BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT HƯNG YÊN

BÀI TẬP LỚN

TÌM HIỂU MÔ HÌNH LOGISTIC REGRESSION, DECISION TREE, RANDOM FOREST, LIGHTGBM ĐỂ DỰ ĐOÁN RỦI RO TÍN DỤNG

NGÀNH: KHOA HỌC MÁY TÍNH

CHUYÊN NGÀNH: TRÍ TUỆ NHÂN TẠO VÀ KHOA HỌC DỮ LIỆU

SINH VIÊN: PHẠM THỊ LINH CHI

MÃ SINH VIÊN: 12423005

MÃ LỚP: 124231

GIẢNG VIÊN HƯỚNG DẪN: PGS. TS. NGUYỄN VĂN HẬU

HƯNG YÊN – 2025

NHẬN XÉT

Nhận xét của giảng viên hướng dẫn:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

GIẢNG VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên)

LỜI CAM ĐOAN

Em xin cam đoan bài tập lớn "Tìm hiểu mô hình logistic regression, decision tree, random forest, lightgbm để dự đoán rủi ro tín dụng" là kết quả thực hiện của bản thân em dưới sự hướng dẫn của Thầy Nguyễn Văn Hậu và thầy Nguyễn Tuấn Anh.

Những phần sử dụng tài liệu tham khảo trong bài tập lớn đã được nêu rõ trong phần tài liệu tham khảo. Các kết quả trình bày trong bài tập lớn và chương trình xây dựng được hoàn toàn là kết quả do bản thân em thực hiện.

Nếu vi phạm lời cam đoan này, em xin chịu hoàn toàn trách nhiệm trước khoa và nhà trường.

Hưng Yên, ngày 30 tháng 12 năm 2025

SINH VIÊN

(Ký, ghi rõ họ tên)

LỜI CẢM ƠN

Để có thể hoàn thành bài tập lớn này, lời đầu tiên em xin phép gửi lời cảm ơn tới bộ môn Khoa học máy tính, Khoa Công nghệ thông tin – Trường Đại học Sư phạm Kỹ thuật Hưng Yên đã tạo điều kiện thuận lợi cho em thực hiện bài tập lớn môn học này.

Đặc biệt em xin chân thành cảm ơn Thầy Nguyễn Văn Hậu đã rất tận tình hướng dẫn, chỉ bảo em trong suốt thời gian thực hiện bài tập lớn vừa qua.

Em cũng xin chân thành cảm ơn tất cả các thầy/cô trong trường đã tận tình giảng dạy, trang bị cho em những kiến thức cần thiết, quý báu để giúp em thực hiện được bài tập lớn này.

Mặc dù em đã có cố gắng, nhưng với trình độ còn hạn chế, trong quá trình thực hiện đề tài không tránh khỏi những thiếu sót. Em hi vọng sẽ nhận được những ý kiến nhận xét, góp ý của các thầy/cô về những kết quả triển khai trong bài tập lớn.

Em xin trân trọng cảm ơn!

MỤC LỤC

DANH MỤC CÁC THUẬT NGỮ

STT	Từ viết tắt	Cụm từ tiếng Anh	Diễn giải
1	ML	Machine Learning	Học máy, lĩnh vực nghiên cứu các thuật toán cho phép máy tính học từ dữ liệu mà không cần lập trình rõ ràng từng bước.
2	LR	Logistic Regression	Mô hình hồi quy logistic, dùng cho phân loại nhị phân, dự đoán xác suất một sự kiện xảy ra.
3	DT	Decision Tree	Cây quyết định, mô hình phân loại phi tuyến dựa trên các quyết định nhánh theo giá trị biến.
4	RF	Random Forest	Rừng ngẫu nhiên, tập hợp nhiều cây quyết định để cải thiện độ chính xác và giảm overfitting.
5	LGBM	Light Gradient Boosting Machine	Mô hình boosting dựa trên cây quyết định, huấn luyện nhanh, hiệu quả trên dữ liệu lớn, hỗ trợ xử lý mất cân bằng lớp.
10	AUC	Area Under Curve	Diện tích dưới đường cong ROC, đánh giá khả năng phân biệt giữa các lớp.
11	ROC	Receiver Operating Characteristic	Đường cong hiển thị trade-off giữa Recall (True Positive Rate) và False Positive Rate.
12	SMOTE	Synthetic Minority Oversampling Technique	Kỹ thuật tạo dữ liệu tổng hợp cho lớp thiểu số nhằm cân bằng dữ liệu.

DANH MỤC CÁC HÌNH VẼ, ĐỒ THỊ

CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

1.1. Lý do chọn đề tài

Trong hoạt động của các tổ chức tài chính và ngân hàng, rủi ro tín dụng luôn là một trong những rủi ro lớn nhất, ảnh hưởng trực tiếp đến lợi nhuận, tính ổn định và khả năng phát triển bền vững của tổ chức. Việc đánh giá chính xác khả năng khách hàng không thực hiện đúng nghĩa vụ trả nợ đóng vai trò quan trọng trong quá trình ra quyết định cấp tín dụng.

Trong bối cảnh dữ liệu ngày càng phong phú và quy mô lớn, các phương pháp đánh giá tín dụng truyền thống dựa nhiều vào kinh nghiệm chuyên gia hoặc các quy tắc thủ công dần bộc lộ hạn chế về độ chính xác, khả năng mở rộng và tính nhất quán. Machine Learning đã và đang trở thành công cụ hiệu quả giúp khai thác dữ liệu lịch sử để tự động hóa và nâng cao chất lượng dự đoán rủi ro tín dụng.

Xuất phát từ thực tiễn đó, đề tài "Dự đoán rủi ro tín dụng sử dụng các mô hình Machine Learning" được lựa chọn nhằm nghiên cứu, so sánh các mô hình học máy phổ biến như Logistic Regression, Decision Tree, Random Forest và LightGBM, từ đó xây dựng một hệ thống dự đoán rủi ro tín dụng có tính ứng dụng thực tế.

1.2. Mục tiêu của đề tài

1.2.1 Mục tiêu tổng quát

Xây dựng và đánh giá hệ thống dự đoán rủi ro tín dụng của khách hàng cá nhân dựa trên dữ liệu lịch sử, thông qua việc áp dụng và so sánh các mô hình Machine Learning, nhằm hỗ trợ quá trình ra quyết định tín dụng.

1.2.2 Mục tiêu cụ thể

- Nghiên cứu cơ sở lý thuyết về rủi ro tín dụng và các mô hình Machine Learning áp dụng trong bài toán phân loại nhị phân.
- Tiền xử lý và phân tích dữ liệu tín dụng, bao gồm xử lý dữ liệu thiếu, dữ liệu mất cân bằng và biến phân loại.

- Xây dựng và huấn luyện các mô hình: Logistic Regression, Decision Tree, Random Forest và LightGBM.
- So sánh hiệu quả các mô hình dựa trên các chỉ số đánh giá như Accuracy, Precision, Recall, F1-score và AUC.
- Lựa chọn mô hình phù hợp và triển khai thử nghiệm dưới dạng ứng dụng Streamlit để minh họa khả năng ứng dụng thực tế.

1.3. Giới hạn và phạm vi của đề tài

1.3.1 Đối tượng nghiên cứu

Đối tượng nghiên cứu của đề tài là bài toán dự đoán rủi ro tín dụng (Credit Risk Prediction) đối với khách hàng cá nhân, trong đó đầu ra là khả năng khách hàng thuộc nhóm rủi ro cao (vỡ nợ) hoặc rủi ro thấp.

1.3.2 Phạm vi nghiên cứu

- Dữ liệu sử dụng là bộ dữ liệu tín dụng lịch sử, bao gồm các thông tin cơ bản về khách hàng và khoản vay.
- Đề tài tập trung vào các mô hình Machine Learning truyền thống và nâng cao, không đi sâu vào Deep Learning.
- Kết quả nghiên cứu mang tính học thuật và mô phỏng, chưa tích hợp các yếu tố pháp lý, đạo đức hay hệ thống nghiệp vụ thực tế của ngân hàng.

1.4. Nội dung thực hiện

Nội dung chính của đề tài bao gồm các bước sau:

- Khảo sát và nghiên cứu tổng quan về bài toán rủi ro tín dụng.
- Thu thập, khám phá và tiền xử lý dữ liệu.
- Xây dựng và huấn luyện các mô hình Machine Learning.
- Đánh giá, so sánh và phân tích kết quả của các mô hình.
- Lựa chọn mô hình tối ưu và triển khai thử nghiệm ứng dụng dự đoán.
- Tổng kết kết quả đạt được, đánh giá hạn chế và đề xuất hướng phát triển.

1.5. Phương pháp tiếp cận

Đề tài được thực hiện theo phương pháp nghiên cứu thực nghiệm (experimental research), kết hợp giữa lý thuyết và thực hành. Cụ thể:

- Áp dụng các phương pháp phân tích dữ liệu và học máy để xây dựng mô hình dự đoán.
- Thực hiện so sánh mô hình dựa trên các chỉ số đánh giá định lượng.
- Sử dụng quy trình Machine Learning chuẩn gồm các bước: tiền xử lý dữ liệu – huấn luyện mô hình – đánh giá – triển khai.
- Áp dụng tư duy hướng ứng dụng nhằm đảm bảo kết quả nghiên cứu có khả năng sử dụng trong thực tế.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Tổng quan bài toán dự đoán rủi ro tín dụng

Rủi ro tín dụng là khả năng khách hàng không thực hiện đầy đủ nghĩa vụ trả nợ theo cam kết, bao gồm việc trả chậm, trả không đầy đủ hoặc vỡ nợ hoàn toàn. Đây là loại rủi ro chiếm tỷ trọng lớn trong tổng rủi ro của ngân hàng thương mại, vì dư nợ tín dụng chiếm phần lớn tài sản của ngân hàng và tổn thất tín dụng thường có giá trị lớn, khó thu hồi.

Bài toán dự đoán rủi ro tín dụng nhằm đánh giá khả năng vỡ nợ của khách hàng trước khi cấp tín dụng, dựa trên thông tin lịch sử, hồ sơ cá nhân và đặc điểm khoản vay. Về phương diện học máy, đây là bài toán phân loại nhị phân, với đầu vào là vector đặc trưng khách hàng x = (x1, x2,..., xd) và đầu ra là nhãn y ∈ {0,1}, trong đó y = 1 biểu thị khách hàng rủi ro cao, y = 0 biểu thị khách hàng rủi ro thấp. Mục tiêu mô hình là ước lượng xác suất khách hàng thuộc lớp rủi ro cao:

$$\hat{p} = P(y = 1 \mid \mathbf{x})$$

Dữ liệu rủi ro tín dụng thường đặc trưng bởi sự mất cân bằng lớp, tính đa dạng của biến (số, phân loại, có thứ tự), yêu cầu giải thích cao, và chi phí sai lầm không đối xứng (false negative gây tổn thất lớn, false positive làm mất cơ hội kinh doanh). Do đó, mô hình không chỉ cần độ chính xác cao mà còn phải nhận diện đúng khách hàng rủi ro.

2.2. Tiền xử lý dữ liệu

Tiền xử lý dữ liệu là bước quan trọng nhằm nâng cao chất lượng dữ liệu, giúp mô hình học hiệu quả và giảm nhiễu. Các bước chính bao gồm xử lý dữ liệu thiếu, mã hóa biến phân loại, chuẩn hóa biến số và xử lý mất cân bằng lớp.

2.2.1. Xử lý dữ liệu thiếu

Dữ liệu thiếu có thể phát sinh từ khách hàng không cung cấp thông tin, lỗi thu thập dữ liệu hoặc lịch sử chưa đầy đủ. Nếu không xử lý, dữ liệu thiếu có thể làm sai kết quả huấn luyện hoặc gây lỗi cho thuật toán.

Trong đề tài này, các biến số thiếu được điền bằng giá trị trung vị (median) của biến:

$$x_i = \begin{cases} x_i, & \text{nếu không thiếu} \\ \text{median}(X), & \text{nếu thiếu} \end{cases}$$

Trung vị được chọn vì ít bị ảnh hưởng bởi outliers và phù hợp với dữ liệu tài chính thường phân phối lệch.

2.2.2. Mã hóa biến phân loại

a) Mã hóa nhị phân (Binary Encoding)

Áp dụng cho các biến có hai giá trị (Yes/No, Y/N):

$$x = \begin{cases} 1 & \text{Yes} \\ 0 & \text{No} \end{cases}$$

Ưu điểm:

- Giữ nguyên ý nghĩa logic
- Phù hợp với mọi mô hình

b) Mã hóa có thứ tự (Ordinal Encoding)

Áp dụng cho biến có thứ tự tự nhiên, ví dụ xếp hạng tín dụng A–G:

$$A \rightarrow 0,\ B \rightarrow 1,\ C \rightarrow 2,\ \ldots,\ G \rightarrow 6$$

Việc giữ thứ tự giúp mô hình:

- Hiểu được mức độ rủi ro tăng dần
- Học được quan hệ tuyến tính hoặc phi tuyến

c) One-Hot Encoding

Áp dụng cho biến phân loại không có thứ tự.

Với biến có k giá trị, One-Hot Encoding tạo ra kkk biến nhị phân.

Ưu điểm:

- Không tạo giả định thứ tự

- Phù hợp với Logistic Regression và Tree-based models

Việc giữ thứ tự giúp mô hình:

- Hiểu được mức độ rủi ro tăng dần
- Học được quan hệ tuyến tính hoặc phi tuyến

2.2.3. Chuẩn hóa dữ liệu số

a) Mục đích chuẩn hóa

Chuẩn hóa dữ liệu là bước quan trọng nhằm đưa các biến số về cùng thang đo, tránh trường hợp biến có giá trị lớn chi phối mô hình và giúp thuật toán tối ưu hội tụ nhanh hơn. Đối với các mô hình dựa trên gradient như Logistic Regression hay LightGBM, việc các biến đầu vào nằm trong cùng phạm vi giúp gradient descent ổn định, giảm số bước cần thiết để đạt cực tiểu hàm mất mát, đồng thời cải thiện hiệu quả học của mô hình.

b) Standardization (Z-score Normalization)

Một phương pháp phổ biến là chuẩn hóa Z-score, trong đó mỗi giá trị của biến được trừ đi trung bình và chia cho độ lệch chuẩn của biến:

$$x' = \frac{x - \mu}{\sigma}$$

Trong đó:

- μ: trung bình
- σ: độ lệch chuẩn

Phương pháp này phù hợp với dữ liệu gần phân phối chuẩn và giúp giữ nguyên các đặc trưng phân phối ban đầu của biến. Trong đề tài, chuẩn hóa theo Z-score được sử dụng chủ yếu cho các biến số liên tục, vì nó hỗ trợ tốt cho Logistic Regression và giúp các biến số có tác động cân bằng trong quá trình tối ưu hàm mất mát.

c) Min-Max Normalization

Một phương pháp khác là chuẩn hóa Min-Max, trong đó giá trị mỗi biến được đưa về khoảng [0, 1]:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

Min-Max Normalization đặc biệt hữu ích khi cần giữ tỷ lệ giữa các giá trị và khi mô hình yêu cầu dữ liệu trong một phạm vi xác định. Tuy nhiên, phương pháp này khá nhạy cảm với outliers: các giá trị ngoại lai có thể làm co cụm phần lớn dữ liệu vào một phạm vi hẹp, ảnh hưởng đến hiệu quả học của mô hình. Trong đề tài, phương pháp này có thể áp dụng cho các biến số mà outliers đã được xử lý hoặc biến có phân phối gần đều.

2.2.4. Xử lý mất cân bằng lớp

Với dữ liệu tín dụng, số khách hàng vỡ nợ thường ít hơn nhiều so với khách hàng tốt. Nếu không xử lý, mô hình có thể đạt accuracy cao nhưng bỏ sót nhiều khách hàng rủi ro. Hai phương pháp phổ biến:

- Class Weight: Gán trọng số wyi cho từng nhãn trong hàm mất mát:

$$Loss = \sum_{i=1}^{N} w_{y_i} \cdot \ell(y_i, \hat{y}_i)$$

Nhãn ít mẫu có trọng số cao, ảnh hưởng lớn hơn khi tính loss, giúp mô hình học chú ý đến lớp thiểu số.

- Scale_pos_weight trong LightGBM:

$$\text{scale\_pos\_weight} = \frac{N_{negative}}{N_{positive}}$$

Hệ số này nhân trực tiếp vào gradient và hessian của lớp dương, làm tăng ảnh hưởng của lớp rủi ro trong quá trình split cây và tối ưu loss.

2.3. Các mô hình học máy áp dụng

2.3.1. Logistic Regression

Công thức:

$$P(y = 1 \mid \mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x} + b) = \frac{1}{1 + e^{-(\mathbf{w}^\top \mathbf{x} + b)}}$$

Trong đó:

- x=(x1,x2,...,xd) là vector đặc trưng khách hàng
- w là vector trọng số, bbb là bias
- σ là hàm sigmoid

Đặc điểm:

- Mô hình tuyến tính, dự đoán xác suất rủi ro trực tiếp.
- Thích hợp cho bài toán phân loại nhị phân.
- Giải thích hệ số dễ dàng: mỗi biến ảnh hưởng đến xác suất rủi ro có thể được diễn giải trực tiếp.
- Tính toán nhanh, phù hợp dataset trung bình.

Lý do chọn:

- Phù hợp làm benchmark cơ bản.
- Hiểu rõ mối quan hệ giữa từng đặc trưng và rủi ro tín dụng.

Ưu nhược điểm:

- Ưu điểm: Đơn giản, dễ giải thích, tính toán nhanh.
- Nhược điểm: Không mạnh khi dữ liệu phi tuyến hoặc biến số phức tạp.

2.3.2. Decision Tree

Công thức:

Cây quyết định dựa trên việc chọn split tối ưu để giảm hàm mất mát. Ví dụ sử dụng Entropy / Gini Index:

- Entropy:

$$H(S) = -\sum_{i=1}^{c} p_i \log_2 p_i$$

- Gain thông tin khi chia tập S theo feature A:

$$\text{Information Gain}(S, A) = H(S) - \sum_{v \in \text{values}(A)} \frac{|S_v|}{|S|} H(S_v)$$

Đặc điểm:

- Mô hình phi tuyến, trực quan.
- Dễ xác định các ngưỡng quyết định quan trọng.

Lý do chọn:

- Trực quan, dễ giải thích, phù hợp trình bày cho người không chuyên.
- Phát hiện tương tác phi tuyến giữa các biến.

Ưu nhược điểm:

- Ưu điểm: Trực quan, dễ giải thích, nhận biết các yếu tố quyết định rủi ro.
- Nhược điểm: Dễ overfitting với dữ liệu nhỏ hoặc nhiều biến.

2.3.3. Random Forest

Công thức tổng quát:

$$\hat{y} = \text{majority\_vote}(T_1(\mathbf{x}), T_2(\mathbf{x}), ..., T_n(\mathbf{x}))$$

Trong đó Ti(x) là dự đoán của cây thứ i.

Đặc điểm:

- Tập hợp nhiều cây quyết định (ensemble) để giảm variance.
- Kỹ thuật Bagging: bootstrap các tập con dữ liệu để huấn luyện từng cây.

Lý do chọn:

- Giảm overfitting so với cây đơn.
- Hoạt động tốt với dữ liệu tabular nhiều biến và nhiễu.

Ưu nhược điểm:

- Ưu điểm: Độ chính xác cao, robust với dữ liệu lớn và phức tạp.
- Nhược điểm: Ít trực quan, tốn tài nguyên hơn LR hoặc DT.

2.3.4. LightGBM

Công thức Gradient Boosting cơ bản:

$$\hat{y}^{(t)} = \hat{y}^{(t-1)} + \eta \cdot h_t(\mathbf{x})$$

Trong đó:

- y^(t) là dự đoán tại vòng boosting thứ t
- ht(x) là cây quyết định học gradient residual
- η là learning rate

Đặc điểm:

- Thuật toán Gradient Boosting dựa trên cây quyết định.
- Huấn luyện nhanh, hiệu quả với dữ liệu lớn.
- Tích hợp xử lý mất cân bằng lớp thông qua scale_pos_weight.

Lý do chọn:

- Thường cho kết quả tốt nhất trong bài toán tín dụng.
- Xử lý mất cân bằng lớp hiệu quả.
- Cho phép đánh giá feature importance để giải thích mô hình.

Ưu nhược điểm:

- Ưu điểm: Nhanh, chính xác, robust với biến nhiễu, dữ liệu lớn.
- Nhược điểm: Ít trực quan, khó giải thích chi tiết các quyết định so với LR hoặc DT.

2.4. Các metric đánh giá mô hình

Để đánh giá hiệu quả của mô hình dự đoán rủi ro tín dụng, đề tài sử dụng nhiều chỉ số đánh giá. Mỗi metric cung cấp một góc nhìn khác nhau về khả năng dự đoán của mô hình, đặc biệt trong bối cảnh dữ liệu mất cân bằng.

2.4.1. Confusion Matrix

Confusion Matrix là ma trận hiển thị số lượng dự đoán đúng/sai của mô hình theo từng lớp.

	Predicted 0	**Predicted 1**
Actual 0	TN	FP
Actual 1	FN	TP

- TP (True Positive): Khách hàng rủi ro được dự đoán đúng.
- TN (True Negative): Khách hàng không rủi ro được dự đoán đúng.

- FP (False Positive): Khách hàng không rủi ro nhưng bị dự đoán là rủi ro.
- FN (False Negative): Khách hàng rủi ro nhưng bị dự đoán là không rủi ro.

Confusion Matrix giúp trực quan hóa các lỗi dự đoán, đặc biệt là FN, vì bỏ sót khách hàng rủi ro có thể gây tổn thất lớn.

2.4.2. Accuracy (Độ chính xác tổng thể)

Accuracy đo tỷ lệ dự đoán đúng trên tổng số dự đoán.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- Ý nghĩa: Phản ánh khả năng dự đoán đúng tổng thể.
- Hạn chế: Khi dữ liệu mất cân bằng, Accuracy có thể đánh giá quá cao lớp chiếm đa số, bỏ qua lớp thiểu số (rủi ro).

2.4.3. Precision (Độ chính xác của lớp rủi ro)

Precision đo tỷ lệ dự đoán rủi ro đúng trên tổng số dự đoán là rủi ro.

$$\text{Precision} = \frac{TP}{TP + FP}$$

- Ý nghĩa: Khi mô hình dự đoán khách hàng rủi ro, tỷ lệ dự đoán đúng là bao nhiêu.
- Ứng dụng: Trong tín dụng, precision cao giúp hạn chế false alarms – không cảnh báo khách hàng an toàn là rủi ro.

2.4.4. Recall (Độ nhạy / Khả năng phát hiện rủi ro)

Recall đo khả năng phát hiện đúng khách hàng rủi ro trên tổng số khách hàng rủi ro thực tế.

$$\text{Recall} = \frac{TP}{TP + FN}$$

- Ý nghĩa: Tỷ lệ khách hàng rủi ro được mô hình phát hiện.

- Ứng dụng: Trong tín dụng, recall cao giúp giảm thiểu false negatives – bỏ sót khách hàng rủi ro, điều này quan trọng để tránh tổn thất.

2.4.5. F1-score

F1-score là trung bình điều hòa của Precision và Recall:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- Ý nghĩa: Kết hợp cả độ chính xác và khả năng phát hiện rủi ro, hữu ích khi dữ liệu mất cân bằng.
- Ứng dụng: Giúp chọn mô hình cân bằng giữa phát hiện rủi ro đúng và hạn chế cảnh báo sai.

2.4.6. ROC-AUC

ROC (Receiver Operating Characteristic) là đường cong thể hiện trade-off giữa True Positive Rate (Recall) và False Positive Rate (FPR):

$$\text{FPR} = \frac{FP}{FP + TN}, \quad \text{TPR} = \text{Recall} = \frac{TP}{TP + FN}$$

AUC (Area Under Curve): Diện tích dưới đường cong ROC, dao động từ 0 đến 1.

- AUC = 0.5 → Mô hình không tốt hơn random.
- AUC → 1 → Mô hình phân biệt hai lớp rất tốt.

Ý nghĩa: ROC-AUC đánh giá khả năng phân biệt khách hàng rủi ro và không rủi ro, đặc biệt quan trọng khi dữ liệu mất cân bằng.

2.4.7. Tổng kết lựa chọn metric

- Recall & F1-score: Ưu tiên để giảm bỏ sót khách hàng rủi ro (FN).
- Precision: Đảm bảo không cảnh báo sai quá nhiều khách hàng an toàn.

- ROC-AUC: Đánh giá tổng quan khả năng phân biệt hai lớp, không phụ thuộc threshold.
- Accuracy: Dùng tham khảo nhưng không ưu tiên vì dữ liệu mất cân bằng.

2.5. Ngưỡng quyết định (Decision Threshold)

Mặc định, mô hình phân loại sử dụng ngưỡng 0.5. Tuy nhiên, trong bài toán rủi ro tín dụng, việc điều chỉnh ngưỡng giúp:

- Tăng Recall cho lớp rủi ro
- Phù hợp với mục tiêu nghiệp vụ

Đề tài sử dụng threshold tuning để tối ưu hiệu quả dự đoán trên tập validation trước khi đánh giá trên tập test.

CHƯƠNG 3: CÀI ĐẶT VÀ THỰC NGHIỆM

3.1. Môi trường cài đặt và dữ liệu thực nghiệm

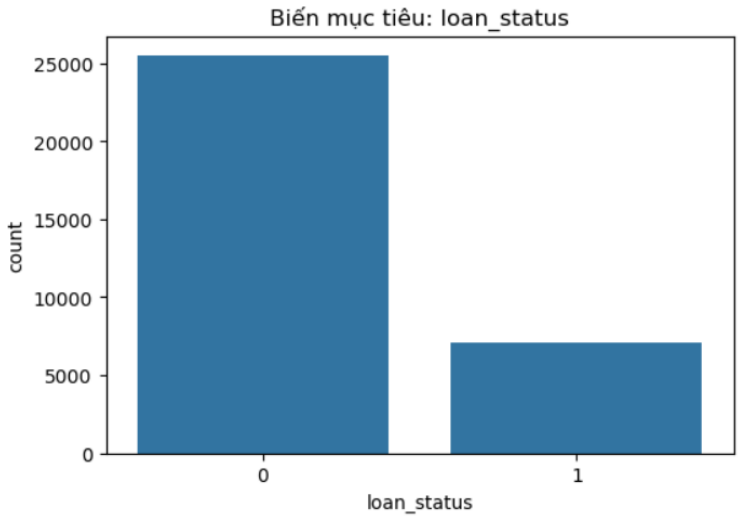
Môi trường cài đặt:

- Ngôn ngữ: Python 3.12.7
- Thư viện chính: pandas, numpy, scikit-learn, lightgbm, imblearn, matplotlib, seaborn
- Công cụ: Jupyter Notebook

Dữ liệu thực nghiệm:

- Tập dữ liệu: credit_risk_dataset.csv
- Số lượng bản ghi: 32,581 dòng
- Số lượng đặc trưng: 12 cột
  + person_age: Tuổi của khách hàng
  + person_income: Thu nhập hàng năm (USD)
  + person_home_ownership: Hình thức sở hữu nhà (RENT/OWN/MORTGAGE...)
  + person_emp_length: Số năm làm việc
  + loan_intent: Mục đích vay (PERSONAL, MEDICAL, EDUCATION...)
  + loan_grade: Xếp hạng tín dụng của khoản vay (A–G)
  + loan_amnt: Số tiền vay
  + loan_int_rate: Lãi suất vay (%)
  + loan_status: 1 = rủi ro (default), 0 = tốt
  + loan_percent_income: Tỷ lệ tiền vay / thu nhập
  + cb_person_default_on_file: Từng vỡ nợ (Y/N)
  + cb_person_cred_hist_length: Thời gian lịch sử tín dụng (năm)
- Biến mục tiêu: loan_status (0 – Nợ tốt, 1 – Nợ xấu)

Phân tích biến mục tiêu:

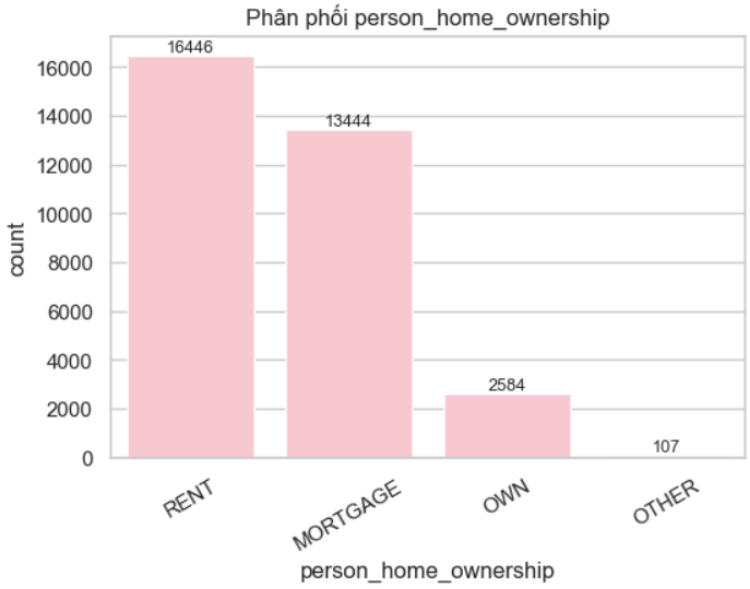


Hình 3.1 Biểu đồ tỷ lệ nợ xấu và nợ tốt

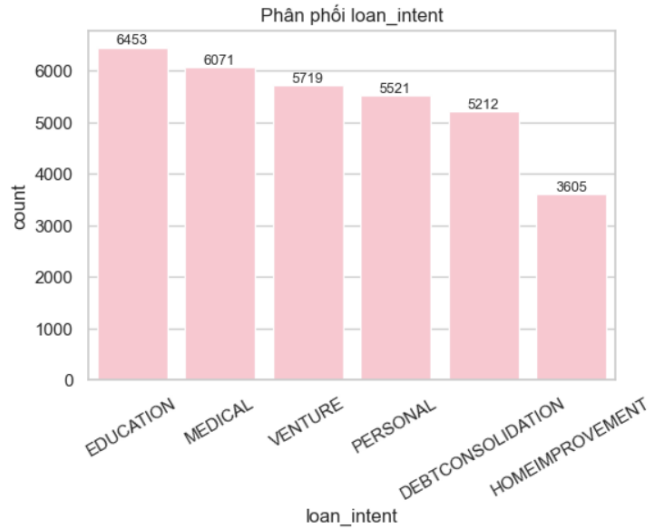
- Nợ xấu: 20%
- Nợ tốt: 80%
- Dữ liệu có sự mất cân bằng lớp.
- Tỷ lệ nhóm 1 (rủi ro cao) thấp hơn đáng kể → cần xử lý imbalance khi train model.

Phân phối biến phân loại (Categorical EDA):

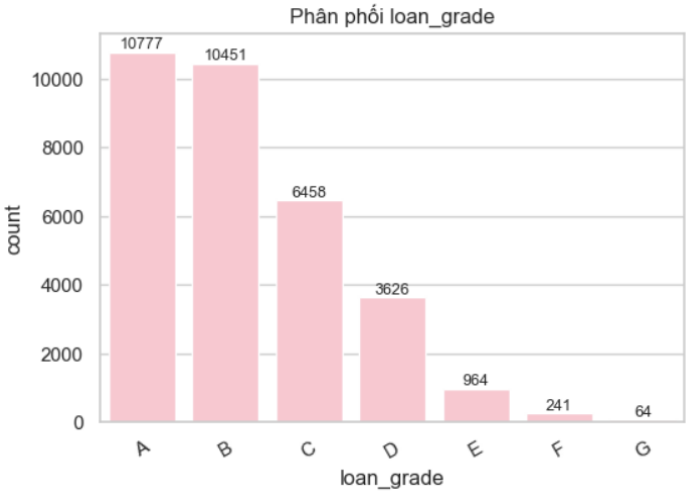
Các biến: person_home_ownership, loan_intent, loan_grade, cb_person_default_on_file



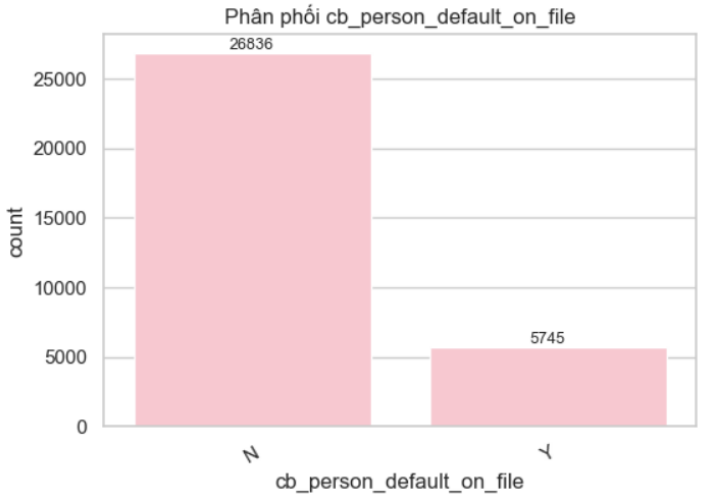
Hình 3.2 Phân phối biến person_home_ownership



Hình 3.3 Phân phối biến loan_intent



Hình 3.4 Phân phối biến loan_grade

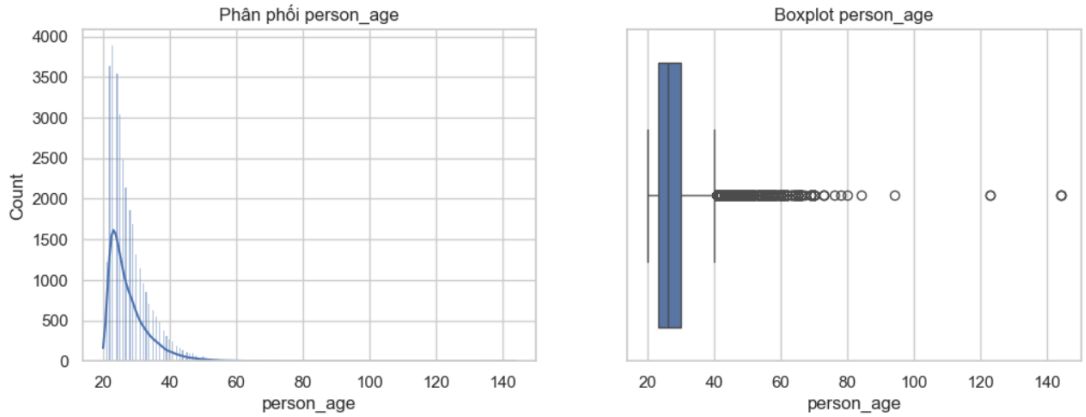


Hình 3.5 Phân phối biến cb_person_default_on_file

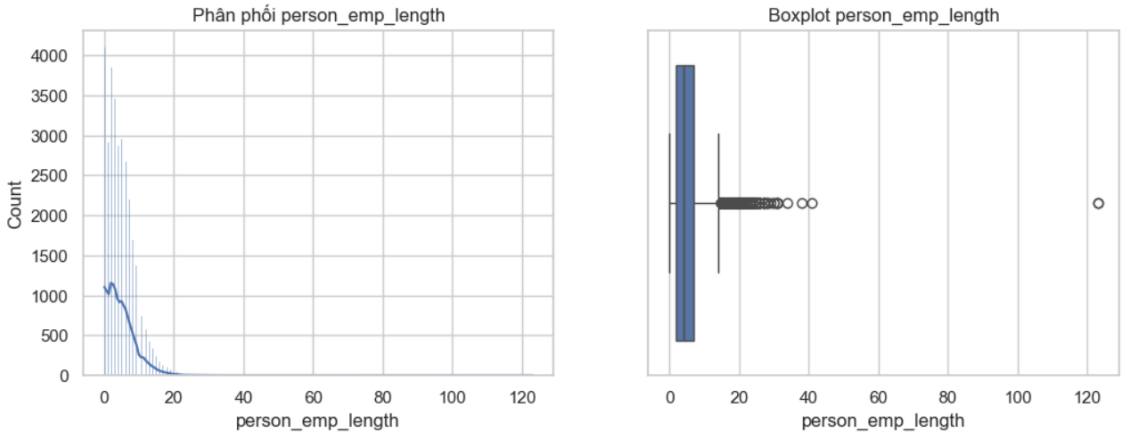
- Hầu hết khách hàng sở hữu nhà riêng (OWN) hoặc thuê(RENT).
- Mục đích vay chủ yếu là debt_consolidation và credit_card.
- Phần lớn khách hàng chưa từng vỡ nợ trước đó (cb_person_default_on_file = N).

Phân phối biến số (Numerical EDA):

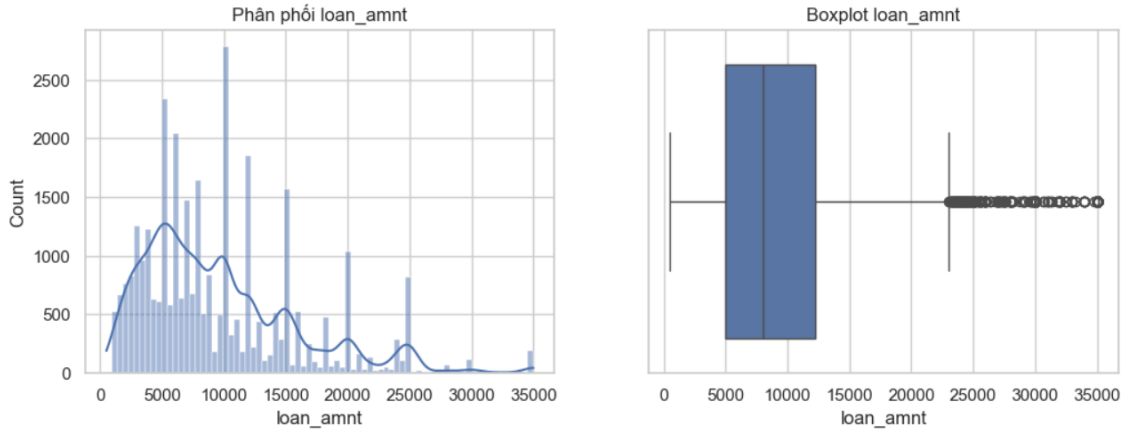
Các biến số: person_age, person_income, person_emp_length, loan_amnt, loan_int_rate, loan_percent_income, cb_person_cred_hist_length được phân tích bằng histplot và boxplot.



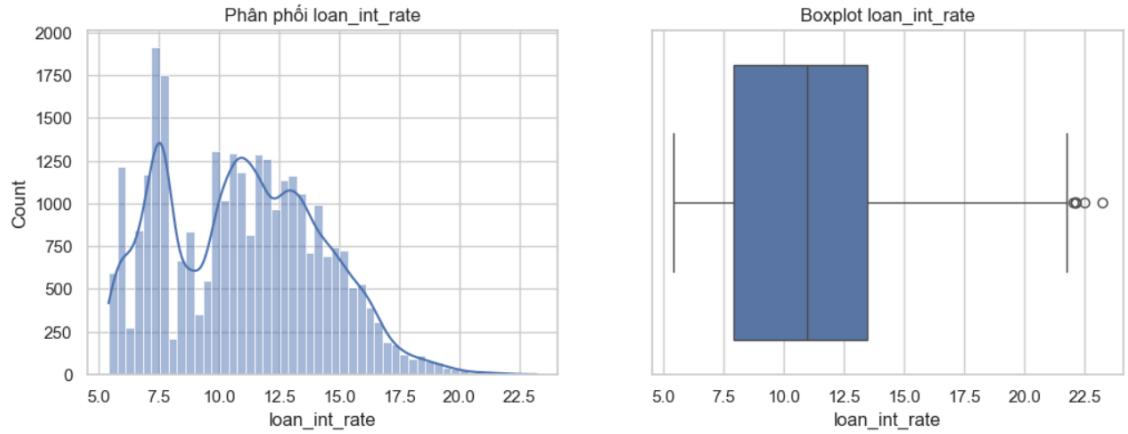
Hình 3.6 Phân phối biến person_age



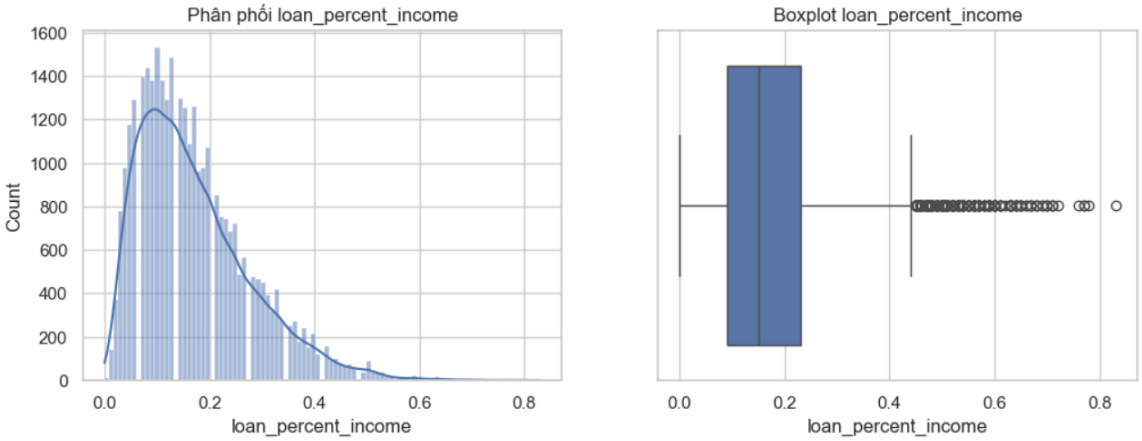
Hình 3.7 Phân phối biến person_emp_length



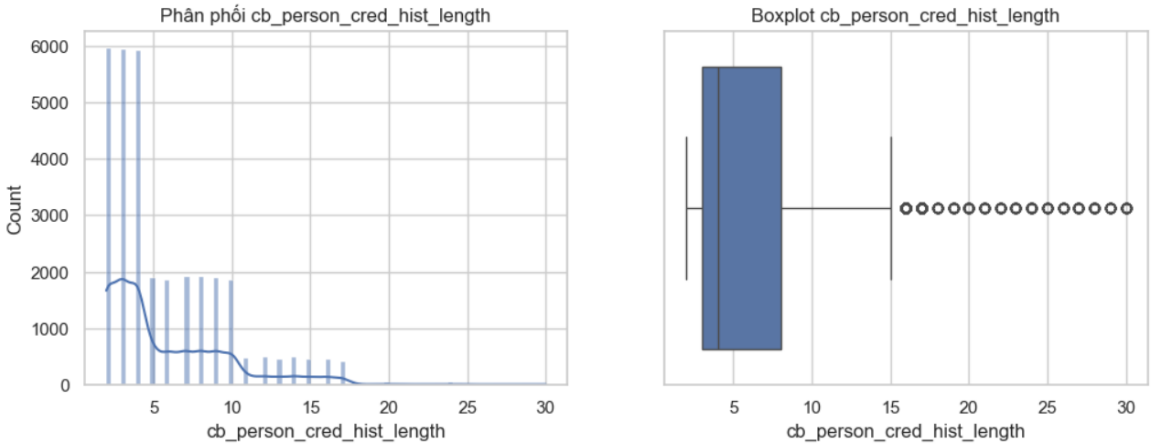
Hình 3.8 Phân phối biến loan_amnt



Hình 3.9 Phân phối biến loan_int_rate



Hình 3.10 Phân phối biến loan_percent_income



Hình 3.11 Phân phối biến cb_person_cred_hist_length

- Phần lớn dữ liệu tập trung ở các giá trị nhỏ, trong khi một số ít giá trị rất lớn tạo ra đuôi phân phối kéo dài về phía bên phải.
- Các biến số liên tục có phân phối lệch phải và xuất hiện nhiều giá trị ngoại lai, thể hiện rõ qua boxplot.
- Hiện tượng này là đặc trưng phổ biến của dữ liệu tài chính và cần được xem xét trong bước tiền xử lý để hạn chế ảnh hưởng đến mô hình.

3.2. Quy trình xử lý dữ liệu (Data Pipeline)

Quy trình tiền xử lý dữ liệu được thực hiện tuần tự nhằm đảm bảo dữ liệu sạch, đồng nhất và phù hợp với mô hình học máy. Các bước chính bao gồm:

3.2.1. Xử lý giá trị thiếu

- Các biến có dữ liệu thiếu: person_emp_length, loan_int_rate
- Giá trị thiếu được điền bằng giá trị trung vị (median) của biến tương ứng.
- Lý do chọn trung vị: ít bị ảnh hưởng bởi outliers và phù hợp với đặc trưng dữ liệu tài chính thường lệch.

3.2.2. Mã hóa biến phân loại (Encoding)

Biến nhị phân:

- cb_person_default_on_file được mã hóa thành 0/1 (N → 0, Y → 1)
- Giúp mô hình hiểu rõ lớp thiểu số (khách hàng từng vỡ nợ).

Biến thứ tự (Ordinal):

- loan_grade được mã hóa từ A–G thành 0–6
- Giữ thứ tự giúp mô hình học được mối quan hệ tuyến tính giữa thứ hạng tín dụng và rủi ro.

Biến không thứ tự (Nominal):

- person_home_ownership, loan_intent được mã hóa bằng One-Hot Encoding
- Không tạo giả định thứ tự, phù hợp với các mô hình Linear hoặc Tree-based.

3.2.3. Chuẩn hóa dữ liệu số

Các biến số liên tục: person_age, person_income, person_emp_length, loan_amnt, loan_int_rate, loan_percent_income, cb_person_cred_hist_length sử dụng StandardScaler (Z-score Normalization)

3.2.4. Xử lý mất cân bằng lớp

Dữ liệu nợ xấu chiếm ~20%, nợ tốt ~80% → dữ liệu mất cân bằng

Các phương pháp xử lý:

- Class Weight: Logistic Regression, Decision Tree, Random Forest. class_weight='balanced' → tăng trọng số cho lớp ít mẫu, giảm bias.
- Scale_pos_weight: LightGBM: Tăng ảnh hưởng của lớp nợ xấu khi tính gradient trong quá trình huấn luyện.

3.2.5. Chia tập dữ liệu

Tỷ lệ chia:

- Train: 70%
- Validation: 15%
- Test: 15%

Sử dụng stratify=y khi chia dữ liệu:

- Đảm bảo tỷ lệ nợ xấu và nợ tốt trong từng tập giữ nguyên như tập gốc
- Tránh tình trạng tập test hoặc validation có quá ít bản ghi lớp thiểu số, ảnh hưởng đến đánh giá mô hình.

3.3. Cài đặt các mô hình dự đoán

Trong đề tài này, bốn mô hình học máy được lựa chọn để thử nghiệm và so sánh hiệu quả dự đoán rủi ro tín dụng: Logistic Regression, Decision Tree, Random Forest và LightGBM. Mỗi mô hình được cấu hình với các tham số (hyperparameters) phù hợp với đặc trưng dữ liệu và mục tiêu dự đoán.

3.3.1. Logistic Regression (LR)

Cấu hình chính:

- solver='lbfgs': Thuật toán tối ưu hóa theo gradient, phù hợp cho bài toán nhị phân và dataset có kích thước vừa phải.
- max_iter=1000: Số vòng lặp tối đa để thuật toán hội tụ; tăng giá trị này nếu gradient descent chưa hội tụ.
- class_weight='balanced': Tự động điều chỉnh trọng số cho từng lớp dựa trên tần suất xuất hiện, giúp xử lý mất cân bằng lớp (nợ xấu ít hơn nợ tốt).

Giải thích:

- Logistic Regression dự đoán xác suất rủi ro dựa trên công thức:

$$P(y = 1 \mid \mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x} + b) = \frac{1}{1 + e^{-(\mathbf{w}^\top \mathbf{x} + b)}}$$

- Ưu điểm: dễ giải thích hệ số wi → biết biến nào ảnh hưởng mạnh đến rủi ro.

3.3.2. Decision Tree (DT)

Cấu hình chính:

- max_depth=6: Giới hạn chiều sâu cây để tránh overfitting.

- class_weight='balanced': Tăng trọng số cho lớp nợ xấu, giúp cây chú ý đến các trường hợp hiếm.

Giải thích:

- Decision Tree xây dựng dựa trên các split của biến nhằm tối đa hóa thông tin thu được (information gain) hoặc giảm Gini impurity:

$$Gini(D) = 1 - \sum_{i=1}^{C} p_i^2$$

- Thuận tiện trực quan hóa, dễ giải thích các quyết định rủi ro.

3.3.3. Random Forest (RF)

Cấu hình chính:

- n_estimators=200: Số lượng cây trong ensemble, càng nhiều cây → variance giảm nhưng tốn thời gian tính toán.
- max_depth=10: Giới hạn chiều sâu cây con, giảm overfitting.
- class_weight='balanced': Như Decision Tree, giúp xử lý mất cân bằng lớp.
- n_jobs=-1: Sử dụng tất cả CPU để tăng tốc huấn luyện.

Giải thích:

- Random Forest là tập hợp nhiều cây quyết định (ensemble) dựa trên kỹ thuật bagging: mỗi cây học trên bootstrap sample riêng và chỉ xem một tập con các biến khi split.
- Giảm overfitting, mạnh mẽ với dữ liệu tabular phức tạp và nhiều biến.

3.3.4. LightGBM

Cấu hình chính:

- n_estimators=300: Số lượng cây boosting, tăng số cây có thể cải thiện độ chính xác nhưng làm chậm huấn luyện.

- learning_rate=0.05: Tốc độ học (shrinkage); giá trị nhỏ → hội tụ chậm nhưng chính xác hơn.
- max_depth=6: Giới hạn chiều sâu cây để kiểm soát overfitting.
- scale_pos_weight = (#nợ tốt / #nợ xấu): Điều chỉnh gradient cho lớp nợ xấu, xử lý mất cân bằng lớp hiệu quả.
- random_state=42: Đảm bảo kết quả có thể tái lập.

Giải thích:

- LightGBM là thuật toán Gradient Boosting dựa trên cây với kỹ thuật leaf-wise: chọn lá có gain lớn nhất để chia, giúp tối ưu nhanh và hiệu quả.
- Hỗ trợ early stopping, regularization, và có thể trích xuất feature importance để giải thích mô hình.

3.4. Đánh giá và so sánh kết quả

3.4.1. Tổng quan kết quả trên Train và Validation

	Accuracy	Precision	Recall	ROC-AUC	Model	Dataset
0	0.788959	0.510692	0.777688	0.862484	Logistic Regression	Train
1	0.792306	0.515577	0.791745	0.869724	Logistic Regression	Validation
2	0.914233	0.849179	0.737889	0.908104	Decision Tree	Train
3	0.912625	0.844660	0.734522	0.907646	Decision Tree	Validation
4	0.938569	0.924667	0.782111	0.957813	Random Forest	Train
5	0.921629	0.876516	0.745779	0.933901	Random Forest	Validation
6	0.944225	0.883413	0.857487	0.979264	LightGBM	Train
7	0.924289	0.847305	0.796435	0.952654	LightGBM	Validation

Hình 3.12 Đánh giá trên tập train/val

Nhận xét:

- Logistic Regression
  + Đơn giản, hiệu suất vừa phải.

+ Recall ~0.79 trên validation cho thấy mô hình nhận diện nợ xấu tương đối ổn, nhưng Precision thấp (~0.52), nghĩa là nhiều dự đoán dương tính là false positive.
+ Đây là mô hình benchmark để so sánh.

- Decision Tree
  + Accuracy và Precision cao hơn Logistic Regression, nhưng Recall vẫn chưa tối ưu (0.735 trên validation).
  + Có xu hướng overfitting trên train (Accuracy train 0.914 vs Validation 0.913), nhưng chưa nghiêm trọng nhờ giới hạn max_depth=6.
- Random Forest
  + Độ chính xác, Precision cao hơn cả Decision Tree.
  + Recall ~0.746 trên validation, tốt hơn cây đơn, nhờ ensemble giảm overfitting.
  + ROC-AUC 0.934, thể hiện khả năng phân biệt hai lớp tốt hơn.
- LightGBM
  + Đạt hiệu suất tổng thể cao nhất:
  + Recall 0.796 trên validation → nhận diện nợ xấu tốt hơn các mô hình khác.
  + ROC-AUC 0.953 → khả năng phân biệt nợ xấu/nợ tốt tốt nhất.
  + Pipeline xử lý mất cân bằng lớp (scale_pos_weight) giúp cải thiện nhận diện lớp thiểu số.

3.4.2. Đánh giá trên tập Test

Kết quả trên Test set với threshold tối ưu 0.4 cho LightGBM:

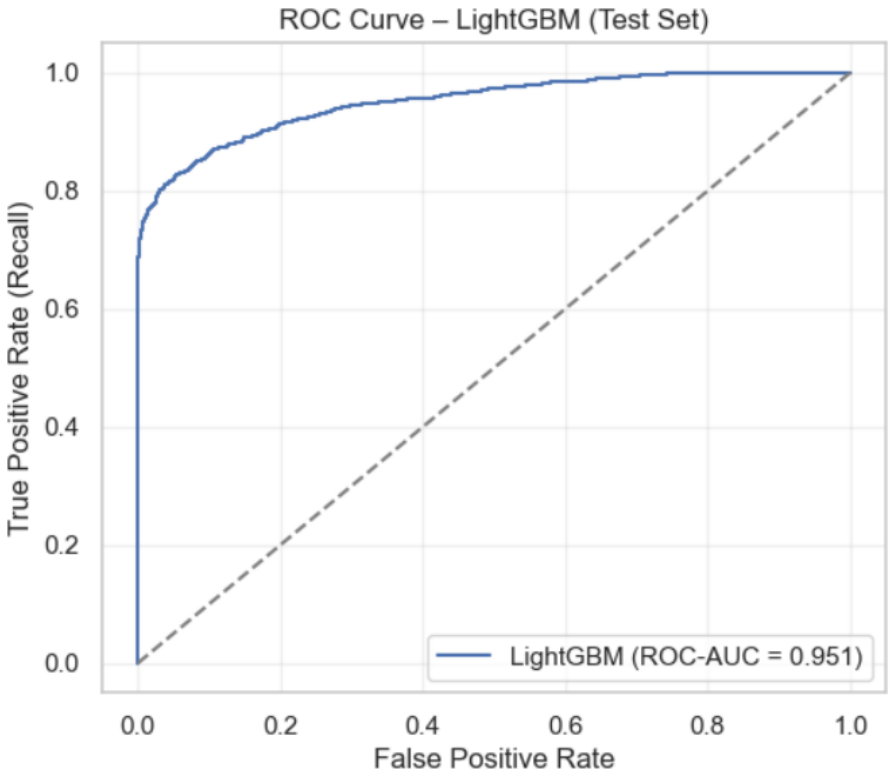
	Model	Accuracy	Precision	Recall	ROC-AUC
0	Logistic Regression	0.791735	0.515040	0.786317	0.868276
1	Decision Tree	0.913871	0.844350	0.742268	0.905213
2	Random Forest	0.921031	0.868906	0.751640	0.930776
3	LightGBM	0.894640	0.716641	0.855670	0.950867

Hình 3.13 Đánh giá trên tập test

Nhận xét:

- LightGBM vẫn giữ hiệu quả cao nhất, nhất là Recall và ROC-AUC, quan trọng trong bài toán tín dụng vì giảm thiểu false negative (bỏ sót nợ xấu).
- Random Forest cũng ổn nhưng Recall thấp hơn, Logistic Regression dễ triển khai nhưng Precision thấp.

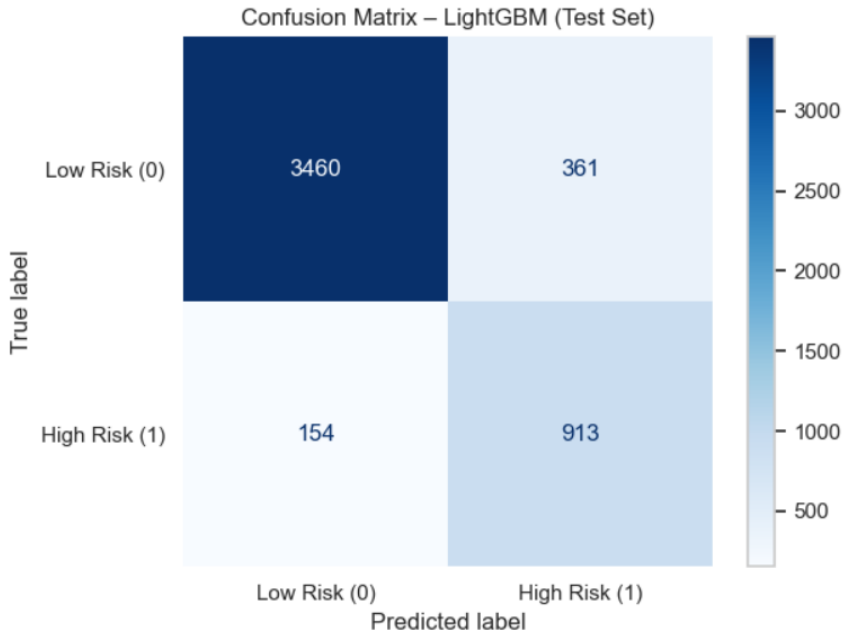
3.4.3 Đường ROC (Test Set – LightGBM):



Hình 3.14 Đường ROC-AUC với lightGBM

Đường ROC của mô hình LightGBM nằm cao và tách biệt rõ so với đường chéo tượng trưng cho dự đoán ngẫu nhiên. Với ROC-AUC ≈ 0.95 trên tập test, mô hình cho thấy khả năng phân biệt xuất sắc giữa khách hàng rủi ro và không rủi ro trên toàn bộ các ngưỡng dự đoán. Điều này chứng tỏ mô hình không chỉ hoạt động tốt tại một ngưỡng nhất định mà còn ổn định khi thay đổi decision threshold.

3.4.4 Confusion Matrix (Test Set – LightGBM, threshold = 0.4):



Hình 3.15 Confusion Matrix (Test Set – LightGBM, threshold = 0.4)

Confusion Matrix minh họa rằng với threshold = 0.4, mô hình LightGBM phát hiện hiệu quả khách hàng rủi ro, thể hiện qua số lượng False Negative thấp. Điều này phù hợp với mục tiêu của bài toán tín dụng, nơi việc bỏ sót khách hàng rủi ro gây hậu quả nghiêm trọng hơn so với dự đoán nhầm khách hàng an toàn.

- True Positive (rủi ro cao dự đoán đúng): 913
- True Negative (rủi ro thấp dự đoán đúng): 3460
- False Positive: 361
- False Negative: 154

KẾT LUẬN

Kết quả đạt được

Qua quá trình thực nghiệm với 4 mô hình học máy (Logistic Regression, Decision Tree, Random Forest, LightGBM), kết quả cho thấy:

- LightGBM nổi bật với độ chính xác cao nhất trên tập validation và test, ROC-AUC khoảng 0.95, khả năng phát hiện khách hàng rủi ro (Recall) tốt và giảm số lượng False Negative, phù hợp với mục tiêu hạn chế tổn thất tín dụng.
- Random Forest đạt hiệu suất tốt, giảm overfitting so với Decision Tree đơn lẻ, đồng thời robust với dữ liệu nhiễu và nhiều biến số.
- Decision Tree trực quan, dễ giải thích nhưng có xu hướng overfitting khi cây quá sâu.
- Logistic Regression đơn giản, dễ hiểu và nhanh, phù hợp làm benchmark, tuy nhiên độ chính xác và khả năng phát hiện khách hàng rủi ro kém hơn các mô hình ensemble.

Pipeline kết hợp tiền xử lý với mô hình học máy giúp đảm bảo đồng nhất giữa train, validation và test, xử lý mất cân bằng lớp hiệu quả, đồng thời giữ tính minh bạch trong quá trình huấn luyện và dự đoán.

Hạn chế của đề tài

- Dữ liệu hạn chế: Tập dữ liệu chỉ gồm 32,581 bản ghi và 12 đặc trưng, có thể chưa phản ánh đầy đủ sự đa dạng của khách hàng thực tế.
- Mô hình chưa thử nghiệm sâu với dữ liệu thời gian: Các biến lịch sử khách hàng không được mô hình hóa theo trình tự thời gian, nên chưa khai thác được các quan hệ động.
- Giải thích mô hình phức tạp: Các mô hình ensemble như Random Forest và LightGBM khó giải thích chi tiết quyết định của từng cá nhân, mặc dù có thể xem feature importance tổng quan.
- Xử lý outliers và biến lệch: Một số biến số vẫn còn outliers nhẹ hoặc phân phối lệch, ảnh hưởng tiềm năng đến mô hình.

Hướng phát triển của đề tài

- Mở rộng dữ liệu: Thu thập thêm khách hàng từ các nguồn khác, hoặc dữ liệu dài hạn để tăng khả năng tổng quát.
- Thử nghiệm mô hình nâng cao: Sử dụng Deep Learning như MLP hoặc LSTM để khai thác các quan hệ phi tuyến phức tạp và dữ liệu tuần tự nếu có.
- Tối ưu hyperparameter nâng cao: Áp dụng Bayesian Optimization hoặc Random Search để tối ưu các tham số mô hình.
- Xử lý mất cân bằng nâng cao: Thử SMOTE, ADASYN hoặc học có chi phí nhạy cảm để cải thiện phát hiện khách hàng rủi ro mà vẫn giữ khách hàng an toàn.
- Giải thích mô hình (Explainable AI): Sử dụng SHAP, LIME để hiểu rõ hơn ảnh hưởng của từng đặc trưng đến quyết định tín dụng, tăng độ minh bạch cho ngân hàng hoặc tổ chức tài chính.

TÀI LIỆU THAM KHẢO

[1] Tiệp, V. H. (2018). Machine Learning Cơ bản [Ebook]. GitHub repository. https://github.com/tiepvupsu/ebookMLCB

[2] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825–2830. https://arxiv.org/abs/1201.0490

[3] Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems (2nd ed.). O'Reilly Media.

[4] Brownlee, J. (2020). Machine Learning Mastery with Python: Understand Your Data, Create Accurate Models, and Work Projects End-to-End. Machine Learning Mastery.

[5] Raschka, S., & Mirjalili, V. (2019). Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2 (3rd ed.). Packt Publishing.

[6] TiepVuPSU. (2018). Machine Learning Cơ bản – Mã nguồn minh họa và ví dụ [GitHub repository]. https://github.com/tiepvupsu/ebookMLCB