

Nội dung

01.

Tính cấp thiết của
đề tài

04.

- Kết quả đánh giá
mô hình và thảo
luận

02.

Mô tả bài toán và
tập dữ liệu

05.

Kết luận



03.

Các kỹ thuật thực
hiện





Tính cấp thiết của đề tài



01



Gia tăng tình trạng gian lận thẻ tín dụng

- Số lượng giao dịch gian lận: Theo báo cáo của Nilson, thiệt hại từ gian lận thẻ tín dụng toàn cầu ước tính sẽ đạt 35 tỷ USD vào năm 2024.
- Tỉ lệ gia tăng: Tỉ lệ gian lận thẻ tín dụng đã tăng trung bình 20% mỗi năm trong thập kỷ qua.
- Mức độ phổ biến: Khoảng 46% người tiêu dùng Mỹ đã từng trải qua gian lận thẻ tín dụng ít nhất một lần.
- Việt Nam: tính đến tháng 7/2023, số lượng thẻ đang lưu hành đạt hơn 140 triệu thẻ (tăng 8,27% so với cuối năm 2021),



TÍNH CẤP THIẾT CỦA ĐỀ TÀI



- Tình trạng gian lận thẻ tín dụng đang gia tăng
- Thiệt hại kinh tế
- Mức độ phức tạp và tinh vi của các vụ gian lận
- Nhu cầu bảo vệ tài chính và tăng cường niềm tin của khách hàng
- Các mô hình học máy có khả năng phát hiện các mẫu giao dịch gian lận phức tạp và tinh vi



Phương Pháp Học Máy Trong Phát Hiện Gian Lận Thẻ Tín Dụng

Các kỹ thuật học máy phổ biến

- Hồi quy logistic (Logistic regression)
- Cây quyết định (Decision tree)
- Rừng ngẫu nhiên (Random Forest)
- Học tăng cường (Gradient boosting)
- Mạng nơ-ron sâu (Deep neural networks)

Lợi ích của học máy trong phát hiện gian lận

- Tự động hóa
- Độ chính xác cao
- Phát hiện các mẫu phức tạp
- Cải thiện theo thời gian



Thách Thức Của Học Máy Trong Phát Hiện Gian Lận Thẻ Tín Dụng

- Dữ liệu không cân bằng
- Bộ dữ liệu phức tạp với rất nhiều thuộc tính đầu vào
- Thời gian xử lý lâu do bộ dữ liệu lớn



Mô tả tập dữ liệu và bài toán



02



Giới thiệu về bộ dữ liệu

- Nguồn: Kaggle
- Tác giả: Andrea Dal Pozzolo, Olivier Caelen, Reid A. Johnson và Gianluca Bontempi.
- Tập dữ liệu này được thu thập và phân tích trong quá trình hợp tác nghiên cứu giữa Worldline và Nhóm Học Máy tại Đại học Libre de Bruxelles (ULB)



Credit Card Fraud Detection

Using the Machine Learning Classification Algorithms to detect Credit Card Fraudulent Activities



Giới thiệu về bộ dữ liệu

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column   Non-Null Count   Dtype  
 --- 
 0   Time     284807 non-null    float64
 1   V1       284807 non-null    float64
 2   V2       284807 non-null    float64
 3   V3       284807 non-null    float64
 4   V4       284807 non-null    float64
 5   V5       284807 non-null    float64
 6   V6       284807 non-null    float64
 7   V7       284807 non-null    float64
 8   V8       284807 non-null    float64
 9   V9       284807 non-null    float64
 10  V10      284807 non-null    float64
 11  V11      284807 non-null    float64
 12  V12      284807 non-null    float64
 13  V13      284807 non-null    float64
 14  V14      284807 non-null    float64
 15  V15      284807 non-null    float64
 16  V16      284807 non-null    float64
 17  V17      284807 non-null    float64
 18  V18      284807 non-null    float64
 19  V19      284807 non-null    float64
 20  V20      284807 non-null    float64
 21  V21      284807 non-null    float64
 22  V22      284807 non-null    float64
 23  V23      284807 non-null    float64
 24  V24      284807 non-null    float64
 25  V25      284807 non-null    float64
 26  V26      284807 non-null    float64
 27  V27      284807 non-null    float64
 28  V28      284807 non-null    float64
 29  Amount    284807 non-null    float64
 30  Class     284807 non-null    int64
dtypes: float64(30), int64(1)
memory usage: 67.4 MB
```

- Các giao dịch thẻ tín dụng của khách hàng Châu Âu trong 2 ngày của tháng 9 năm 2013.
- Dữ liệu có 31 features và 284807 bản ghi.
- Time là số giây kể từ giao dịch đầu tiên trong dataset.
- Amount là giá trị giao dịch (Euros).
- Class là thuộc tính đầu ra:
 - 1: gian lận.
 - 0: bình thường
- Các features V1, V2...V28 được chuyển đổi bởi PCA.

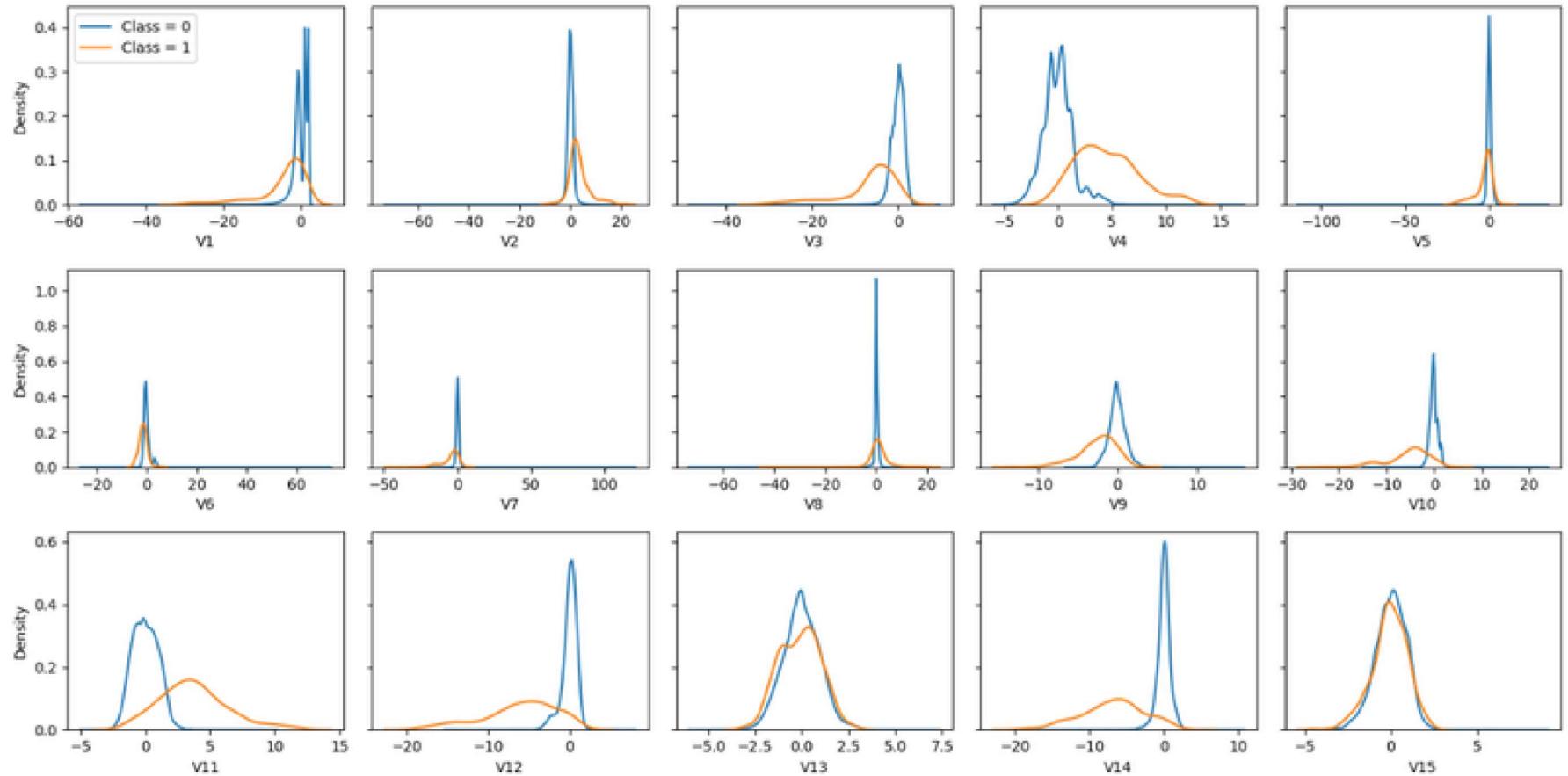


Sử dụng PCA để che dấu thông tin quan trọng

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column   Non-Null Count  Dtype  
--- 
 0   Time      284807 non-null   float64
 1   V1        284807 non-null   float64
 2   V2        284807 non-null   float64
 3   V3        284807 non-null   float64
 4   V4        284807 non-null   float64
 5   V5        284807 non-null   float64
 6   V6        284807 non-null   float64
 7   V7        284807 non-null   float64
 8   V8        284807 non-null   float64
 9   V9        284807 non-null   float64
 10  V10       284807 non-null   float64
 11  V11       284807 non-null   float64
 12  V12       284807 non-null   float64
 13  V13       284807 non-null   float64
 14  V14       284807 non-null   float64
 15  V15       284807 non-null   float64
 16  V16       284807 non-null   float64
 17  V17       284807 non-null   float64
 18  V18       284807 non-null   float64
 19  V19       284807 non-null   float64
 20  V20       284807 non-null   float64
 21  V21       284807 non-null   float64
 22  V22       284807 non-null   float64
 23  V23       284807 non-null   float64
 24  V24       284807 non-null   float64
 25  V25       284807 non-null   float64
 26  V26       284807 non-null   float64
 27  V27       284807 non-null   float64
 28  V28       284807 non-null   float64
 29  Amount    284807 non-null   float64
 30  Class     284807 non-null   int64
dtypes: float64(30), int64(1)
memory usage: 67.4 MB
```

- Những trường thông tin liên quan đến thông tin cá nhân của khách hàng có thể bị lợi dụng
- Một trong những phương pháp hiệu quả để che dấu những thông tin nhạy cảm là sử dụng **phép phân tích thành phần chính** (Principal Components Analysis - PCA).
- PCA phân tách các chiều không gian - các thành phần chính chứa đựng nhiều thông tin nhất của dữ liệu.
- Mỗi một giao dịch có một giá trị V1 và trường V1 sẽ lưu giữ một phần độ phân tán của dữ liệu - có ý nghĩa trong khi đưa vào các mô hình học máy.

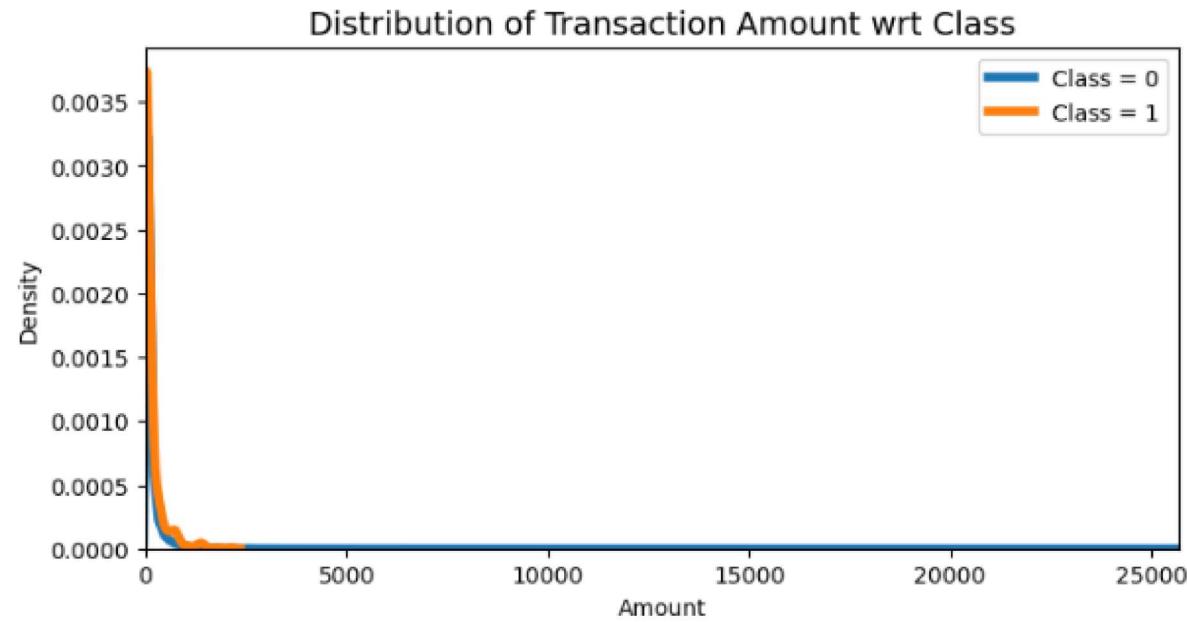
Phân bố trường dữ liệu



- Phân bố các trường V1-V15 với Target



Phân bố trường dữ liệu

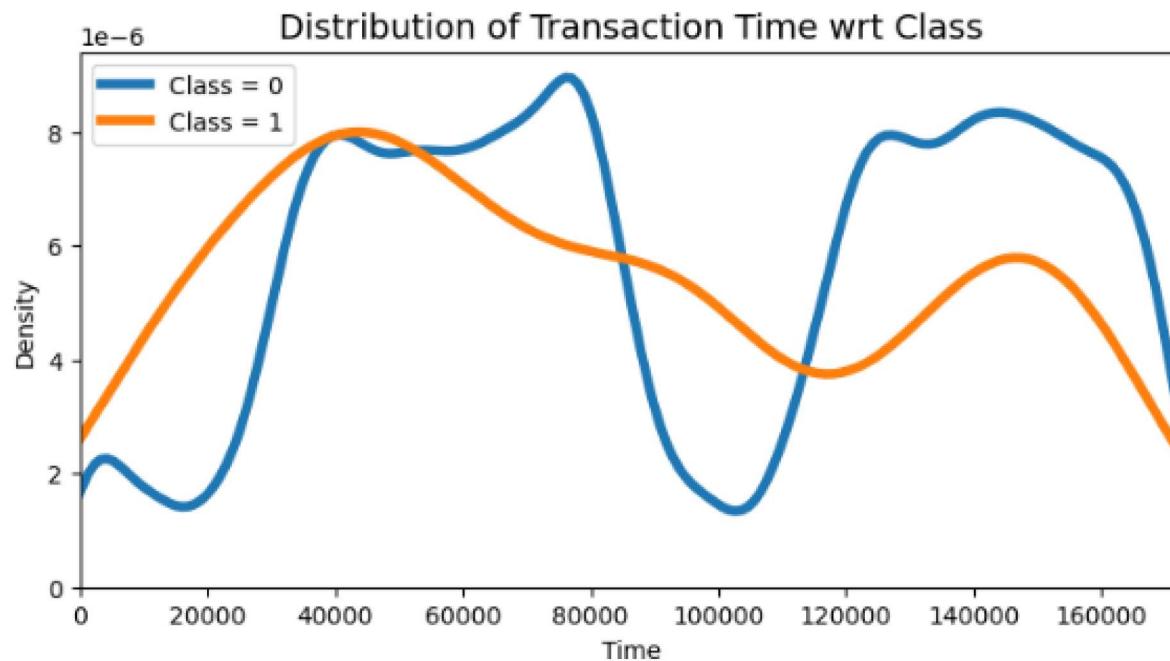


- Phân bố trường Amount - Target



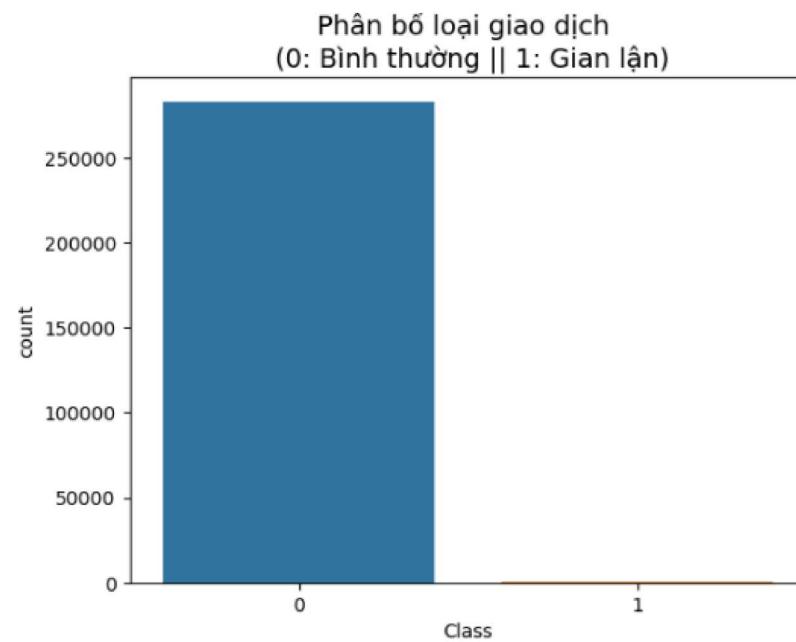
Phân bố dữ liệu

- Phân bố dữ liệu của thuộc tính Time theo Class





Phân bố dữ liệu theo Class



- Có tổng cộng $492/284.807 (0.172\%)$ giao dịch là gian lận.
- Có thể thấy bộ dữ liệu rất mất cân bằng. Một tỷ lệ rất nhỏ các giao dịch là gian lận.
- Gây khó khăn cho việc xây dựng mô hình học máy.

Các kỹ thuật thực hiện



03





Python



Kiểm tra trùng lặp

```
duplicated = df[df.duplicated()]
duplicated
```

114	74.0	1.038370	0.127486	0.184456	1.109950	0.441699	0.945283	-0.036715	0.350995	0.118950	...	0.102520	0.605089	0.023092	-0.6
115	74.0	1.038370	0.127486	0.184456	1.109950	0.441699	0.945283	-0.036715	0.350995	0.118950	...	0.102520	0.605089	0.023092	-0.6
...
282987	171288.0	1.912550	-0.455240	-1.750654	0.454324	2.089130	4.160019	-0.881302	1.081750	1.022928	...	-0.524067	-1.337510	0.473943	0.6
283483	171627.0	-1.464380	1.368119	0.815992	-0.601282	-0.689115	-0.487154	-0.303778	0.884953	0.054065	...	0.287217	0.947825	-0.218773	0.0
283485	171627.0	-1.457978	1.378203	0.811151	-0.603760	-0.711883	-0.471672	-0.282535	0.880654	0.052808	...	0.284205	0.949659	-0.216949	0.0
284191	172233.0	-2.667936	3.160505	-3.355984	1.007845	-0.377397	-0.109730	-0.667233	2.309700	-1.639306	...	0.391483	0.266536	-0.079853	-0.0
284193	172233.0	-2.691642	3.123168	-3.339407	1.017018	-0.293095	-0.167054	-0.745886	2.325616	-1.634651	...	0.402639	0.259746	-0.086606	-0.0

1081 rows × 31 columns

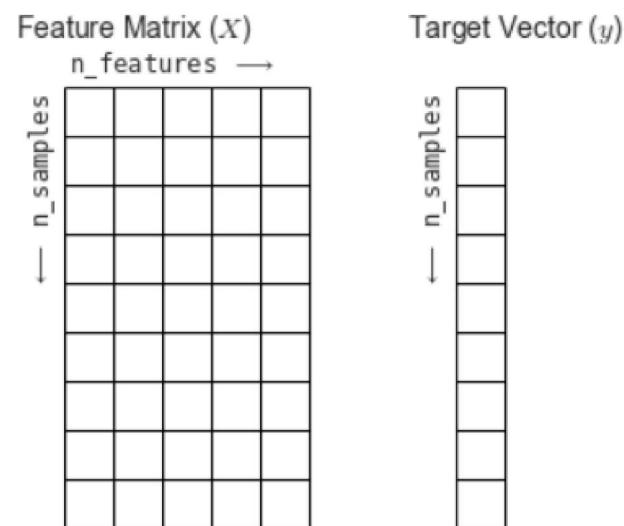
Có 1081 bản ghi trùng lặp

→ Loại bỏ trùng lặp

```
df.drop(duplicated.index, axis=0, inplace=True)
print('Kích thước sau khi bỏ hàng lặp: %d hàng, %d cột' %df.shape)
```

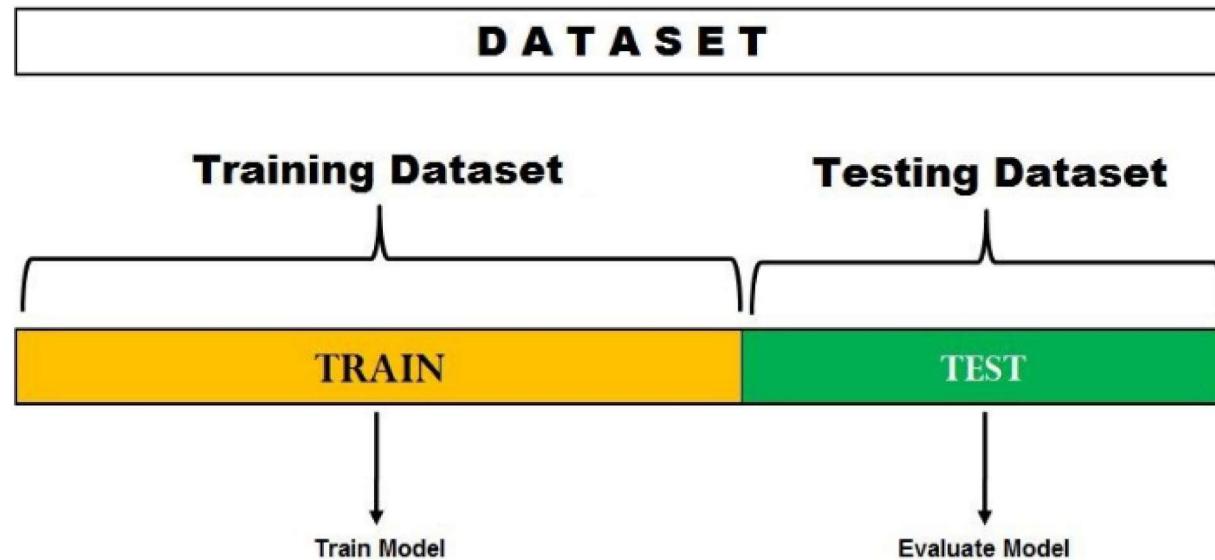
Kích thước sau khi bỏ hàng lặp: 283726 hàng, 31 cột

Trích xuất các biến phụ thuộc và biến độc lập



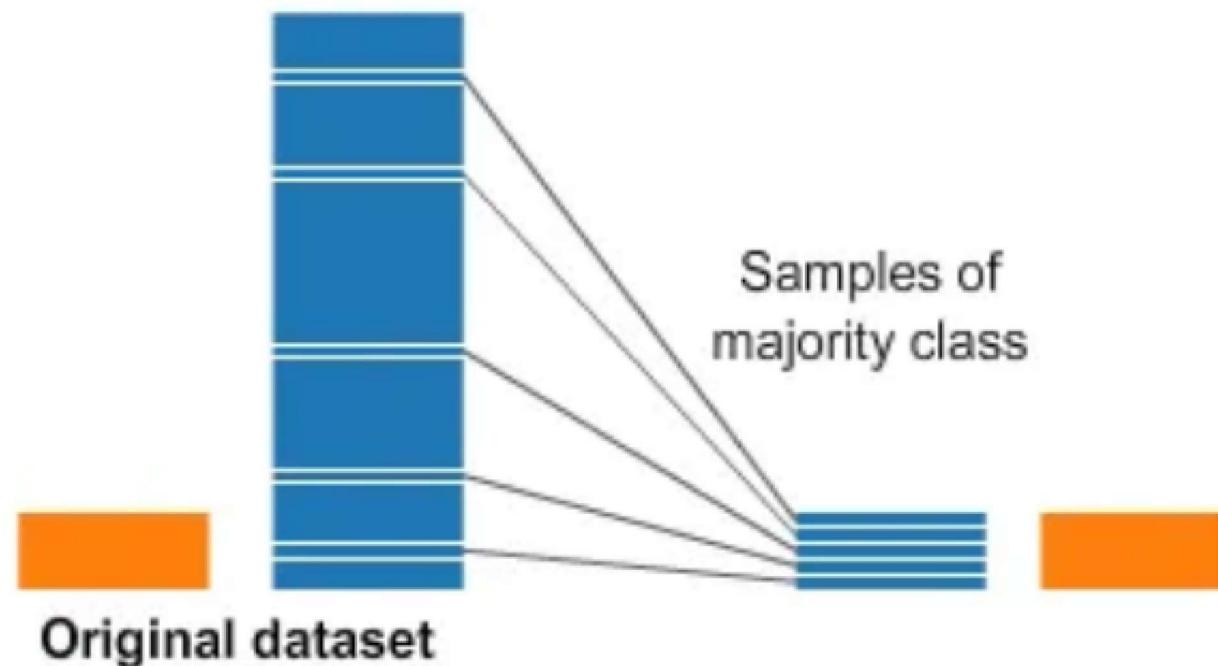
- Biến độc lập ---> các đặc trưng (feature - X)
- Biến phụ thuộc ---> Label/Target - y

Tách tập dữ liệu Train - Test

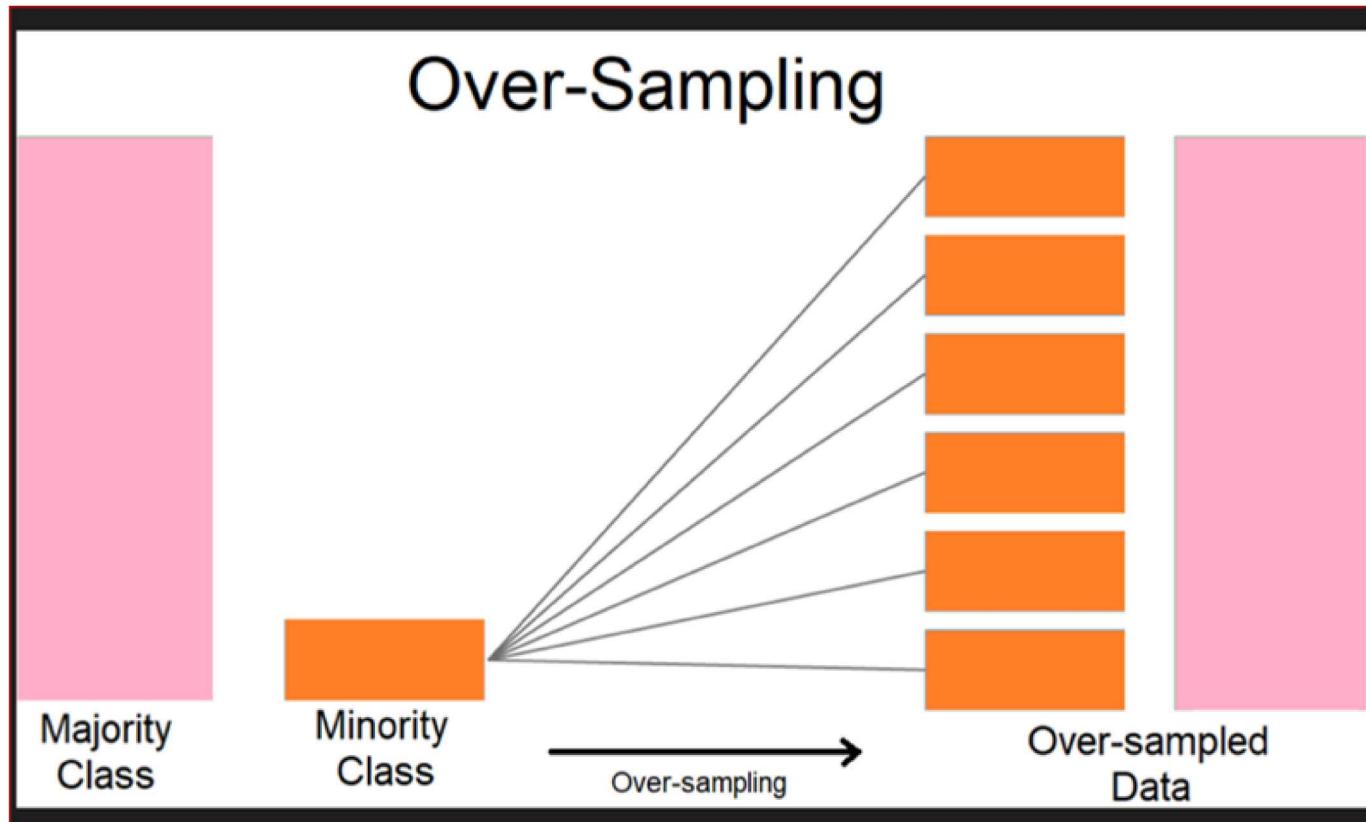


Các kỹ thuật xử lý mất cân bằng

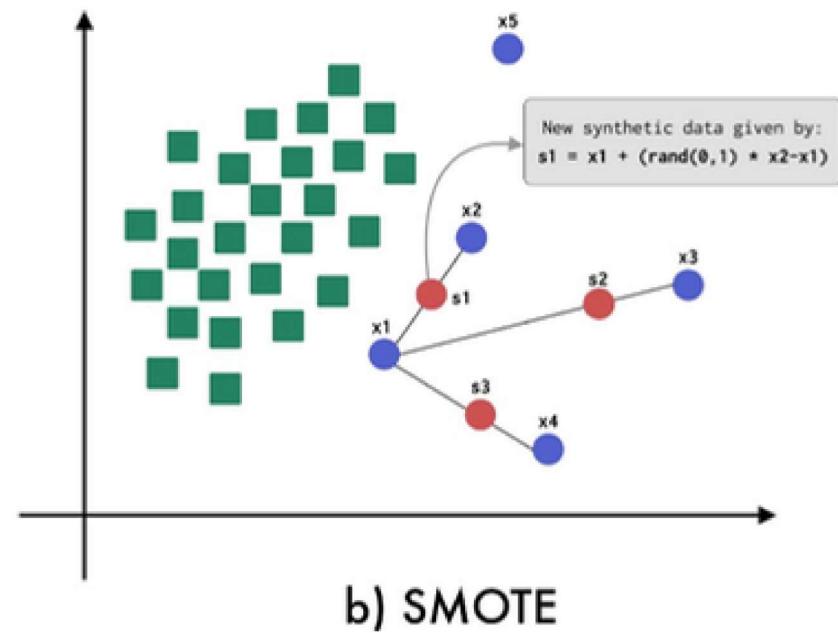
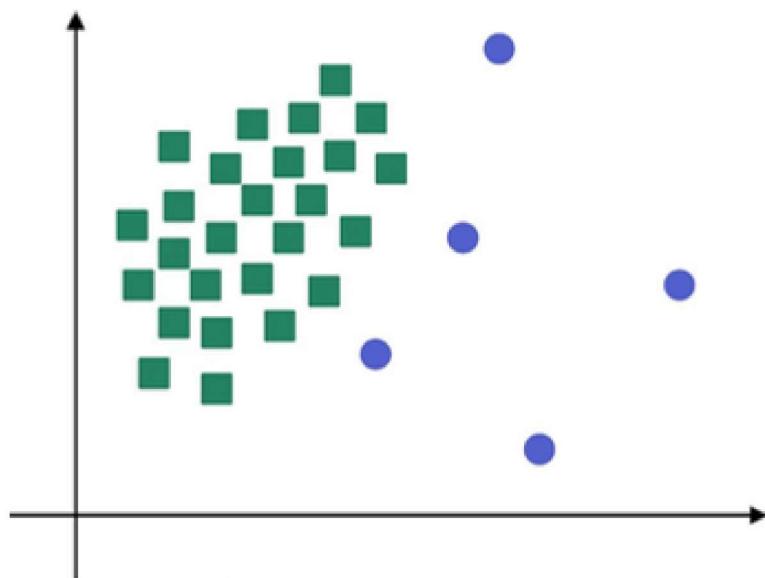
Undersampling



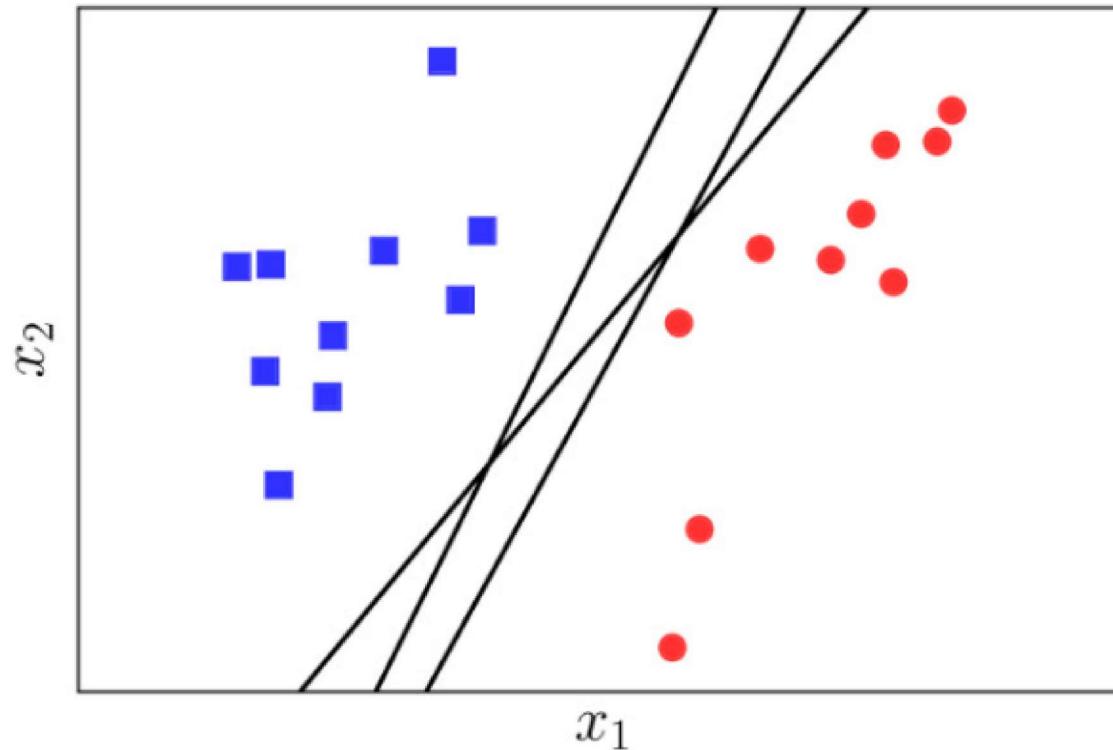
Các kỹ thuật xử lý mất cân bằng



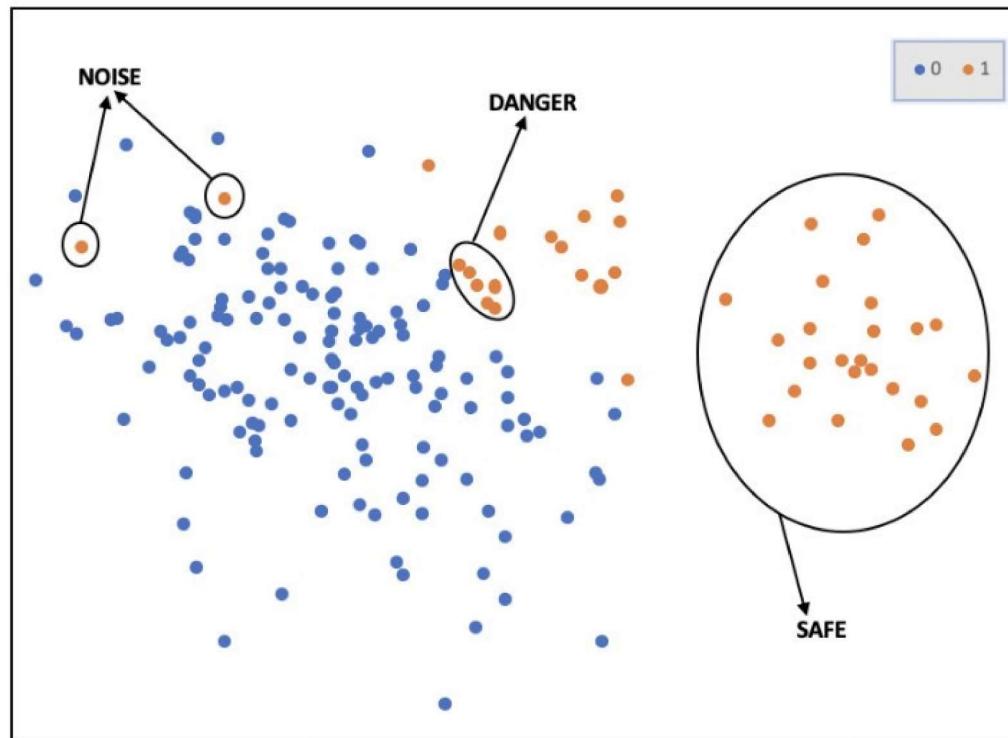
Tham số tối ưu và Smote



Support Vector Machine (SVM)



SVMSMOTE (Synthetic Minority Over-sampling Technique using Support Vector Machines)



Kết quả sử dụng SVMMOTE

Dữ liệu gốc

Class

0 283253
1 473

Name: count, dtype: int64

Dữ liệu tổng hợp

Class

0 284315
1 1421

Name: count, dtype: int64

tỷ lệ lấy mẫu: 0.5%



Mô hình

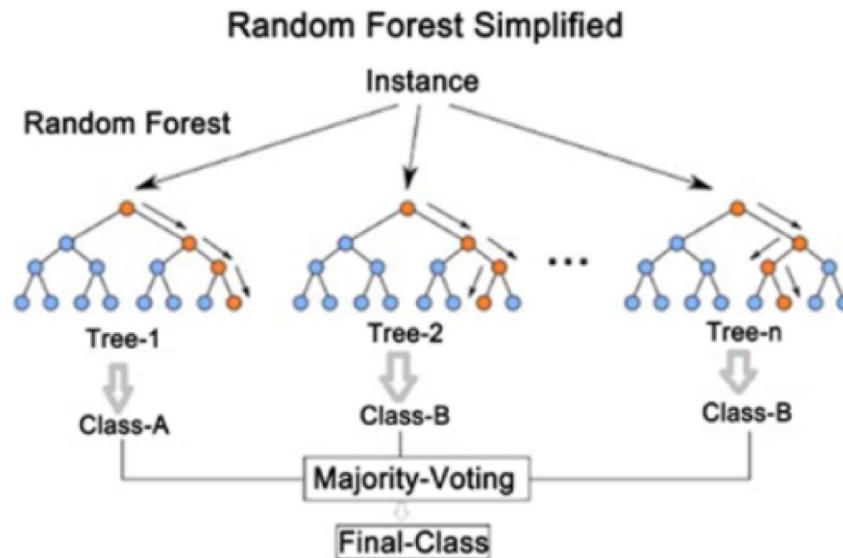
Logistic Regression

$$P(Y = 1) = \frac{e^{\beta_0 + \beta_1 x_1 + \dots + \beta_n x_n}}{1 + e^{\beta_0 + \beta_1 x_1 + \dots + \beta_n x_n}}$$



Mô hình

Random Forest

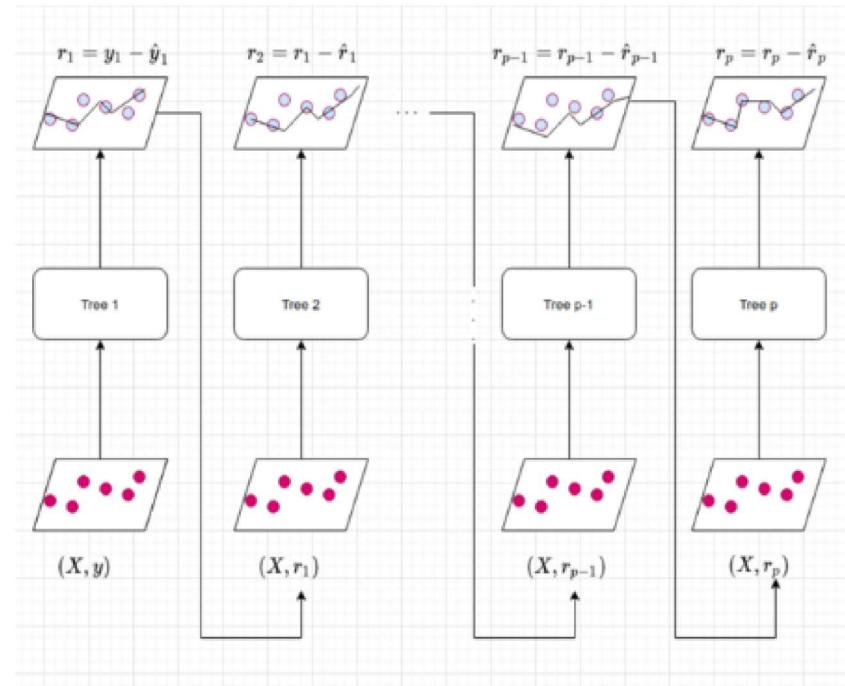


Nguồn: Niculescu & Lam (2019)



Mô hình

Gradient Boosting





Mô hình

Artificial Neural Network

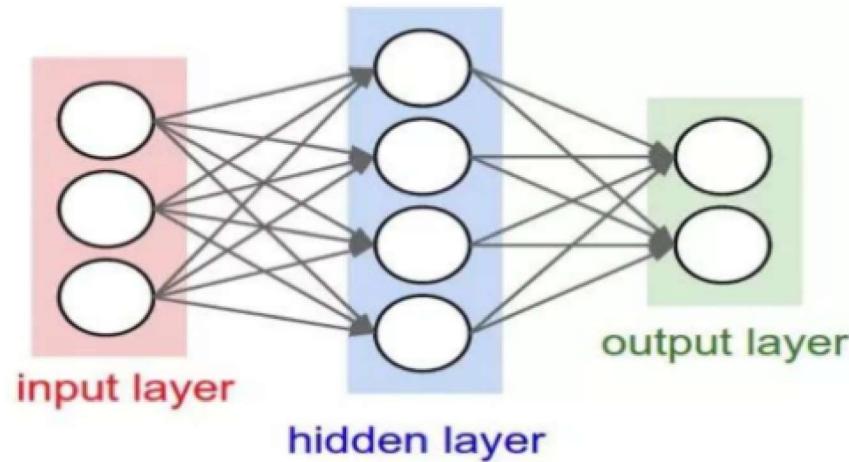
- Mạng nơ-ron nhân tạo là một mô hình toán học lấy cảm hứng từ cấu trúc và chức năng của nơ-ron trong não bộ con người để thực hiện các nhiệm vụ như phân loại và dự đoán
- Giải quyết các vấn đề phi tuyến tính
- Tự động hóa và tăng cường các hệ thống thông minh
- Xử lý và phân tích dữ liệu lớn
- Hỗ trợ ra quyết định
- Nâng cao khả năng phát hiện gian lận



Mô hình

Kiến trúc mạng nơ-ron nhân tạo

Tầng Perceptron





Confusion Matrix

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN



Đánh giá mô hình

$$\text{Accuracy} = \frac{TP + TN}{\text{total sample}}$$

Độ chính xác giúp ta đánh giá hiệu quả dự báo của mô hình trên một bộ dữ liệu.



Đánh giá mô hình

$$\text{Precision} = \frac{\textcolor{brown}{TP}}{\text{total predicted positive}} = \frac{\textcolor{brown}{TP}}{\textcolor{brown}{TP} + \textcolor{brown}{FP}}$$

Precision sẽ cho chúng ta biết mức độ chuẩn xác của mô hình đối với các mẫu dương.



Đánh giá mô hình

$$\text{Recall} = \frac{TP}{\text{total actual positive}} = \frac{TP}{TP + FN}$$

Recall đo lường tỷ lệ dự báo chính xác các trường hợp positive trên toàn bộ các mẫu thuộc nhóm positive



Đánh giá mô hình

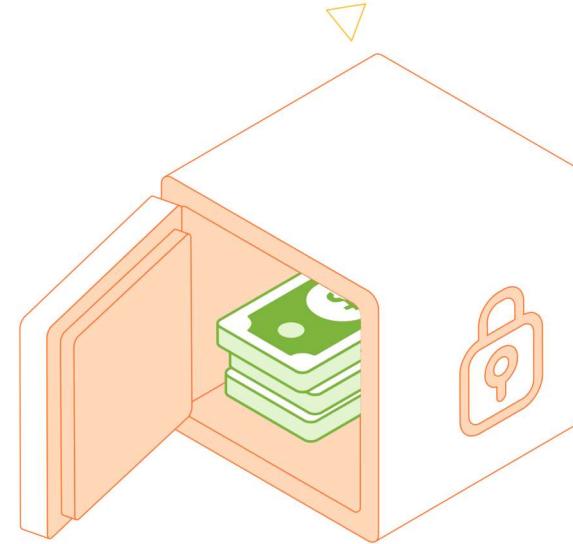
$$F_1 = \frac{2}{\text{precision}^{-1} + \text{recall}^{-1}}$$

F1 Score là trung bình điều hòa giữa precision và recall.



Kết quả đánh giá

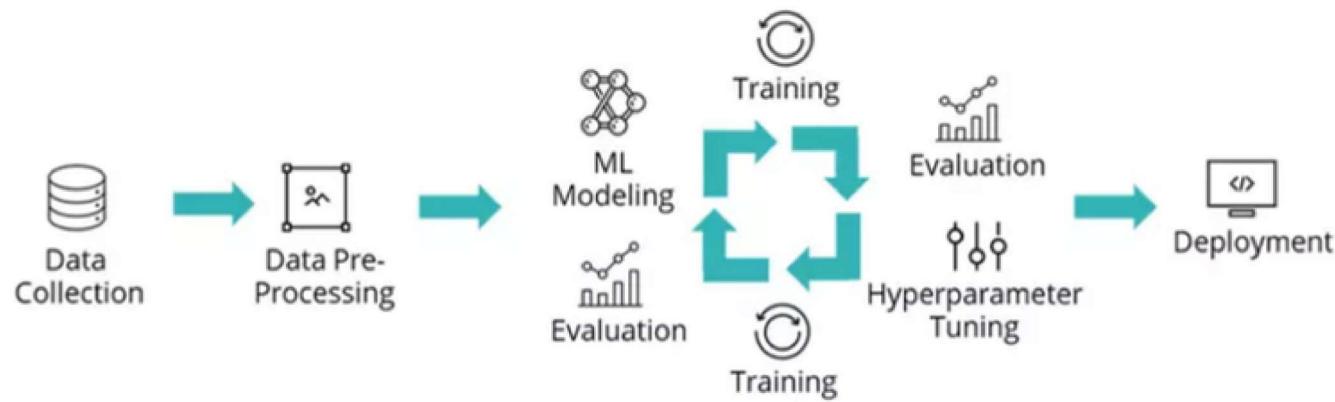
Sử dụng bộ dữ liệu vừa
tạo để xây dựng mô hình



04



Model centric





Bảng 4. Model - centric



Mô hình	ANN	Random Forest	Logistic Regression	Gradient Boosting
Chi số				
Accuracy	0.80	0.80	0.81	0.81
Precision	0.64	0.67	0.67	0.69
Recall	0.53	0.53	0.57	0.53
F1 Score	0.58	0.59	0.61	0.60





Bảng 3. Data - centric

Mô hình	ANN	Random Forest	Logistic Regression	Gradient Boosting
Chi số				
Accuracy	0.78	0.78	0.77	0.84
Precision	0.76	0.74	0.73	0.81
Recall	0.81	0.85	0.82	0.87
F1 Score	0.78	0.79	0.78	0.84

Data-centric vs Model-centric

	Steel defect detection	Solar panel	Surface inspection
Baseline	76.2%	75.68%	85.05%
Model-centric	+0% (76.2%)	+0.04% (75.72%)	+0.00% (85.05%)
Data-centric	+16.9% (93.1%)	+3.06% (78.74%)	+0.4% (85.45%)

Model-centric Giữ nguyên dữ liệu, phát triển model để tăng độ chính xác trên dữ liệu

Data-centric Sử dụng mô hình cố định và sử dụng các công cụ để **tăng cường chất lượng dữ liệu**

Sử dụng mô hình

```
# New sample data
new_sample = np.array([[1.0, -0.358354, 0.340163, 0.773209, 0.37978, -0.503198, 2.800499, 0.791461,
                      2.247676, -1.514654, 1.207643, -1.624501, 1.066084, -0.717293, -0.165946, 1.345865,
                      -2.890083, 1.109969, -0.121359, -2.261857, 2.52498, -0.247998, 2.771679, 1.909412, -1.689281,
                      -1.327642, -0.139097, -1.055353, -0.059752, 687.66]
])

# Make prediction
prediction = gb.predict(new_sample)
predicted_label = 'giao dịch bình thường' if prediction[0] == 0 else 'giao dịch gian lận'

print('Predicted class:', prediction[0], '-', predicted_label)

Predicted class: 0 - giao dịch bình thường
```



Kết luận



05



Kết quả đạt được

- Sử dụng kỹ thuật SVMSMOTE để tăng cường dữ liệu lớp thiểu số
- Mô hình Gradient Boosting có chỉ số f1 score cao nhất.



Hướng phát triển của đề tài

- Tập trung xử lý giữ liệu bằng các phương pháp sinh dữ liệu khác như GAN (Generative Adversarial Networks).
- Sử dụng các thuật toán tối ưu để tìm bộ tham số tối ưu cho từng mô hình.