêm trọng.

Nguyên lý hoạt động:

- Tính khoảng cách Euclidean để tìm hàng xóm gần nhất của từng mẫu trong lớp thiểu số.
- Xác định trọng số (weight) cho từng mẫu dựa trên độ khó phân loại (tức là các điểm gần biên quyết định hơn sẽ được tạo nhiều mẫu hơn).
- Tạo các mẫu mới dựa trên trọng số này.

Ưu điểm, tăng hiệu quả của mô hình tại các vùng biên khó phân loại. Phù hợp với các bài toán dữ liệu phức tạp và phân bố không đồng nhất. Hạn chế, phức tạp hơn về tính toán so với SMOTE. Có thể gây ra sự mất cân bằng cục bộ nếu không điều chỉnh tốt các tham số.

3. Tập Dữ Liệu

3.1. Khám phá dữ liệu

3.1.1. Nguồn gốc dữ liệu

Bộ dữ liệu được sử dụng đến từ Kaggle, với tiêu đề Credit Card Fraud Detection. Đây là một tập dữ liệu về các giao dịch thẻ tín dụng được thực hiện vào tháng 9 năm 2013 bởi các chủ thẻ ở Châu Âu. Dữ liệu bao gồm các giao dịch diễn ra trong vòng hai ngày, với tổng cộng 284,807 giao dịch, trong đó có 492 giao dịch gian lận.

3.1.2. Tổng quan dữ liệu

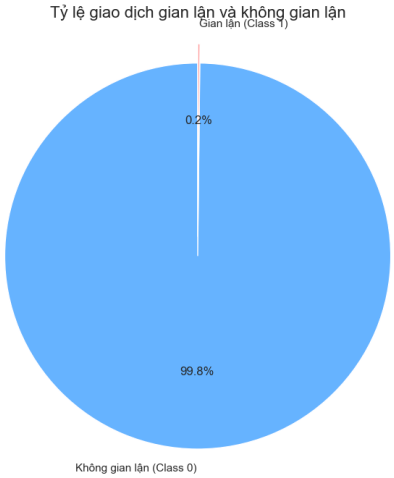
Tính chất dữ liệu:
Đây là một bộ dữ liệu mất cân bằng nghiêm trọng, với nhóm giao dịch gian lận (lớp 1) chỉ chiếm 0.172% tổng số giao dịch.

Cấu trúc dữ liệu:

- Số lượng mẫu: 284,807.
- Số lượng đặc trưng: 31.
- Các đặc trưng từ V1 đến V28 là các thành phần chính được rút trích từ Phân tích Thành phần Chính (PCA).
- Time: Số giây đã trôi qua kể từ giao dịch đầu tiên.
- Amount: Số tiền của giao dịch.
- Class: Biến mục tiêu, với 0 là giao dịch hợp lệ, 1 là giao dịch gian lận.

3.1.3. Phân tích dữ liệu

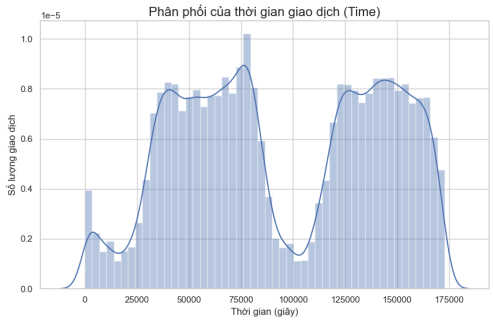
1. Tỷ lệ giao dịch gian lận và không gian lận Bộ dữ liệu có sự mất cân bằng nghiêm trọng giữa các giao dịch gian lận (class 1) và không gian lận (class 0). Dưới đây là biểu đồ thể hiện tỷ lệ phần trăm giao dịch của hai nhóm:



Hình 3.1: Tỷ lệ giao dịch gian lận và không gian lận

Kết quả phân tích:
Lớp 0 chiếm khoảng 99.8% tổng số giao dịch, trong khi lớp 1 (gian lận) chỉ chiếm 0.2%.

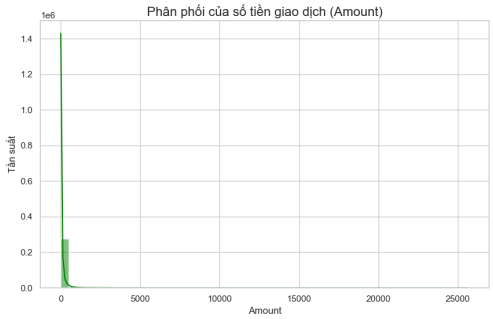
2. Phân phối của thời gian giao dịch (Time) Đặc trưng Time biểu diễn thời gian đã trôi qua tính bằng giây kể từ giao dịch đầu tiên. Biểu đồ dưới đây thể hiện phân phối của đặc trưng này:



Hình 3.2: Phân phối của thời gian giao dịch (Time)

Nhận xét:
Phân phối thời có quy luật rõ ràng. Biểu đồ cho thấy có 2 khoảng thời gian giao dịch chính. Các giao dịch được thực trong khoảng hai ngày. Hai đỉnh giao dịch có thể là vào ban ngày, còn hai khu vực trũng có thể là ban đêm. Phân phối có sự khác biệt lớn, có thể dẫn tới ảnh hưởng giữa các giao dịch gian lận và không gian lận khi xét về thời gian thực hiện.

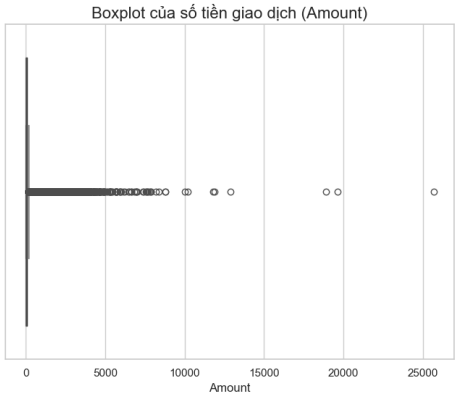
3. Phân phối của số tiền giao dịch (Amount) Đặc trưng Amount biểu diễn số tiền của từng giao dịch. Biểu đồ dưới đây thể hiện phân phối của đặc trưng này:



Hình 3.3: Phân phối của số tiền giao dịch (Amount)

Nhận xét:
Phần lớn giao dịch có số tiền nhỏ hơn $100. Một số giao dịch có số tiền rất lớn, tạo thành các giá trị ngoại lai (outliers).

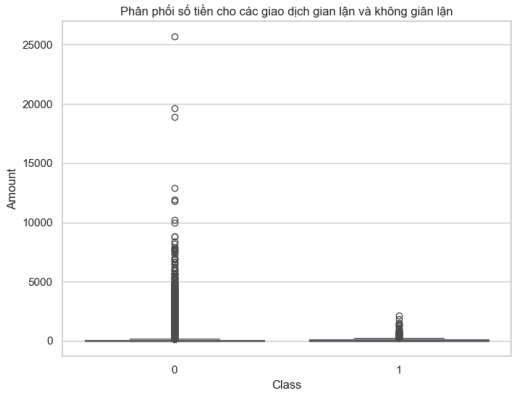
4. Phân tích ngoại lai của số tiền giao dịch (Amount) Biểu đồ dưới đây sử dụng boxplot để minh họa các giá trị ngoại lai trong đặc trưng Amount:



Hình 3.4: Phân tích ngoại lai của số tiền giao dịch (Amount)

Nhận xét:
Các giao dịch có số tiền lớn hơn $10,000 được coi là ngoại lai. Tuy nhiên, các giao dịch ngoại lai không nhất thiết là gian lận; cần phân tích sâu hơn về sự tương quan với biến mục tiêu (Class).

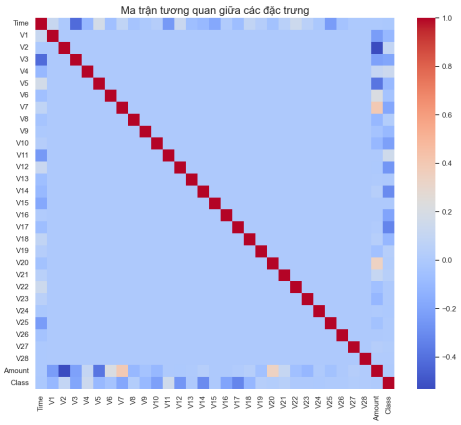
Để phân tích kĩ hơn về các giao dịch ngoại lai có phải là giao dịch gian lận không, ta xem biểu đồ bên dưới:



Hình 3.5: Phân tích ngoại lai giữa Amount và Class

Nhận xét:
Biểu đồ cho thấy các nhãn giao dịch là gian lận tập chung ở phần có số tiền giao dịch thấp. Điều này chứng tỏ dữ liệu ngoại lai không gây ảnh hưởng đến nhãn gian lận. Ta có thể sẽ loại bỏ dữ liệu có số tiền giao dịch quá lớn vì nó có ảnh hưởng không tốt đến việc huấn luyện mô hình sau này.

5. Ma trận tương quan giữa các đặc trưng Biểu đồ heatmap dưới đây thể hiện mối tương quan giữa các đặc trưng trong tập dữ liệu:



Hình 3.6: Ma trận tương quan giữa các đặc trưng

Nhận xét:
Phần lớn các đặc trưng có tương quan thấp, điều này có thể do dữ liệu đã được giảm chiều bằng PCA. Một số đặc trưng có mối tương quan âm hoặc dương đáng kể, có thể ảnh hưởng đến hiệu suất mô hình nếu không được xử lý cẩn thận.

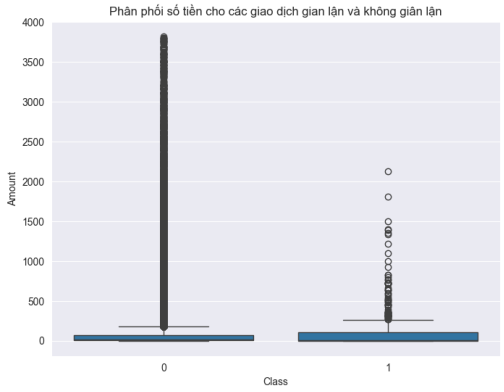
3.2. Xử lý dữ liệu cơ bản

3.2.1. Xử lý thiếu dữ liệu

Tập dữ liệu được kiểm tra để phát hiện các giá trị thiếu. Kết quả cho thấy không có đặc trưng nào trong dữ liệu bị thiếu giá trị.

3.2.2. Xử lý dữ liệu ngoại lệ

Đặc trưng Amount (Số tiền giao dịch) được kiểm tra ngoại lệ bằng biểu đồ boxplot bên trên. Các giao dịch có số tiền quá lớn (nằm ngoài râu của boxplot) được xác định là ngoại lệ.
Loại bỏ ngoại lệ:
Những giao dịch này có thể làm sai lệch kết quả mô hình, do đó, các mẫu chứa giá trị ngoại lệ sẽ được loại bỏ.



Hình 3.7: Boxplot của Amount trước và sau khi loại bỏ ngoại lệ

Boxplot được sử dụng để thể hiện rõ các ngoại lệ trước và sau khi xử lý.

3.2.3. Chuẩn hóa dữ liệu

Để cải thiện hiệu quả của các thuật toán học máy, đặc biệt với các đặc trưng có phân phối không đều hoặc lệch chuẩn, ba phương pháp chuẩn hóa được áp dụng và so sánh:

1. Log Transformation Mục tiêu: Giảm độ lệch của phân phối, đặc biệt hữu ích với các đặc trưng có phân phối lệch phải (skewed distribution). Log Transformation thể hiện phân phối cân đối hơn so với trước xử lý.

Công thức áp dụng:

$$Amount_{log} = \log(Amount + 0.0001)$$

(Thêm 0.0001 để tránh giá trị 0 khi Amount = 0.)

2. Standardization Mục tiêu: Chuyển dữ liệu về phân phối chuẩn (mean = 0, std = 1). Standardization giúp phân phối tập trung hơn quanh giá trị trung bình.

Công thức áp dụng:

$$Amount_{std} = \frac{Amount - \mu}{\sigma}$$

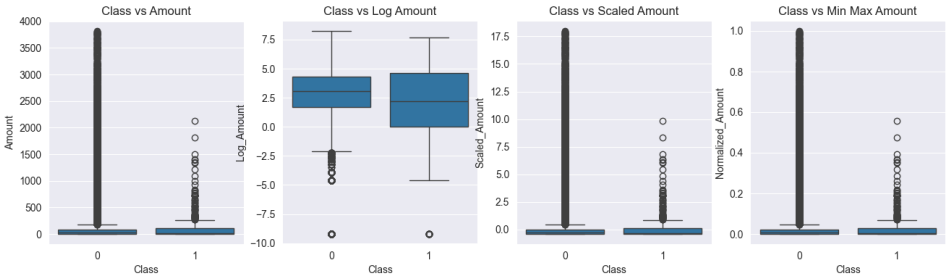
Trong đó:
μ: Giá trị trung bình.
σ: Độ lệch chuẩn.

3. Normalization (Min-Max Scaling) Mục tiêu: Chuyển dữ liệu về khoảng giá trị [0, 1]. Amount sau khi chuẩn hóa Min-Max Scaling thể hiện toàn bộ giá trị được đưa về khoảng [0, 1].

Công thức áp dụng:

$$Amount_{norm} = \frac{Amount - \min(Amount)}{\max(Amount) - \min(Amount)}$$

Minh họa bằng biểu đồ:



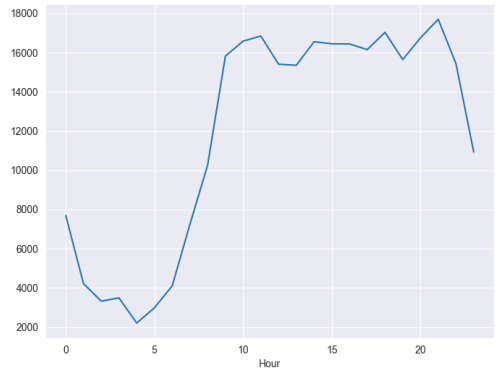
Hình 3.8: So sánh các dạng chuẩn hoá dữ liệu

3.2.4. Tạo các biến mới

Biến Day và Hour được tạo ra từ cột Time có sẵn trong dữ liệu. Biến Time thể hiện số giây đã trôi qua kể từ giao dịch đầu tiên trong tập dữ liệu. Chúng tôi sử dụng giá trị này để trích xuất hai biến mới:

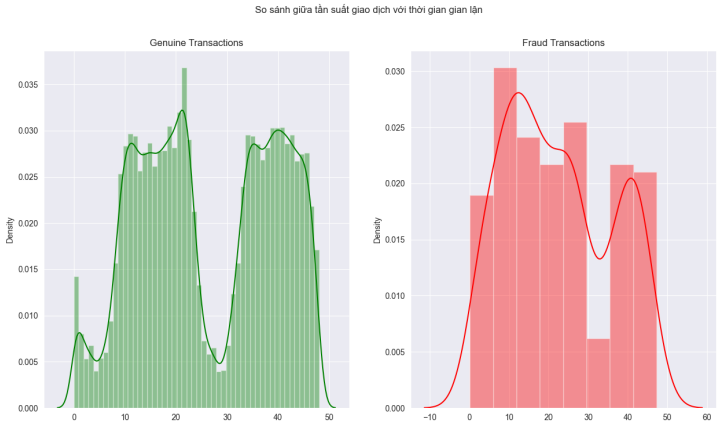
- Biến Day: Là số ngày tính từ ngày đầu tiên trong bộ dữ liệu. Biến này giúp xác định giao dịch diễn ra vào ngày nào.
- Biến Hour: Là giờ trong ngày (từ 0 đến 23) mà giao dịch được thực hiện. Biến này cho phép phân tích chi tiết hơn về các giao dịch theo thời gian trong ngày.

Một trong những mục tiêu quan trọng là phân tích sự phân bố của các giao dịch gian lận theo giờ trong ngày. Việc này có thể giúp phát hiện các mô hình gian lận theo thời gian, ví dụ như gian lận thường xuyên xảy ra vào các giờ cụ thể.



Hình 3.9: Phân bố của các giao dịch gian lận theo giờ

Chúng tôi cũng phân tích sự khác biệt giữa giao dịch gian lận và không gian lận theo giờ trong ngày. Phân tích này giúp nhận ra các giờ cao điểm của giao dịch gian lận, đồng thời cho thấy sự phân bố của các giao dịch không gian lận.



Hình 3.10: Tần suất giao dịch gian lận và không gian lận

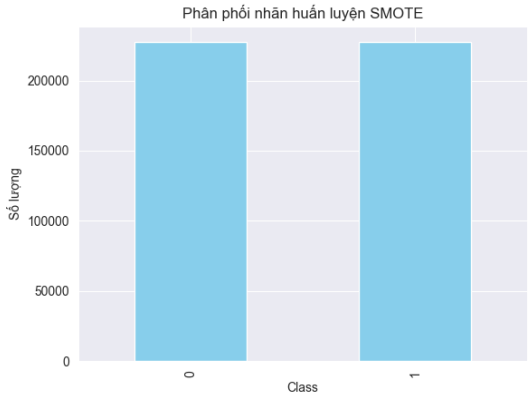
Biểu đồ này hiển thị tần suất giao dịch theo giờ trong ngày, với sự phân biệt giữa giao dịch gian lận và không gian lận. Nó giúp nhận diện các giờ có tỷ lệ gian lận cao hơn, qua đó hỗ trợ việc tối ưu hóa chiến lược giám sát.

3.3. Xử lý dữ liệu mất cân bằng

3.3.1. Dữ liệu gốc (Imbalanced)

Trong trường hợp này, chúng tôi giữ nguyên tập dữ liệu ban đầu mà không áp dụng bất kỳ kỹ thuật cân bằng nào. Tập dữ liệu này vẫn giữ tỷ lệ rất thấp của các giao dịch gian lận so với các giao dịch không gian lận (0.2%). Việc sử dụng tập dữ liệu này cho phép chúng tôi thấy được hiệu suất của các mô hình khi làm việc với dữ liệu mất cân bằng.

3.3.2. Xử lý dữ liệu bằng SMOTE



Hình 3.11: Xử lý dữ liệu bằng SMOTE

Sau khi áp dụng SMOTE, tỷ lệ giao dịch gian lận và không gian lận sẽ gần bằng nhau. Điều này giúp mô hình học được cách phân biệt các giao dịch gian lận và không gian lận một cách hiệu quả hơn.

3.3.3. Xử lý dữ liệu bằng ADASYN

Biểu đồ dưới đây thể hiện sự phân bố của các giao dịch gian lận và không gian lận sau khi áp dụng ADASYN. Tương tự như SMOTE, ADASYN giúp tạo ra các điểm dữ liệu gian lận mới, làm tăng tỷ lệ gian lận trong dữ liệu.

Bằng cách sử dụng SMOTE và ADASYN, chúng tôi có thể cải thiện tỷ lệ giao dịch gian lận

Hình 3.12: Xử lý dữ liệu bằng ADASYN

trong tập huấn luyện, từ đó nâng cao khả năng phân loại các giao dịch gian lận của mô hình học máy. Các kỹ thuật này giúp mô hình không bị "thiên lệch" vào lớp không gian lận và đảm bảo rằng các giao dịch gian lận được phát hiện một cách chính xác hơn.

3.4. Phân chia dữ liệu

Phân chia dữ liệu là một bước quan trọng trong quá trình huấn luyện và đánh giá mô hình học máy. Mục đích của việc phân chia dữ liệu là tạo ra các tập con dữ liệu để huấn luyện mô hình, kiểm tra mô hình và đánh giá kết quả. Việc chia đúng tỷ lệ giữa các tập dữ liệu sẽ giúp mô hình học một cách hiệu quả và tránh tình trạng overfitting hoặc underfitting.

Trong trường hợp bài toán phát hiện gian lận thẻ tín dụng, chúng tôi thực hiện phân chia dữ liệu thành hai tập chính:

- Tập huấn luyện (Training Set): 80% dữ liệu.
- Tập kiểm tra (Test Set): 20% dữ liệu

Tuy nhiên, do dữ liệu có sự mất cân bằng giữa các lớp (gian lận và không gian lận), chúng tôi sẽ thực hiện phân chia sao cho tỷ lệ gian lận trong tập huấn luyện và tập kiểm tra được duy trì giống nhau. Điều này giúp đảm bảo mô hình có thể học được các đặc trưng của cả hai lớp mà không bị thiên lệch.

Biểu đồ dưới đây thể hiện sự phân chia tỷ lệ gian lận và không gian lận trong dữ liệu sau khi áp dụng phân chia. Như có thể thấy, tỷ lệ gian lận trong tập huấn luyện và tập kiểm tra là tương đồng, giúp mô hình học được các đặc trưng của cả hai lớp một cách hiệu quả.

Hình 3.13: Tỷ lệ gian lận và không gian lận trong dữ liệu huấn luyện và kiểm tra

4. Xây Dựng Mô Hình Học Máy

Trong phần này, chúng tôi sẽ trình bày các mô hình học máy được sử dụng để giải quyết bài toán xác định gian lận tín dụng. Mục tiêu là xây dựng các mô hình có khả năng phân loại chính xác các giao dịch gian lận và hợp pháp dựa trên ba tập dữ liệu khác nhau: dữ liệu không cân bằng (imbalance), dữ liệu đã qua xử lý với SMOTE và dữ liệu đã qua xử lý với ADASYN.

4.1. Lựa Chọn Mô Hình Học Máy

Mô hình học máy được lựa chọn dựa trên đặc điểm của bài toán và yêu cầu về độ chính xác trong việc phát hiện gian lận tín dụng. Dưới đây là các mô hình được chọn để xây dựng và so sánh:

4.1.1. Logistic Regression (Baseline Model)

Logistic Regression được chọn làm mô hình cơ bản (baseline model) cho bài toán phân loại gian lận tín dụng. Đây là một mô hình đơn giản nhưng rất hiệu quả trong các bài toán phân loại nhị phân. Logistic Regression dự đoán xác suất của một giao dịch là gian lận dựa trên các đặc trưng của nó. Mặc dù mô hình này có thể không đạt độ chính xác cao nhất trong môi trường dữ liệu phức tạp, nhưng nó cung cấp một nền tảng cơ bản để so sánh với các mô hình phức tạp hơn.

4.1.2. Decision Tree

Decision Tree là một thuật toán học máy dễ hiểu và dễ giải thích, dựa trên nguyên lý phân tách dữ liệu thành các nhánh theo các thuộc tính. Mặc dù Decision Tree dễ bị overfitting khi dữ liệu có quá nhiều đặc trưng, nhưng khi được kết hợp với các phương pháp khác như Random Forest, nó có thể hoạt động hiệu quả. Mô hình này có khả năng xử lý các vấn đề phân loại nhị phân và có thể áp dụng trong bài toán xác định gian lận tín dụng.

4.1.3. Random Forest

Random Forest là một thuật toán học máy mạnh mẽ, dựa trên việc kết hợp nhiều cây quyết định (decision trees) để cải thiện độ chính xác của mô hình. Bằng cách xây dựng và huấn luyện nhiều cây quyết định từ các mẫu dữ liệu khác nhau, Random Forest có thể giảm thiểu hiện tượng overfitting và cải thiện độ chính xác trong việc phân loại các giao dịch gian lận. Đây là một lựa chọn phổ biến khi dữ liệu có nhiều biến và mối quan hệ phi tuyến.

4.1.4. XGBoost

XGBoost là một trong những thuật toán học máy mạnh mẽ nhất hiện nay, đặc biệt hiệu quả trong các bài toán phân loại với dữ liệu không cân bằng. Thuật toán này sử dụng phương pháp boosting, giúp tối ưu hóa hiệu quả của mô hình qua từng vòng học. XGBoost có thể xử lý tốt các mối quan hệ phi tuyến và cung cấp các công cụ để điều chỉnh các tham số siêu cấu hình, giúp nâng cao độ chính xác trong việc phát hiện gian lận tín dụng.

4.2. Quá Trình Huấn Luyện Mô Hình

Để xây dựng và huấn luyện các mô hình, chúng tôi sử dụng ba tập dữ liệu khác nhau:

- **Tập dữ liệu không cân bằng (Imbalance)**: Đây là tập dữ liệu gốc, trong đó tỷ lệ các giao dịch gian lận thấp hơn nhiều so với các giao dịch hợp pháp. Dữ liệu không cân bằng có thể làm giảm độ chính xác của mô hình, đặc biệt là trong các bài toán phân loại nhị phân như xác định gian lận tín dụng.
- **Tập dữ liệu với SMOTE**: Để xử lý vấn đề dữ liệu không cân bằng, chúng tôi sử dụng kỹ thuật SMOTE (Synthetic Minority Over-sampling Technique). SMOTE tạo ra các mẫu mới từ lớp thiểu số (gian lận) bằng cách kết hợp các điểm dữ liệu gần nhau, từ đó giúp cân bằng tỷ lệ giữa các lớp và cải thiện hiệu quả mô hình.

- **Tập dữ liệu với ADASYN**: ADASYN (Adaptive Synthetic Sampling) là một kỹ thuật khác để xử lý dữ liệu không cân bằng, tương tự như SMOTE. Tuy nhiên, ADASYN tập trung vào việc tạo ra các điểm dữ liệu bổ sung gần các điểm biên, nơi mà sự phân biệt giữa các lớp có thể khó khăn hơn. Kỹ thuật này giúp cải thiện khả năng phân loại của mô hình trong các trường hợp có sự chênh lệch lớn giữa các lớp.

4.3. Các Bước Xây Dựng Mô Hình

Các bước xây dựng các mô hình học máy như sau:

1. **Tiền xử lý dữ liệu**: Bao gồm việc xử lý dữ liệu thiếu, chuẩn hóa dữ liệu và chia tập dữ liệu thành tập huấn luyện và tập kiểm tra.
2. **Xử lý dữ liệu không cân bằng**: Áp dụng các phương pháp SMOTE và ADASYN để tạo ra các mẫu dữ liệu mới từ lớp thiểu số, giúp cân bằng dữ liệu và giảm thiểu tác động của vấn đề không cân bằng trong việc huấn luyện mô hình.
3. **Huấn luyện mô hình**: Áp dụng các thuật toán Logistic Regression, XGBoost, Decision Tree và Random Forest để huấn luyện mô hình trên các tập dữ liệu đã qua xử lý.
4. **Đánh giá mô hình**: Sử dụng các chỉ số như Accuracy, Precision, Recall và F1-score để đánh giá hiệu quả của các mô hình. Các chỉ số này giúp so sánh các mô hình với nhau và chọn ra mô hình có hiệu suất tốt nhất.

5. Kết Quả Và Phân Tích

5.1. Kết quả đánh giá mô hình

Bảng kết quả dưới đây trình bày các chỉ số đánh giá của bốn mô hình học máy (Logistic Regression, Decision Tree, Random Forest và XGBoost) trên ba loại dữ liệu: dữ liệu không cân bằng (imbalance), dữ liệu sau khi áp dụng SMOTE và ADASYN. Các chỉ số bao gồm Độ chính xác (Accuracy), Độ chính xác (Precision), Tỷ lệ hồi đáp (Recall), Điểm F1 (F1 Score), và Diện tích dưới đường cong ROC (ROC-AUC).

Model	Data Type	Accuracy	Precision	Recall	F1 Score	ROC-AUC
logistic	imbalance	0.9991	0.836	0.571	0.679	0.9741
decision_tree	imbalance	0.9990	0.703	0.724	0.714	0.8620
random_forest	imbalance	0.9994	0.884	0.776	0.826	0.9526
xgboost	imbalance	0.9994	0.889	0.735	0.804	0.9817
logistic	smote	0.9784	0.068	0.908	0.127	0.9812
decision_tree	smote	0.9973	0.365	0.745	0.490	0.8713
random_forest	smote	0.9993	0.819	0.786	0.802	0.9629
xgboost	smote	0.9991	0.705	0.806	0.752	0.9741
logistic	adasyn	0.9384	0.025	0.929	0.049	0.9797
decision_tree	adasyn	0.9975	0.385	0.714	0.500	0.8562
random_forest	adasyn	0.9993	0.833	0.765	0.798	0.9737
xgboost	adasyn	0.9991	0.708	0.816	0.758	0.9744

Bảng 5.1: Kết quả đánh giá các mô hình học máy

5.2. Phân tích hiệu quả và so sánh mô hình

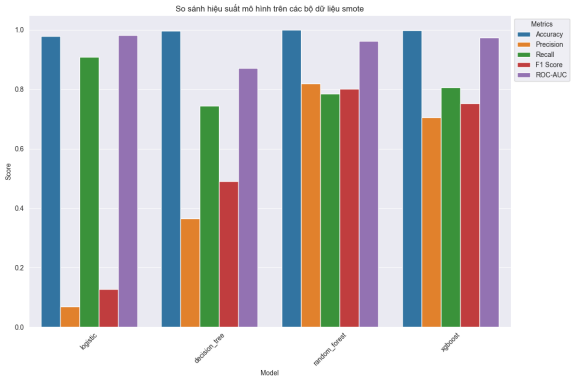
Kết quả mô phỏng cho thấy các mô hình học máy có sự khác biệt rõ rệt khi áp dụng trên ba loại dữ liệu: imbalanced (mất cân bằng), SMOTE, và ADASYN. Mỗi phương pháp đều có ưu điểm và hạn chế riêng, đặc biệt là trong bối cảnh phát hiện gian lận thẻ tín dụng.

5.2.1. Dữ liệu imbalance

Các mô hình Logistic Regression, Decision Tree, Random Forest và XGBoost đều có độ chính xác (Accuracy) rất cao (gần 100%) khi áp dụng trên dữ liệu không cân bằng. Tuy nhiên, độ chính xác này không phản ánh đúng hiệu quả thực tế vì các mô hình này chủ yếu dự đoán lớp không gian lận (0) mà bỏ qua lớp gian lận (1), do lớp gian lận rất nhỏ.
Tỷ lệ hồi đáp (Recall) cho thấy một thực tế đáng lo ngại: các mô hình đều không nhận diện

tốt gian lận. Ví dụ, Logistic Regression chỉ đạt Recall là 0.571, điều này có nghĩa là chỉ có khoảng 57% giao dịch gian lận được phát hiện. Điều này chỉ ra rằng việc phát hiện gian lận trong một bộ dữ liệu mất cân bằng như vậy cực kỳ khó khăn.
Điểm F1 (F1 Score) cho thấy sự chênh lệch giữa Precision và Recall, giúp chúng ta thấy rõ hiệu quả của mô hình khi tối ưu hóa cho sự cân bằng giữa độ chính xác và tỷ lệ hồi đáp.

5.2.2. Dữ liệu SMOTE

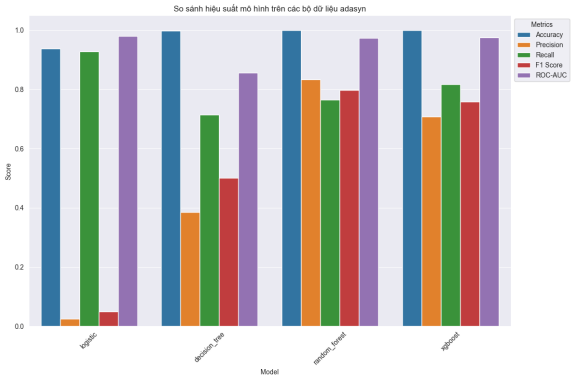


Hình 5.1: Biểu đồ so sánh các chỉ số cho dữ liệu SMOTE

Khi áp dụng SMOTE (Synthetic Minority Over-sampling Technique) để xử lý mất cân bằng lớp, tỷ lệ hồi đáp (Recall) của các mô hình được cải thiện rõ rệt. Mặc dù độ chính xác giảm (Accuracy thấp hơn), các mô hình như Logistic Regression có thể nhận diện gian lận tốt hơn (Recall 0.908), với tỷ lệ hồi đáp tăng mạnh nhưng đồng thời cũng làm giảm độ chính xác.
Các mô hình như Random Forest và XGBoost duy trì hiệu quả tương đối ổn định với độ chính xác cao và Recall cải thiện. F1 Score cho thấy sự cân bằng giữa Precision và Recall được cải thiện, đặc biệt là ở Random Forest và XGBoost.

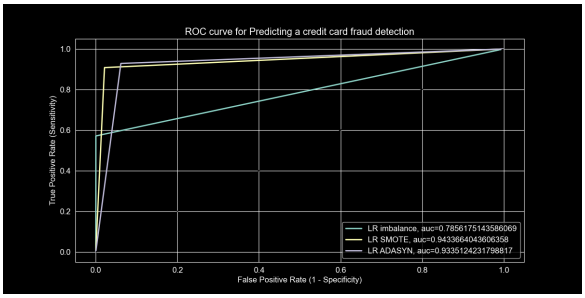
5.2.3. Dữ liệu ADASYN

ADASYN (Adaptive Synthetic Sampling) tiếp tục giúp cải thiện tỷ lệ hồi đáp (Recall) cho các mô hình. Tuy nhiên, Logistic Regression với ADASYN lại có một sự sụt giảm mạnh về độ chính xác (Accuracy chỉ đạt 0.9384), trong khi Random Forest và XGBoost có sự cải thiện rõ rệt về khả năng phát hiện gian lận (Recall 0.765 và 0.816).
Điểm F1 với ADASYN cho thấy khả năng cải thiện sự cân bằng giữa Precision và Recall, mặc dù vẫn chưa thể vượt qua kết quả của Random Forest và XGBoost.



Hình 5.2: Biểu đồ so sánh các chỉ số cho dữ liệu ADASYN

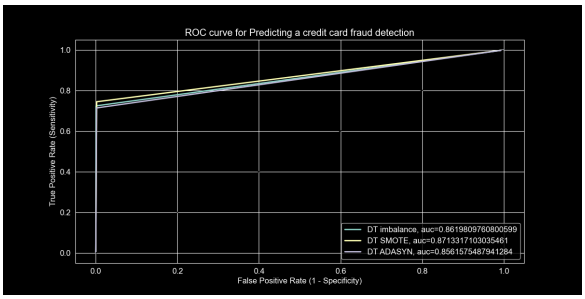
5.3. So sánh chỉ số ROC - AUC của các mô hình



Hình 5.3: Biểu đồ ROC cho Logistic Regression

Biểu đồ này cho thấy hiệu quả của Logistic Regression trên ba loại dữ liệu. Chúng ta có thể nhận thấy rằng:

Dữ liệu imbalance làm cho mô hình phân loại kém hơn, với ROC-AUC thấp.
Sau khi áp dụng SMOTE và ADASYN, khả năng phân loại gian lận của mô hình được cải thiện rõ rệt với sự tăng trưởng của ROC-AUC, đặc biệt là trên dữ liệu ADASYN.

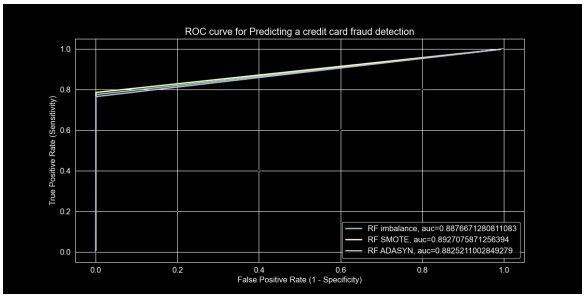


Hình 5.4: Biểu đồ ROC cho Decision Tree

Biểu đồ ROC cho Decision Tree cho thấy:

Trên dữ liệu imbalance, diện tích dưới đường cong (ROC-AUC) khá cao nhưng không đạt mức tối ưu do tỷ lệ hồi đáp thấp.

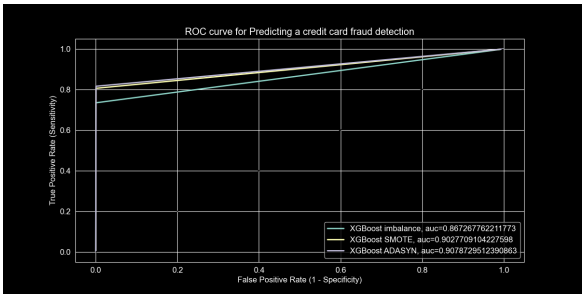
Khi áp dụng SMOTE và ADASYN, khả năng phân loại tăng lên đáng kể, đặc biệt là trên ADASYN với ROC-AUC cải thiện mạnh mẽ.



Hình 5.5: Biểu đồ ROC cho Random Forest

Biểu đồ ROC cho Random Forest cho thấy:

Random Forest duy trì hiệu quả rất tốt ngay cả trên dữ liệu mất cân bằng, với ROC-AUC cao (gần 1).
Sau khi áp dụng SMOTE và ADASYN, mô hình này duy trì hiệu quả ổn định, và ROC-AUC vẫn rất cao.



Hình 5.6: Biểu đồ ROC cho XGBoost

XGBoost có hiệu quả cao trên dữ liệu imbalance, nhưng có sự cải thiện đáng kể khi áp dụng SMOTE và ADASYN. Mô hình này đạt được ROC-AUC rất cao trên cả ba loại dữ liệu, đặc biệt trên ADASYN, cho thấy khả năng phân loại tốt hơn trong việc phát hiện gian lận thẻ tín dụng.

5.4. Những thách thức trong phát hiện gian lận thẻ tín dụng

Việc phát hiện gian lận thẻ tín dụng là một bài toán rất thách thức, chủ yếu do:

- `Mất cân bằng lớp dữ liệu:` Lớp gian lận chỉ chiếm một tỷ lệ rất nhỏ trong tổng số giao dịch, điều này gây khó khăn trong việc phát hiện gian lận. Các mô hình học máy dễ dàng dự đoán giao dịch không gian lận, nhưng việc nhận diện gian lận là rất khó khăn.

- `Vấn đề với dữ liệu thiếu hoặc không đầy đủ:` Bộ dữ liệu này không cung cấp các thông tin gốc, chỉ có các thành phần chính thu được từ PCA, điều này khiến cho việc giải thích các mô hình trở nên phức tạp hơn.
- `Các phương pháp xử lý mất cân bằng:` Các phương pháp như SMOTE và ADASYN giúp cải thiện tỷ lệ hồi đáp nhưng có thể làm giảm độ chính xác. Cần phải cân nhắc kỹ khi áp dụng các kỹ thuật này, vì chúng có thể gây ra sự thay đổi không mong muốn trong phân phối lớp.
- `Mô hình học máy không giải thích được:` Các mô hình như Random Forest và XGBoost dù hiệu quả, nhưng khó giải thích rõ ràng lý do tại sao chúng đưa ra dự đoán nhất định, điều này là một vấn đề lớn trong các ứng dụng thực tế như phát hiện gian lận.

6. Ứng Dụng Và Triển Khai

6.1. Ứng dụng mô hình trên dữ liệu thực tế

Việc ứng dụng các mô hình học máy vào dữ liệu thực tế giúp tăng cường khả năng dự đoán, phân tích và tối ưu hóa trong nhiều ngành công nghiệp. Những mô hình này có thể tự động hóa quy trình, hỗ trợ ra quyết định chính xác, và cải thiện hiệu quả hoạt động, mang lại lợi ích lớn cho các doanh nghiệp và tổ chức trong các lĩnh vực khác nhau.

6.2. Đề xuất cải tiến và ứng dụng thực tế

Việc cải tiến hiệu suất của mô hình sẽ làm tăng độ chính xác nhắm giúp các tổ chức tài chính có thể đưa ra quyết định chính xác nhắm giảm thiểu rủi ro về các giao dịch tín dụng gian lận, dưới đây là một số đề xuất cải tiến:
1. Cải thiện chất lượng dữ liệu:

- Thu thập thêm đặc trưng: Bổ sung các thông tin như lịch sử công việc, loại hợp đồng lao động, điểm tín dụng, và thông tin tài sản đảm bảo.
- Cập nhật dữ liệu định kỳ: Đảm bảo mô hình luôn được huấn luyện trên dữ liệu mới nhất để phản ánh đúng thị trường.
- Xử lý dữ liệu thiếu và bất thường: Sử dụng các kỹ thuật ước lượng dữ liệu thiếu bằng phương pháp KNN Imputation hoặc Multiple Imputation.

2. Tối ưu hóa mô hình:

- Tìm kiếm siêu tham số (Hyperparameter Tuning): Sử dụng Grid Search hoặc Bayesian Optimization để tìm cấu hình tối ưu cho các mô hình như Random Forest và XGBoost.
- Kết hợp nhiều mô hình (Ensemble Learning): Áp dụng các kỹ thuật như Stacking hoặc Blending để kết hợp nhiều mô hình nhằm cải thiện độ chính xác..
- Học chuyển giao (Transfer Learning): Tận dụng mô hình đã được huấn luyện từ các tổ chức khác hoặc trên tập dữ liệu tương tự.

3. Xử lý mất cân bằng dữ liệu nâng cao:

- Tích hợp nhiều kỹ thuật resampling: Kết hợp SMOTE với Tomek Links hoặc sử dụng kỹ thuật Balanced Random Forest để giảm thiểu lỗi do dữ liệu nhiễu.
- Focal Loss: Thay đổi hàm mất mát để tập trung vào các mẫu khó phân loại, giúp tăng hiệu suất của mô hình trên lớp thiểu số.

6.3. Khả năng mở rộng và triển khai trong môi trường sản xuất

Việc dự đoán các gian lận tài chính nói chung cũng như trong việc gian lận giao dịch tín dụng nói riêng đem lại lợi ích rất lớn vậy nên tiềm năng để mở rộng và triển khai trong môi trường sản xuất cũng rất lớn đặc biệt trong bối cảnh phát triển của thế giới ngày nay. Một số cách để mở rộng dự án và triển khai:
1. Khả năng mở rộng:

- Sử dụng hệ thống phân tán: Áp dụng công nghệ như Apache Spark hoặc Hadoop để xử lý tập dữ liệu lớn.
- Dịch vụ đám mây: Sử dụng dịch vụ AWS, Google Cloud, hoặc Azure để linh hoạt mở rộng tài nguyên theo nhu cầu thực tế.
- Huấn luyện mô hình phân tán: Sử dụng XGBoost và LightGBM với hỗ trợ phân tán để tăng tốc độ huấn luyện trên nhiều máy chủ.
- Microservices architecture: Tách biệt các thành phần của hệ thống thành các dịch vụ nhỏ (microservices) giúp dễ dàng quản lý và mở rộng.
- Auto-scaling: Cấu hình tự động tăng/giảm tài nguyên dựa trên khối lượng công việc trong thời gian thực.

2. Triển khai trong môi trường sản xuất

- Triển khai qua API: Đóng gói mô hình thành REST API hoặc gRPC với các framework như Flask, FastAPI hoặc TensorFlow Serving.
- CI/CD pipelines: Sử dụng các công cụ như Jenkins, GitLab CI/CD để tự động hóa quá trình triển khai mô hình từ phát triển đến môi trường thực tế.
- Theo dõi hiệu suất: Sử dụng các công cụ như Prometheus và Grafana để giám sát độ chính xác, độ trễ và các lỗi phát sinh.
- Drift Detection: Áp dụng kỹ thuật phát hiện sự thay đổi dữ liệu (data drift) và mô hình (model drift) để tái huấn luyện khi cần thiết.

- MLFlow: Quản lý toàn bộ vòng đời từ theo dõi thí nghiệm, triển khai, đến giám sát mô hình.

7. Kết Luận Và Hướng Phát Triển Tương Lai

7.1. Kết luận

Dự án đã đạt được các mục tiêu đặt ra và mở ra nhiều tiềm năng cho việc ứng dụng trí tuệ nhân tạo trong quản lý rủi ro tài chính. Việc tiếp tục cải tiến và tối ưu hóa sẽ giúp hệ thống trở nên mạnh mẽ và hiệu quả hơn trong tương lai.

7.2. Hướng phát triển mô hình trong tương lai

Dự án dự đoán rủi ro tín dụng có tiềm năng phát triển mạnh mẽ trong tương lai thông qua việc áp dụng các mô hình tiên tiến như Deep Learning và Ensemble Learning, kết hợp với AutoML và tối ưu hóa tham số để nâng cao hiệu suất. Đồng thời, việc thu thập thêm dữ liệu phi cấu trúc và xử lý dữ liệu thời gian thực sẽ cải thiện chất lượng đầu vào. Hệ thống giám sát tự động và học online giúp duy trì hiệu suất mô hình trong môi trường sản xuất. Ngoài ra, Explainable AI (XAI) sẽ tăng cường tính minh bạch và tin cậy. Cuối cùng, mô hình có thể được mở rộng sang các lĩnh vực khác và tích hợp vào hệ thống quản lý rủi ro toàn diện nhằm hỗ trợ ra quyết định hiệu quả hơn.

7.2.1. Tích hợp dữ liệu thời gian thực

Tích hợp dữ liệu thời gian thực vào mô hình dự đoán rủi ro tín dụng giúp phát hiện sớm các dấu hiệu bất thường, hỗ trợ ra quyết định tức thời và cải thiện hiệu suất mô hình thông qua học liên tục. Dữ liệu cập nhật liên tục từ các giao dịch tài chính và thị trường cho phép mô hình phản ứng nhanh với những thay đổi. Mặc dù việc xử lý dữ liệu lớn đòi hỏi hạ tầng mạnh mẽ, các công nghệ như Apache Kafka và TensorRT có thể tối ưu hóa hiệu suất, giúp mô hình hoạt động hiệu quả trong môi trường kinh doanh năng động.

7.2.2. Sử dụng các mô hình nâng cao hơn

Sử dụng các mô hình nâng cao như Deep Learning, Ensemble Learning, AutoML, Explainable AI và Reinforcement Learning trong dự đoán rủi ro tín dụng có thể nâng cao độ chính xác và hiệu suất của hệ thống. Những mô hình này giúp nhận diện các mẫu dữ liệu phức tạp, tối ưu hóa quyết định tín dụng, và cải thiện tính minh bạch. Kết hợp với dữ liệu thời gian thực, các mô hình này sẽ giúp hệ thống phản ứng nhanh chóng và hiệu quả, từ đó hỗ

trợ ra quyết định tín dụng chính xác và đáng tin cậy hơn.

7.3. Ứng dụng trong các lĩnh vực khác

Các mô hình học máy tiên tiến như Deep Learning, Ensemble Learning, AutoML và Explainable AI có thể được ứng dụng rộng rãi trong nhiều lĩnh vực, bao gồm y tế (chẩn đoán bệnh, phân tích gen), tự động hóa và sản xuất (dự đoán bảo trì, tối ưu hóa chuỗi cung ứng), tài chính (dự đoán thị trường chứng khoán, phát hiện gian lận), marketing (phân tích hành vi khách hàng, dự đoán nhu cầu), giao thông (dự đoán tắc nghẽn, quản lý vận tải), và năng lượng (dự đoán tiêu thụ, quản lý lưới điện). Những ứng dụng này giúp nâng cao hiệu quả, tối ưu hóa quy trình và hỗ trợ ra quyết định chính xác hơn trong các ngành công nghiệp khác nhau.

8. Lời Cảm Ơn

Chúng em xin gửi lời cảm ơn chân thành đến giảng viên Nguyễn Thị Ngọc Diệp, người đã tận tình giảng dạy và hướng dẫn chúng em trong suốt quá trình học tập môn học. Những kiến thức và kinh nghiệm quý báu mà cô đã chia sẻ không chỉ giúp chúng em hiểu sâu hơn về học máy mà còn truyền cảm hứng để chúng em hoàn thành tốt dự án này.

Chúng em cũng xin cảm ơn nhà trường và viện Trí tuệ nhân tạo đã tạo điều kiện để chúng em có cơ hội thực hiện dự án. Những nguồn tài liệu và môi trường học tập lý tưởng đã hỗ trợ chúng em rất nhiều trong quá trình nghiên cứu.

Cuối cùng, lời chúc mừng đến các thành viên trong nhóm. Sự hợp tác, nỗ lực và tinh thần trách nhiệm của mọi người là yếu tố quan trọng giúp dự án được hoàn thành đúng thời hạn với chất lượng tốt nhất.

Xin chân thành cảm ơn!
Nguyễn Đình Khải
Vi Minh Hiển