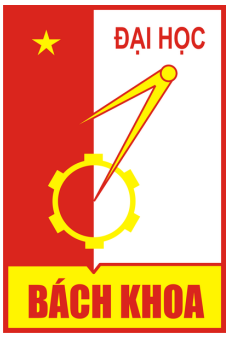
ĐẠI HỌC BÁCH KHOA HÀ NỘI
KHOA TOÁN - TIN



BÁO CÁO BÀI TẬP LỚN

Môn học: Hệ Hỗ trợ quyết định

Phân loại khách hàng theo khả năng thanh toán

Giảng viên hướng dẫn: TS. Trần Ngọc Thăng

Sinh viên thực hiện: Trương Việt Dũng 20206278

Đoàn Đức Thanh 20206261

Nhóm: 16

Hà Nội, tháng 6 năm 2024

NHẬN XÉT CỦA GIẢNG VIÊN

1. Mục tiêu

(a)

(b)

(c)

2. Nội dung

(a)

(b)

(c)

3. Đánh giá kết quả đạt được

(a)

(b)

(c)

Hà Nội, ngày ... tháng ... năm 2024

Giảng viên hướng dẫn

TS. Trần Ngọc Thăng

Lời Cảm Ơn

Báo cáo này được thực hiện và hoàn thành tại Đại học Bách Khoa Hà Nội, nằm trong nội dung học phần Hệ Hỗ trợ quyết định của kì học 2023-2.
Chúng em xin được dành lời cảm ơn chân thành tới TS. Trần Ngọc Thăng, là giảng viên đã trực tiếp hướng dẫn và gợi ý cho em đề tài rất thú vị này, đồng thời thầy cũng đã giúp đỡ tận tình và có những góp ý, định hướng bổ ích để chúng em hiểu hơn về đề tài, từ đó có thể hoàn thành báo cáo này một cách tốt nhất.

Hà Nội, tháng 06 năm 2024

Tóm tắt nội dung báo cáo

Trong báo cáo này, chúng em đã tiến hành nghiên cứu và thực nghiệm để dự đoán khả năng thanh toán đúng hạn của khách hàng dựa trên các thông tin cá nhân, tài chính, lịch sử tín dụng và các yếu tố khác, sử dụng các kỹ thuật học máy. Sau khi tìm kiếm và thu thập được bộ dữ liệu phù hợp, chúng em đã tiến hành lần lượt các bước xử lý dữ liệu, phân tích và khám phá dữ liệu, chọn và huấn luyện mô hình, đánh giá mô hình, triển khai và cải tiến mô hình. Sau quá trình nghiên cứu chúng em đã áp dụng thành công 6 thuật toán học máy để thử nghiệm và thu được kết quả tương đối khả quan. Chúng em cũng tiếp tục thử nghiệm thêm nhiều cách để cải tiến và tối ưu mô hình nhằm hạn chế tính trạng overfitting và nâng cao hiệu quả dự đoán. Kết quả cuối cùng cho thấy mỗi mô hình đều có những điểm mạnh khi áp dụng vào bài toán thực tế, tuy nhiên vẫn sẽ cần cải thiện và nâng cấp thêm để có thể đạt được mục tiêu ứng dụng trong tương lai.

Mục lục

Chương 1

Giới thiệu

1.1 Động cơ thực hiện báo cáo

Vấn đề quản lý rủi ro tín dụng cũng như thực hiện đánh giá và phân tích định lượng về rủi ro tín dụng và xếp hạng người đi vay có liên quan đến tất cả các ngân hàng tham gia cho vay đối với cá nhân và pháp nhân. Nhìn chung, khi ngân hàng thương mại cấp tín dụng cho cá nhân và pháp nhân, rủi ro tín dụng liên quan được đặc trưng bởi các thông số định lượng sau: rủi ro là xác suất người đi vay không trả được nợ; rủi ro chấp nhận được; nguy cơ trung bình; tổn thất có thể xảy ra do vỡ nợ; giá trị tổn thất trung bình; mức tổn thất tối đa cho phép; số lượng khoản vay mà ngân hàng đưa ra; số lượng các khoản vay khác nhau mà ngân hàng có thể cung cấp; số lượng các khoản vay có vấn đề.

Do đó, quản lý rủi ro tín dụng của danh mục tín dụng là một trong những nhiệm vụ quan trọng nhất đối với tính thanh khoản tài chính và sự ổn định của ngành ngân hàng do độ nhạy cảm ngày càng tăng của các ngân hàng đối với rủi ro tín dụng và những thay đổi trong diễn biến giá của các công cụ tài chính. Tác động đáng kể nhất đến hiệu quả hoạt động của doanh nghiệp chỉ là rủi ro tài chính. Rủi ro phi hệ thống có tác động đến hiệu quả hoạt động của doanh nghiệp cao hơn rủi ro hệ thống. Việc xác định từng khoản vay, hoặc người đi vay, kỹ thuật đánh giá rủi ro đóng vai trò chính trong việc quản lý và giảm thiểu rủi ro tín dụng. Chỉ sau khi xác định được rủi ro của từng cá nhân người vay và của mỗi dịch vụ tín dụng riêng lẻ mà người ta có thể bắt đầu quản lý toàn bộ danh mục cho vay. Đánh giá

rủi ro tín dụng của người đi vay bao gồm việc nghiên cứu và đánh giá các chỉ số định tính và định lượng về tình hình kinh tế của người đi vay. Việc đánh giá các yếu tố rủi ro liên quan đến việc cấp một khoản vay cụ thể và phân tích toàn diện và có hệ thống của chúng cho phép ngân hàng tính đến các yếu tố này trong quản lý rủi ro tín dụng và ngăn ngừa tác động bất lợi và thường xuyên của chúng đến kết quả hoạt động trong tương lai của ngân hàng. Các phương pháp được sử dụng để định lượng rủi ro tín dụng đi kèm với yêu cầu đặc biệt về tính minh bạch, bao gồm đánh giá định lượng về độ chính xác của phương pháp và đặc tính của phương pháp thống kê được gọi là độ tin cậy. Tính minh bạch của phương pháp luận về rủi ro tín dụng mang lại cơ hội xem xét một hiện tượng nhất định không chỉ một cách tổng thể mà còn chi tiết. Tính minh bạch đã trở thành đặc điểm quan trọng nhất của các phương pháp đánh giá rủi ro tín dụng nhờ nhu cầu xác định kỹ lưỡng nhất cả rủi ro tín dụng và bản thân mô hình rủi ro tín dụng. Tính minh bạch về phương pháp đề cập đến độ chính xác của các phương pháp toán học được sử dụng, việc giảm thiểu yếu tố chủ quan trong đánh giá của chuyên gia, sự rõ ràng của kết quả đánh giá và phân tích rủi ro, sự hiểu biết thấu đáo của nhân viên ngân hàng về những kết quả này và khả năng tiếp cận các thông tin nhất định. các biện pháp đối với các cơ quan quản lý và người đi vay. Để phân tích, dự báo và quản lý rủi ro tín dụng, mỗi ngân hàng phải có khả năng định lượng các yếu tố rủi ro tín dụng liên quan, phân tích rủi ro liên quan và giám sát thường xuyên các yếu tố rủi ro tín dụng. Các quyết định của ngân hàng về việc cấp hoặc từ chối cấp một khoản vay, về lãi suất và mức trích lập dự phòng rủi ro cho khoản vay sẽ phụ thuộc vào tính chính xác của việc nhận biết và đánh giá rủi ro.

Việc tạo ra một mô hình đánh giá rủi ro hiệu quả và quản lý rủi ro tín dụng thành công chỉ có thể thực hiện được nhờ phân tích định lượng liên tục các thông tin thống kê về thành công tín dụng. Có nhiều cách tiếp cận khác nhau để xác định rủi ro tín dụng do một người đi vay cụ thể gây ra như đánh giá chủ quan của các chuyên gia ngân hàng và hệ thống đánh giá rủi ro tự động. Tuy nhiên, kinh nghiệm toàn cầu cho thấy các hệ thống đánh giá rủi ro tín dụng dựa trên các mô hình toán học hiệu quả và đáng tin cậy hơn bất kỳ hệ thống nào khác. Vì vậy việc nghiên cứu và phát triển các mô hình học máy nhằm hỗ trợ đánh giá rủi ro tín dụng mang lại rất nhiều giá trị thực tiễn.

1.2 Đối tượng và phạm vi nghiên cứu

Dữ liệu được dùng để nghiên cứu là dữ liệu về thông tin của những khách hàng vay nợ và thông tin trả nợ của họ. Bộ dữ liệu bao gồm 16 cột và 2895 hàng trong đó 15 cột là thông tin khách hàng và cột còn lại là nhãn với nhã 0 là trả nợ đúng hạn và 1 là trả nợ trễ hạn. Bài toán sẽ nằm trong phạm vi nghiên cứu của bài toán phân loại 2 lớp.

1.3 Nghiên cứu liên quan

Do tính chất quan trọng của mình mà vấn đề dự báo rủi ro tín dụng được rất nhiều tổ chức cá nhân quan tâm và nghiên cứu. Một trong số đó là bài báo "Machine Learning for Credit Risk Prediction: A Systematic Literature Review" được thực hiện chính bởi tác giả Jomark Pablo Noriaga.Đây là một đánh giá có hệ thống việc sử dụng Machine Learning (ML) để dự đoán rủi ro tín dụng, chúng tôi nêu lên nhu cầu các tổ chức tài chính sử dụng Trí tuệ nhân tạo (AI) và ML để đánh giá rủi ro tín dụng, phân tích khối lượng lớn thông tin. Họ đã đặt ra các câu hỏi nghiên cứu về thuật toán, số liệu, kết quả, bộ dữ liệu, biến số và các hạn chế liên quan trong việc dự đoán rủi ro tín dụng. [1] Chúng em đã tiến hành thực hiện báo cáo này dựa trên định hướng và cách tiếp cận của bài báo trên.

1.4 Mục tiêu của Báo cáo

Báo cáo này gồm có 3 mục tiêu quan trọng:

- **1.** Đánh nhãn các trường không phải số và thực hiện xử lý để dữ liệu sạch và thuận tiện cho quá trình nghiên cứu.
- **2.** Đề xuất mô hình và các tiêu chí đánh giá.
- **3.** Cải tiến và tối ưu hiệu suất của các mô hình nghiên cứu.

1.5 Kết cấu Báo cáo

Báo cáo này sẽ được trình bày gồm 4 chương như sau:

- Chương 1: Giới thiệu về vấn đề nghiên cứu, đối tượng và phạm vi nghiên cứu,phương pháp nghiên cứu.
- Chương 2: Tiền xử lý, phân tích và khám phá dữ liêu.
- Chương 3: Xây dựng mô hình và kết quả thực nghiệm.
- Chương 4: Kết luận và hướng phát triển đề tài.

Chương 2

Tiền xử lý, phân tích và khám phá dữ liệu

2.1 Phân tích và khám phá dữ liệu - EDA

Trước hết, ta cần nạp vào các gói cần thiết trong Python để làm việc với các bộ dữ liệu, sau đó là dữ liệu từ các tập tin Excel.

```
# Import dataset and libraries
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt

file_path = "/content/Data_credit.xlsx"
data = pd.read_excel(file_path)
```

```
from copy import deepcopy
data_0 = deepcopy(data)
```

Lệnh sau đây cho ta thấy năm dòng đầu bộ dữ liệu:

```
data_0.head(5)
```

	ID khách hàng	Khả năng trả nợ	Giới tính	Độ tuổi	Tình trạng hôn nhân	Tình trạng công việc hiện tại (số năm kinh nghiệm)	Thu nhập	Chi tiêu (trung bình một tháng)	Hóa đơn tiền điện	Tài sản thế chấp	Tình trạng sở hữu nhà ở	Số tiền dự kiến vay	Mục đích vay	Thời hạn khoản vay	Chứng minh thu nhập	Số thành viên phụ thuộc trong gia đình	Trình độ học vấn
0	0	0. Trả nợ đúng hạn	0. Nữ	37	1. Có gia đình	7.0	30000000	15000000	600000.0	0. Có tài sản thế chấp	0. Đã sở hữu nhà ở	250000000	1. Tiêu dùng	12	0. Có giấy tờ chứng minh thu nhập	1	2. Thạc sĩ
1	1	0. Trả nợ đúng hạn	0. Nữ	41	1. Có gia đình	8.0	35000000	19000000	650000.0	0. Có tài sản thế chấp	0. Đã sở hữu nhà ở	250000000	1. Tiêu dùng	12	0. Có giấy tờ chứng minh thu nhập	2	3. Đại học
2	2	0. Trả nợ đúng hạn	1. Nam	33	1. Có gia đình	6.0	35000000	20000000	600000.0	0. Có tài sản thế chấp	0. Đã sở hữu nhà ở	350000000	1. Tiêu dùng	12	0. Có giấy tờ chứng minh thu nhập	2	2. Thạc sĩ
3	3	0. Trả nợ đúng hạn	1. Nam	36	1. Có gia đình	7.0	35000000	4500000	750000.0	0. Có tài sản thế chấp	0. Đã sở hữu nhà ở	300000000	1. Tiêu dùng	12	0. Có giấy tờ chứng minh thu nhập	2	1. Tiến sĩ
4	4	0. Trả nợ đúng hạn	1. Nam	35	1. Có gia đình	5.0	35000000	4500000	650000.0	0. Có tài sản thế chấp	0. Đã sở hữu nhà ở	350000000	1. Tiêu dùng	8	0. Có giấy tờ chứng minh thu nhập	3	1. Tiến sĩ

Hình 2.1: Năm dòng đầu bộ dữ liệu

2.1.1 Quan sát số lượng và các thuộc tính của bộ dữ liệu

Lệnh `data_0.info()` cho phép ta theo dõi và đếm số các thuộc tính của toàn bộ dữ liệu:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3006 entries, 0 to 3005
Data columns (total 17 columns):
```

TT	Column	Non-Null Count	Dtype
0	ID khách hàng	3006 non-null	int64
1	Khả năng trả nợ	3006 non-null	object
2	Giới tính	3006 non-null	object
3	Độ tuổi	3006 non-null	int64
4	Tình trạng hôn nhân	2992 non-null	object
5	Tình trạng công việc hiện tại (số năm kinh nghiệm)	2990 non-null	float64
6	Thu nhập	3006 non-null	int64
7	Chi tiêu (trung bình một tháng)	3006 non-null	int64
8	Hóa đơn tiền điện	2992 non-null	float64
9	Tài sản thế chấp	3006 non-null	object
10	Tình trạng sở hữu nhà ở	3006 non-null	object
11	Số tiền dự kiến vay	3006 non-null	int64
12	Mục đích vay	3006 non-null	object

13	Thời hạn khoản vay	3006 non-null	int64
14	Chứng minh thu nhập	3006 non-null	object
15	Số thành viên phụ thuộc trong gia đình	3006 non-null	int64
16	Trình độ học vấn	2985 non-null	object

Bảng 2.1: Danh sách thuộc tính và số lượng

Ta có thể đưa ra nhận xét như sau:

- Bộ dữ liệu bao gồm 3006 yêu cầu vay từ các khách hàng;
- Mỗi khách hàng được mô tả với 16 thuộc tính (trừ biến "ID khách hàng"). Trong 16 thuộc tính đó,
  - 7 thuộc tính có kiểu dữ liệu là `int64`;
  - 8 thuộc tính có kiểu dữ liệu là `object`;
  - 2 thuộc tính có kiểu dữ liệu là `float64`.
- Các thuộc tính sau bị thiếu một vài giá trị: "Tình trạng hôn nhân", "Tình trạng công việc hiện tại (số năm kinh nghiệm)", "Hoá đơn tiền điện", "Trình độ học vấn".

Do thuộc tính "ID khách hàng" không đem lại hiệu quả đáng kể khi huấn luyện mô hình, ta có thể loại bỏ nó.

```
1 data_1 = pd.DataFrame(data_0)
2 data_1 = data_0.iloc[:,1:]
```

2.1.2 Thống kê mô tả

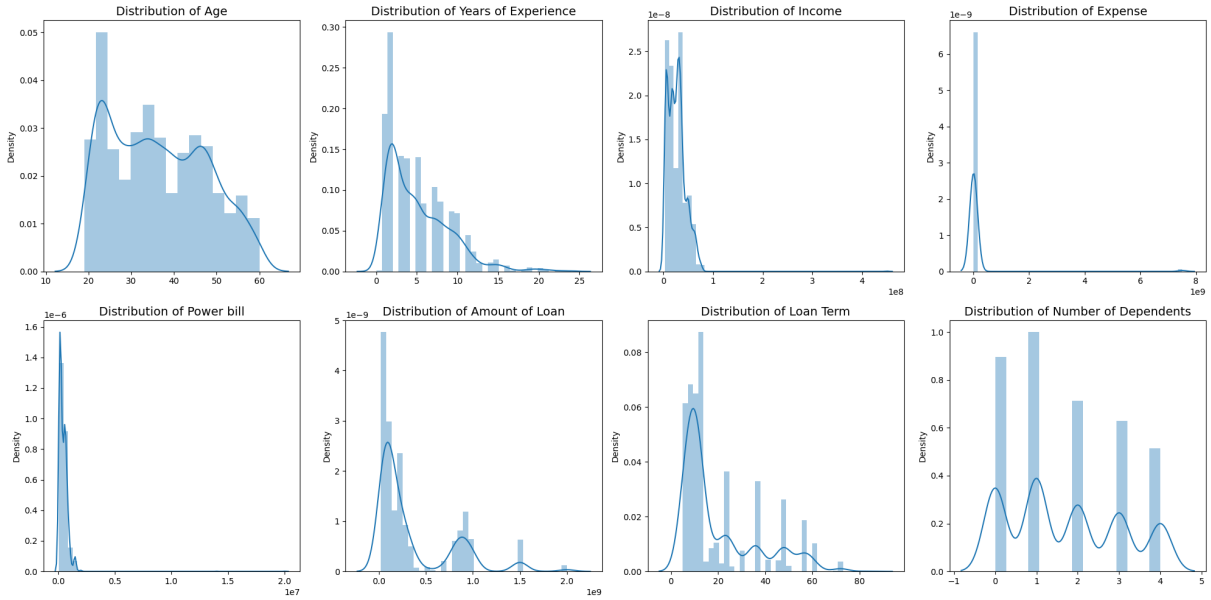
Phân tích dữ liệu số

Qua quan sát, ta xác định được các trường là dữ liệu số. Chúng em tiến hành lấy các trường này ra để thuận lợi cho việc phân tích. Hàm `describe()` được sử dụng để lấy ra thống kê mô tả của cá trường này:

	Độ tuổi	Tình trạng công việc hiện tại (số năm kinh nghiệm)	Thu nhập	Chi tiêu (trung bình một tháng)	Hóa đơn tiền điện	Số tiền dự kiến vay	Thời hạn khoản vay	Số thành viên phụ thuộc trong gia đình
count	3006.000000	2990.000000	3.006000e+03	3.006000e+03	2.992000e+03	3.006000e+03	3006.000000	3006.000000
mean	36.365935	5.175251	2.687435e+07	8.018328e+07	4.650780e+05	3.529910e+08	19.924484	1.697605
std	11.200004	3.865598	1.849750e+07	7.196772e+08	5.419622e+05	4.097955e+08	16.457092	1.359030
min	19.000000	0.000000	3.000000e+06	1.000000e+06	9.000000e+04	2.000000e+07	5.000000	0.000000
25%	26.000000	2.000000	1.500000e+07	3.000000e+06	1.900000e+05	7.500000e+07	8.000000	1.000000
50%	35.000000	4.000000	2.700000e+07	5.000000e+06	3.500000e+05	1.500000e+08	12.000000	1.000000
75%	46.000000	7.000000	3.500000e+07	1.500000e+07	6.500000e+05	5.000000e+08	24.000000	3.000000
max	60.000000	24.000000	4.500000e+08	7.500000e+09	2.000000e+07	2.000000e+09	84.000000	4.000000

Hình 2.2: Thống kê mô tả dữ liệu số

Để có thể quan sát trực quan hơn, chúng em thực hiện vẽ biểu đồ phân bố của các trường dữ liệu:



Hình 2.3: Biểu đồ phân phối dữ liệu số

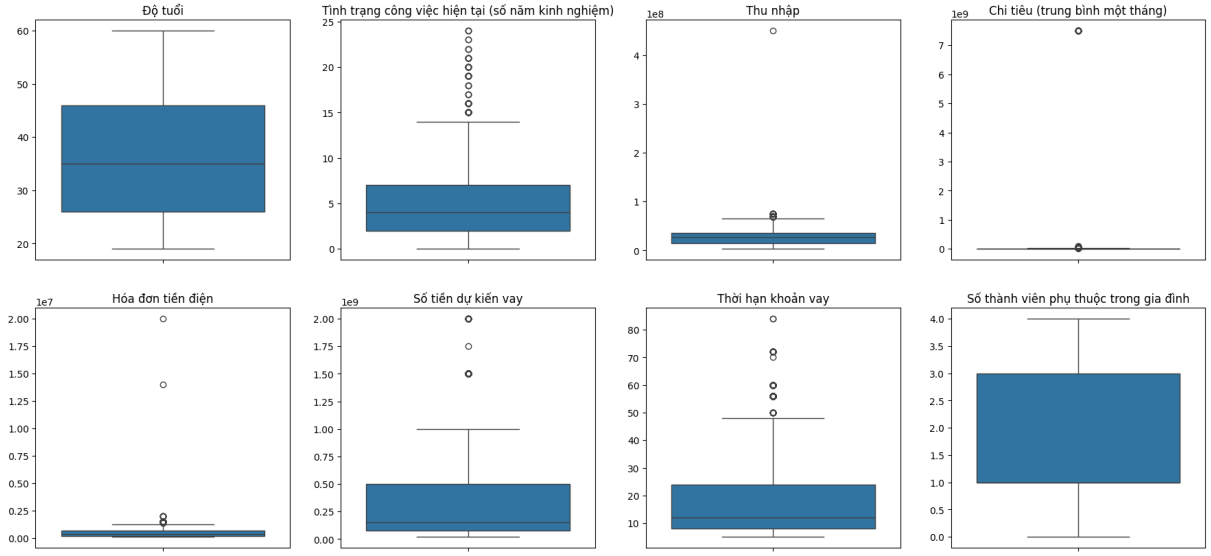
Qua đó, chúng em cũng hiểu rõ hơn về thông tin các trường số. Ý nghĩa của cá trường sẽ được mô tả như sau:

- **Độ tuổi**: Mẫu khách hàng nghiên cứu có độ tuổi từ 19 đến 60, độ tuổi trung bình của tất cả khách hàng là 36 tuổi.
- **Tình trạng công việc hiện tại (số năm kinh nghiệm)**: Bên vay có số năm kinh nghiệm (thâm niên nghề nghiệp) từ 0 đến 24 năm, trung bình có 5 năm kinh nghiệm.

- **Thu nhập**: Mức thu nhập của người vay dao động từ 3 triệu đồng đến 450 triệu đồng, trong đó thu nhập bình quân là 27 triệu đồng cho thấy người vay có thu nhập tương đối cao.
- **Chi tiêu (trung bình một tháng)**: Chi tiêu tiêu dùng trong một tháng dao động từ 1 triệu đồng đến 75 triệu đồng, chi tiêu bình quân 8 triệu đồng.
- **Hóa đơn tiền điện**: Hóa đơn tiền điện hàng tháng của khách hàng dao động từ 90.000 đồng đến 20 triệu đồng. Trung bình một người sẽ phải trả 465.078 đồng tiền điện mỗi tháng.
- **Số tiền dự kiến vay**: Số tiền vay dao động từ 20 triệu đồng đến 2 tỷ đồng. Số tiền vay trung bình là 353 triệu đồng.
- **Thời hạn tài khoản vay**: Thời hạn vay dài nhất là 84 tháng và ngắn nhất là 5 tháng, thời hạn vay trung bình trong mẫu nghiên cứu là 20 tháng.
- **Số thành viên phụ thuộc trong gia đình**: Là người (con cái, cha mẹ, ông bà,...) mà người vay có trách nhiệm cấp dưỡng hàng tháng, số người phụ thuộc càng nhiều thì số tiền phải trả càng thấp nợ của khách hàng, ảnh hưởng đến khả năng trả nợ. Trung bình, khách hàng sẽ có từ 1 – 2 người phụ thuộc trong gia đình. Số lượng người thân trong gia đình của khách hàng dao động từ 0 đến 4 người.

Các lệnh sau đây sẽ vẽ sơ đồ để tìm ra các điểm ngoại lai trong bộ dữ liệu.

```
plt.figure(figsize=(22,10))
for i in enumerate(numerical_features):
    plt.subplot(2,4,i[0]+1)
    sns.boxplot(y=data_1[i[1]])
    plt.title(i[1])
    plt.ylabel("")
```



Hình 2.4: Boxplot thể hiện các dữ liệu ngoại lai cho từng thuộc tính

```
fig, ax = plt.subplots(8, 2, **{"figsize": (20, 50)})
type_graph = ['histplot', 'boxplot']
for i, graph in enumerate(type_graph):
    for j, numerical in enumerate(numerical_features):
        if graph == 'histplot':
            sns.histplot(data=data_1, x=numerical, hue='Kha nang tra no',
    ax=ax[j][i], kde=True)
        else:
            sns.boxplot(data=data_1, x='Kha nang tra no', y=numerical, ax=ax[j][i])
```

Phân tích dữ liệu phân loại

Thực hiện tương tự như với dữ liệu số, chúng em lấy ra những trường dữ liệu có các biến phân loại:

	Number of categorical values	Categorical values
Khả năng trả nợ	2	[0. Trả nợ đúng hạn, 1. Trả nợ trễ hạn]
Giới tính	2	[0. Nữ, 1. Nam]
Tình trạng hôn nhân	4	[1. Có gia đình, 2. Độc thân, 3. Ly hôn, 4. Góa, nan]
Tài sản thế chấp	2	[0. Có tài sản thế chấp, 1. Không có tài sản thế chấp]
Tình trạng sở hữu nhà ở	2	[0. Đã sở hữu nhà ở, 1. Chưa sở hữu nhà ở]
Mục đích vay	6	[1. Tiêu dùng, 4. Mua xe, 6. Đầu tư chứng khoán, 2. Học tập, 3. Sản xuất kinh doanh, 5. Mua nhà]
Chứng minh thu nhập	2	[0. Có giấy tờ chứng minh thu nhập, 1. Không có giấy tờ chứng minh thu nhập]
Trình độ học vấn	4	[2. Thạc sĩ, 3. Đại học, 1. Tiến sĩ, 4. Cấp ba, nan]

Hình 2.5: Dữ liệu phân loại

Thống kê mô tả các biến phân loại ta có:

	Giới tính	Tình trạng hôn nhân	Tài sản thế chấp	Tình trạng sở hữu nhà ở	Mục đích vay	Chứng minh thu nhập	Trình độ học vấn
count	3006	2992	3006	3006	3006	3006	2985
unique	2	4	2	2	6	2	4
top	1. Nam	1. Có gia đình	1. Không có tài sản thế chấp	1. Chưa sở hữu nhà ở	1. Tiêu dùng	0. Có giấy tờ chứng minh thu nhập	3. Đại học
freq	1589	1210	1528	1560	1026	1574	1293

Hình 2.6: Thống kê mô tả dữ liệu phân loại

Các trường của dữ liệu phân loại được hiểu như sau:

-"Giới tính": Có 2 loại, (1) là nam và (2) là nữ. Giới tính nam chiếm số lượng lớn nhất với 1589 khách hàng.

-"Tình trạng hôn nhân": Có 4 loại được trình bày trong bảng tóm tắt các biến loại ở trên. Số lượng khách hàng đã lập gia đình có nhu cầu vay vốn lớn nhất với 1210 khách hàng.

-"Tài sản thế chấp": Số lượng khách hàng cá nhân đến vay không cần thế chấp lớn nhất với 1528 khách hàng.

-"Tình trạng sở hữu nhà ở": Phần lớn khách hàng cá nhân (1560 khách hàng) đến vay tiền đều không sở hữu nhà.

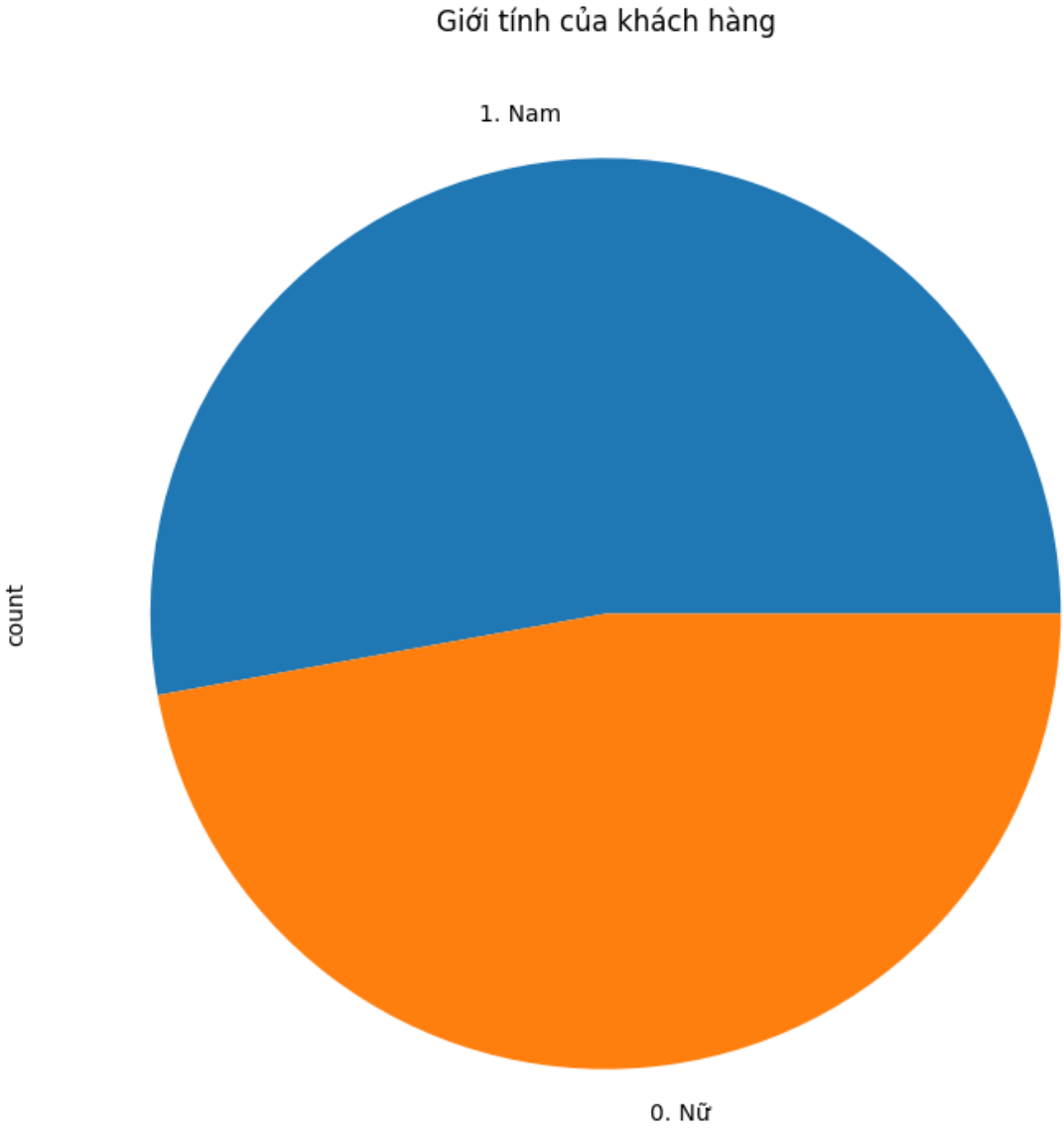
-"Mục đích vay": Có 6 loại được trình bày trong bảng tóm tắt trên. Khách hàng có nhu cầu vay tiêu dùng chiếm số lượng lớn nhất với 1026 người.

-"Chứng minh thu nhập": Có 2 loại lần lượt là (0) và (1) có và không. Hầu hết

khách hàng vay đều có đầy đủ giấy tờ chứng minh nguồn thu nhập, con số này lên tới 1574 khách hàng.

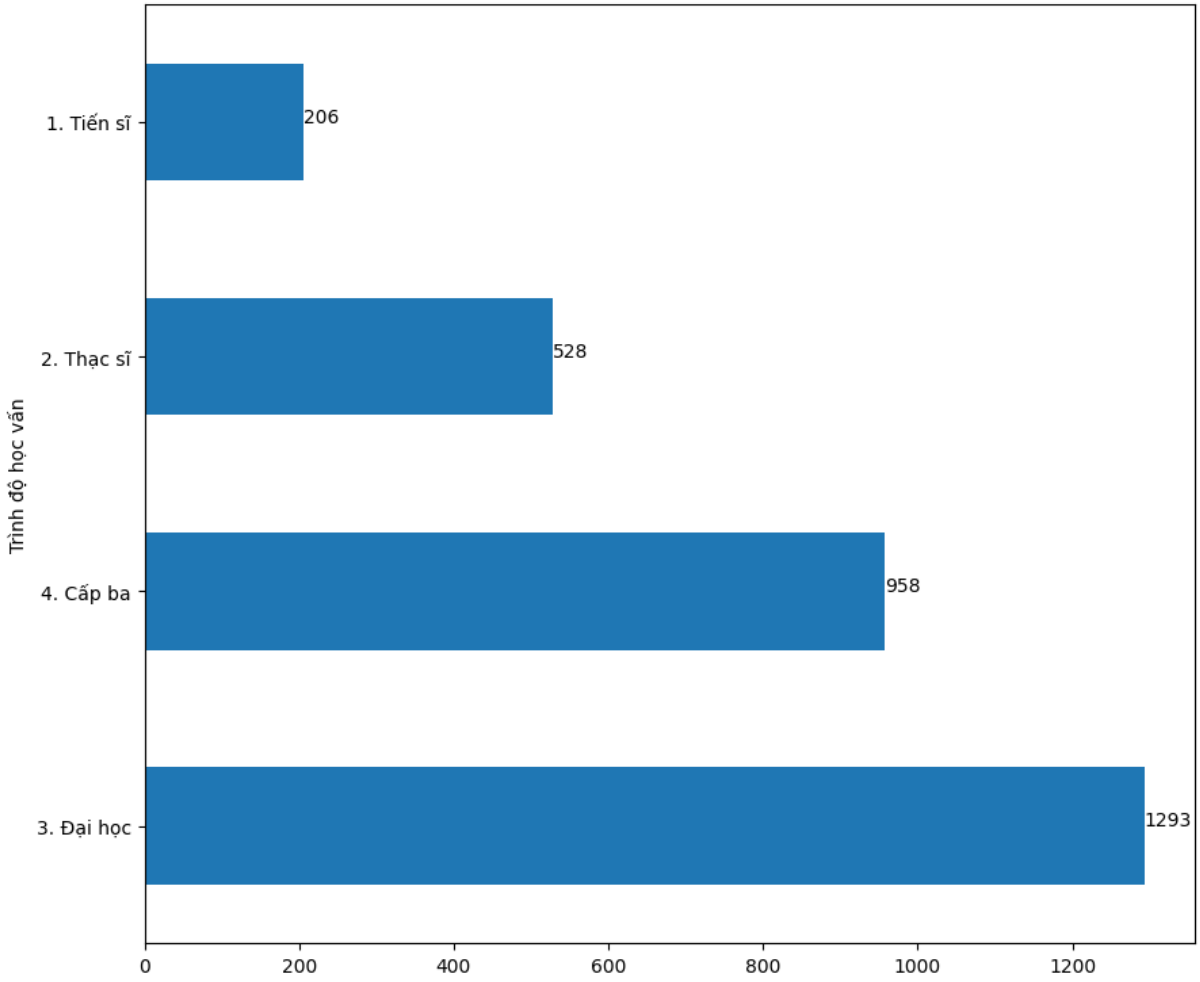
-"Trình độ học vấn": Có 4 loại. Khách hàng có trình độ đại học có nhu cầu vay vốn lớn hơn các khách hàng còn lại với 1293 khách hàng.

Tiếp tục thực hiện trực quan hóa dữ liệu phân loại ta có được những đánh giá cơ bản về cá trường như sau:



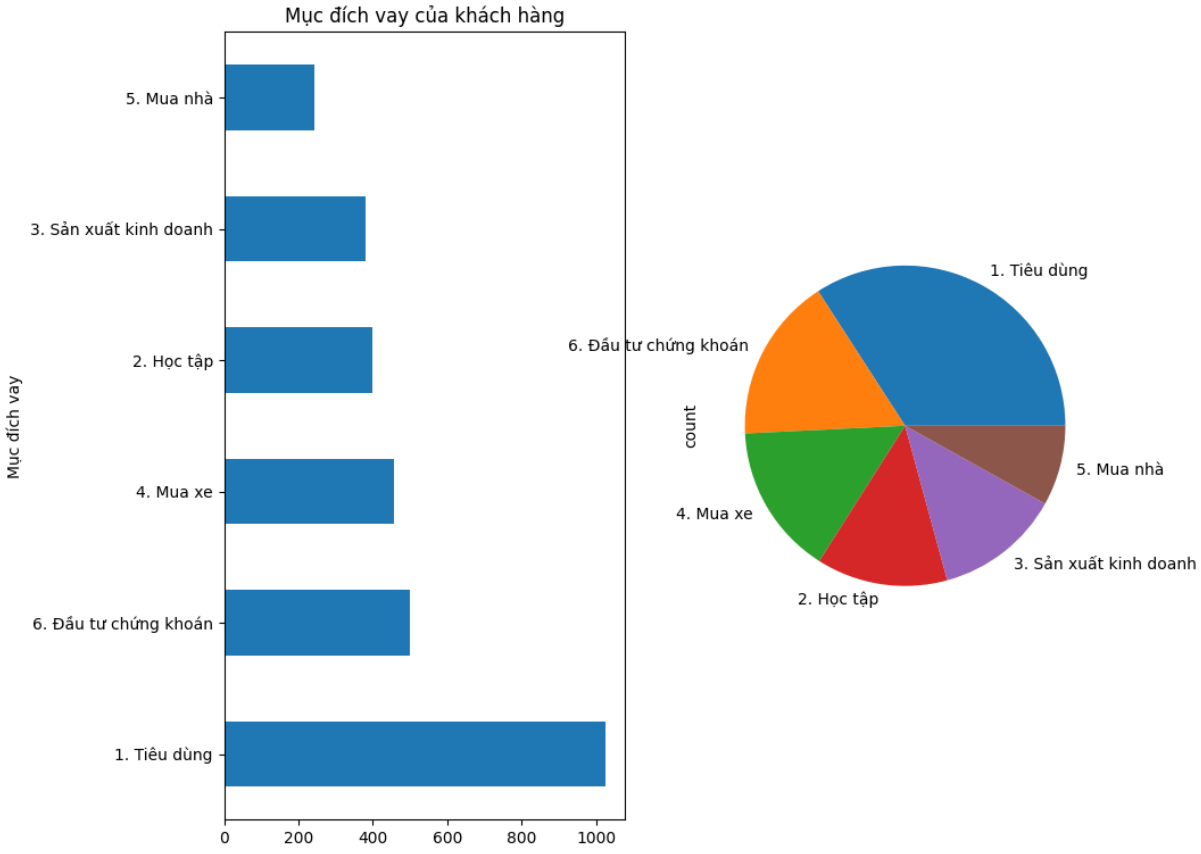
Hình 2.7: Giới tính của khách hàng

Tỷ lệ Nam vay vốn cao hơn Nữ, tuy nhiên chênh lệch không nhiều.



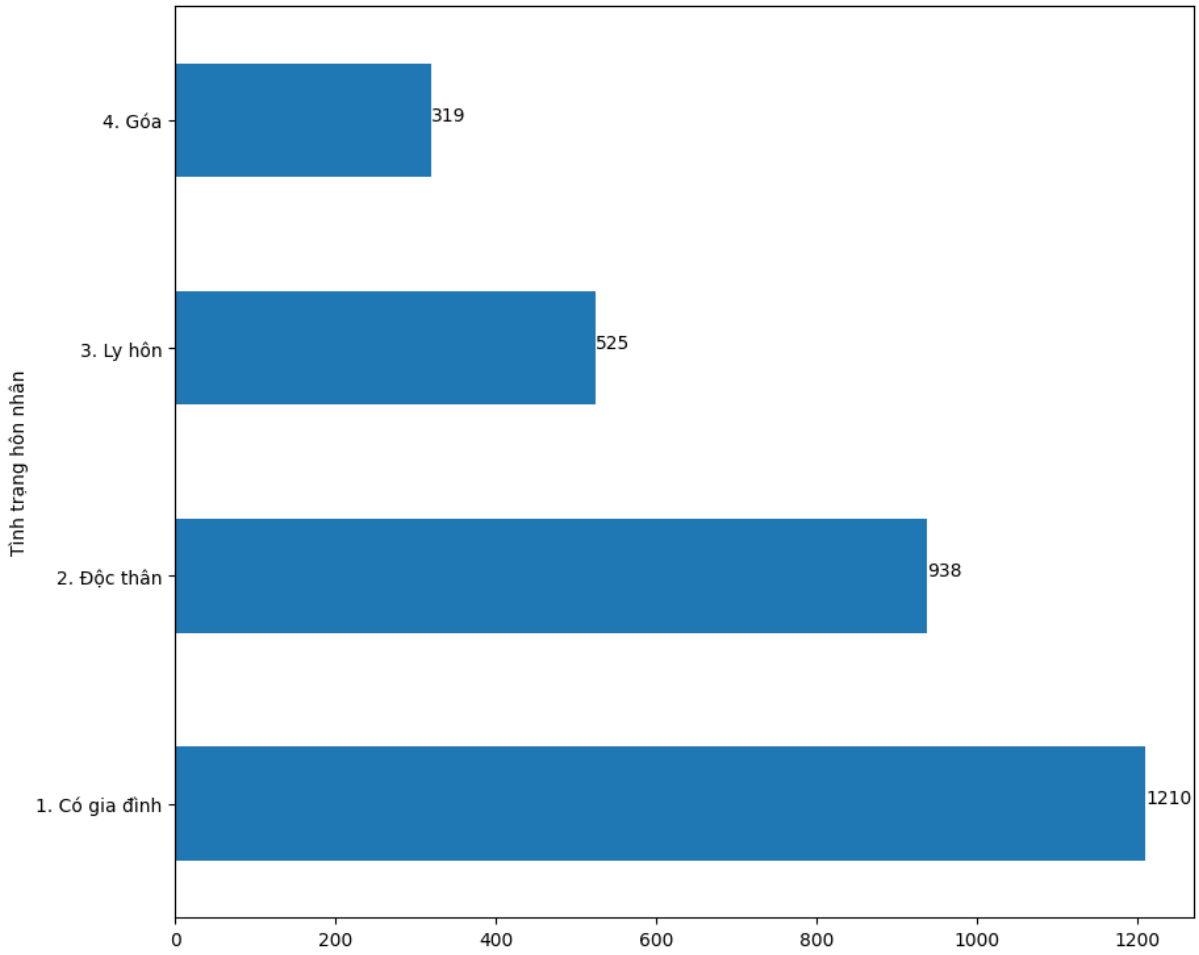
Hình 2.8: Trình độ học vấn của khách hàng

Tỷ lệ vay vốn đối với người có trình độ Đại học là cao nhất, kế đến là người có trình độ Cấp ba và những người có trình độ học vấn cao như Thạc sĩ, Tiến sĩ có tỷ lệ vay vốn tương đối thấp.



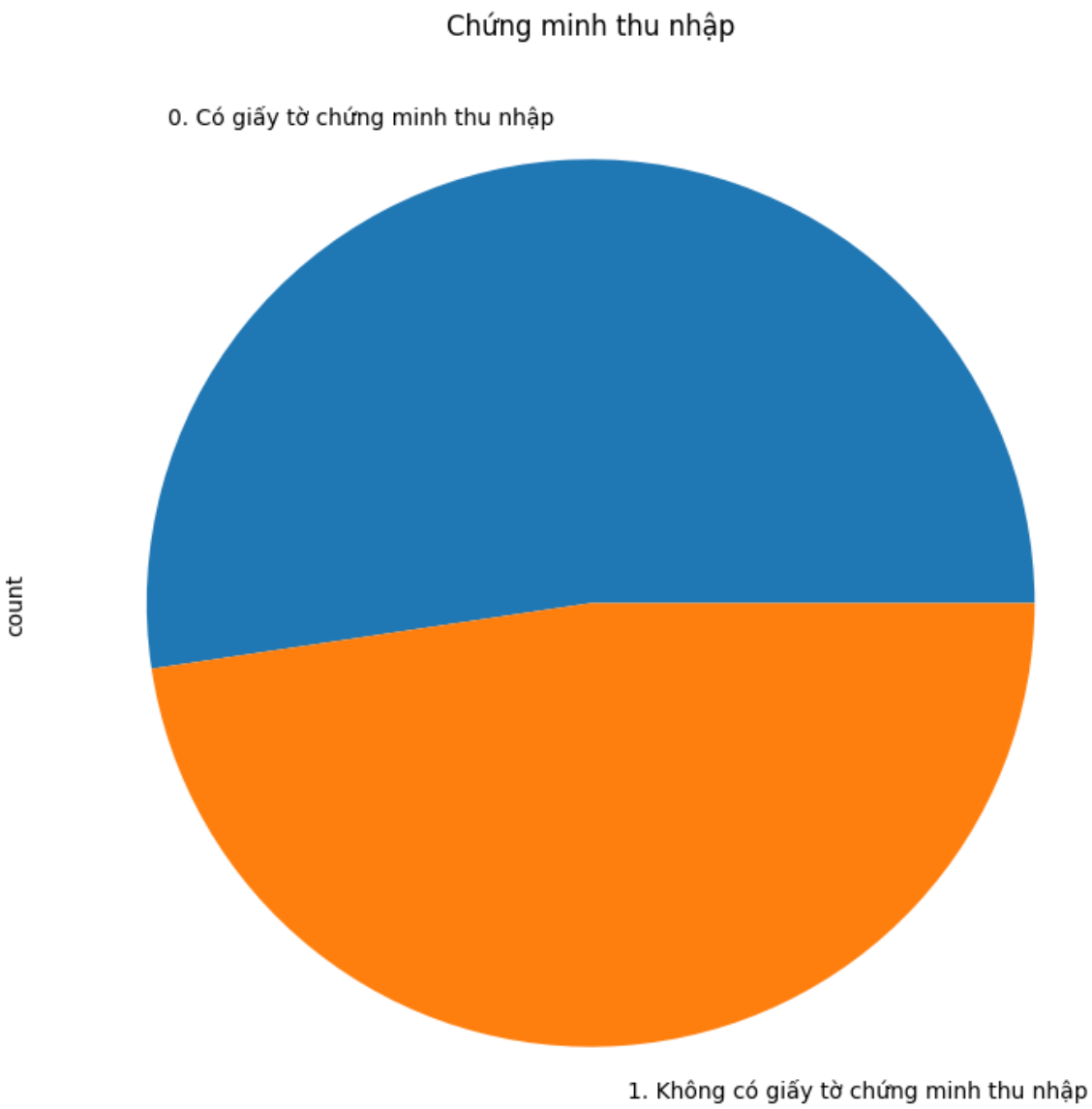
Hình 2.9: Mục đích vay của khách hàng

Đa số những người đi vay vốn với mục đích tiêu dùng, kế đến là đầu tư chứng khoán, mua xe, học tập, sản xuất kinh doanh. Những người vay vốn với mục đích mua nhà chiếm tỉ lệ thấp nhất.



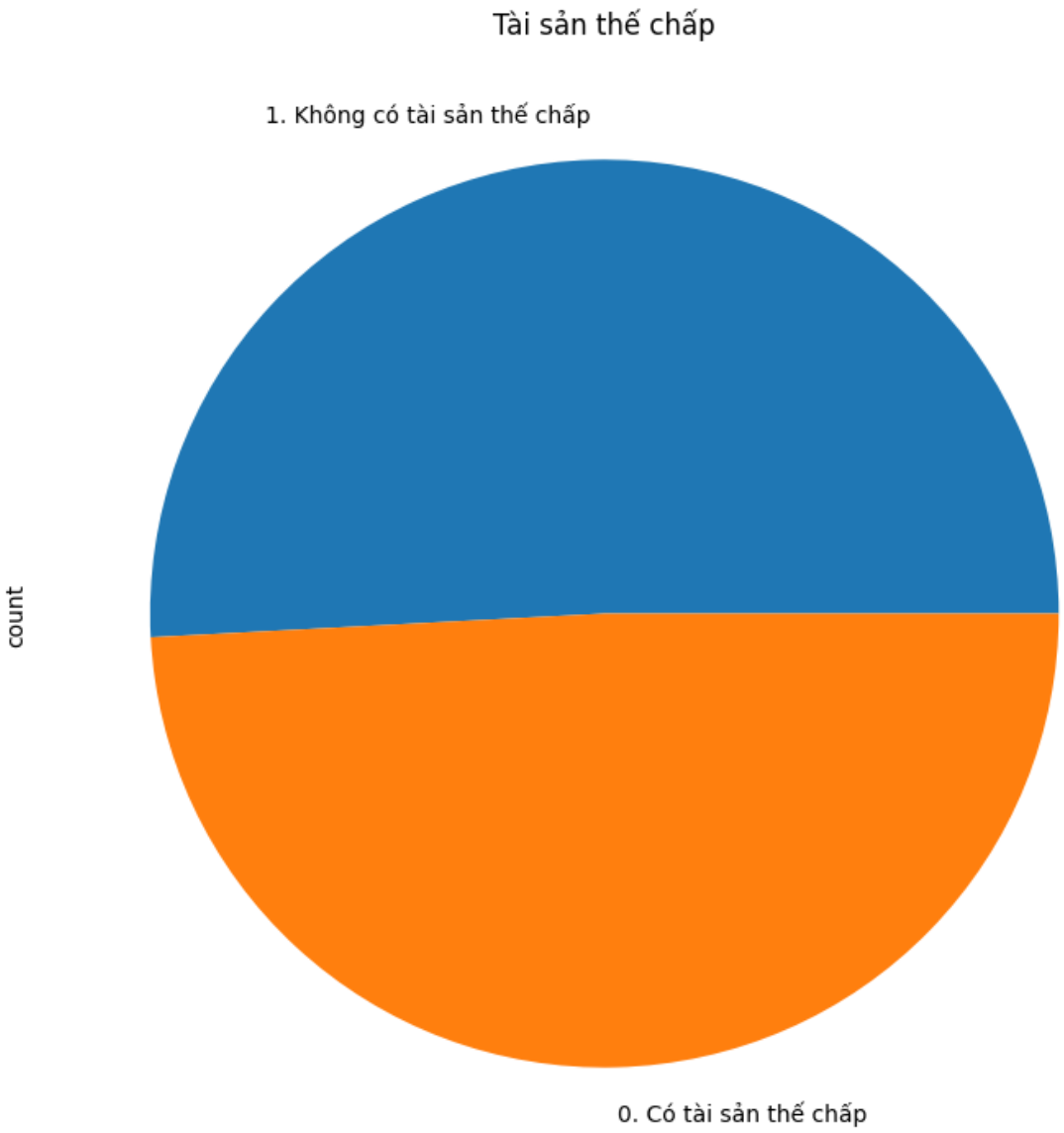
Hình 2.10: Tình trạng hôn nhân

Tỷ lệ người có gia đình vay vốn cao hơn đáng kể so với những người chưa kết hôn, ly hôn hoặc góa.



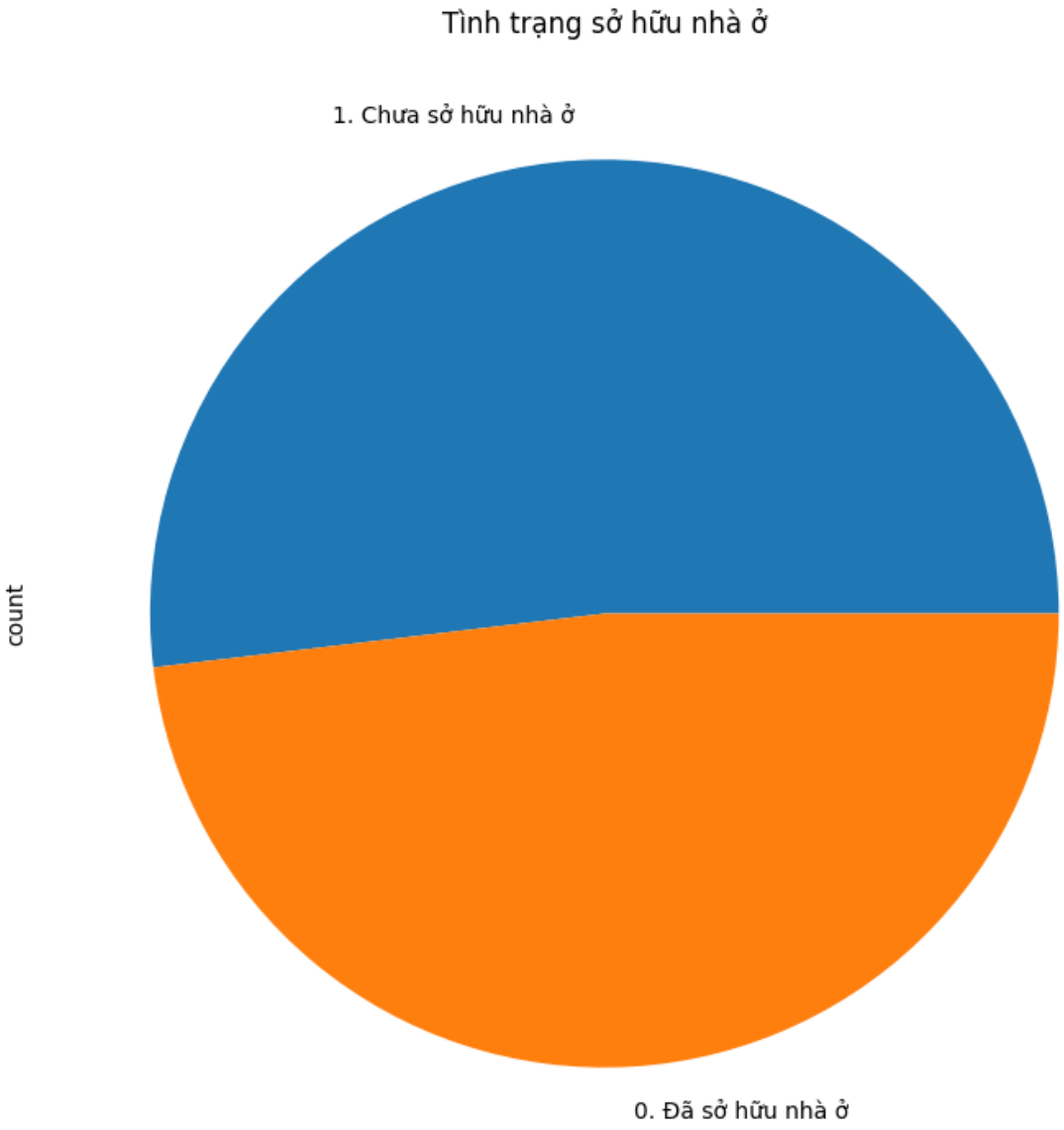
Hình 2.11: Chứng minh thu nhập

Tỷ lệ những người đi vay vốn có chứng minh thu nhập (nguồn thu nhập minh bạch) cao hơn những người không có giấy tờ chứng minh nguồn thu nhập, tuy nhiên mức độ chênh lệch không nhiều.



Hình 2.12: Tài sản thế chấp

Không có sự chênh lệch nhiều giữa tỉ lệ người đi vay có tài sản thế chấp hoặc không.



Hình 2.13: Tình trạng sở hữu nhà

Tỷ lệ khách hàng chưa sở hữu nhà ở vay vốn cao hơn nhóm khách hàng đã sở hữu nhà ở, tuy nhiên chênh lệch không đáng kể.

Qua đó ta có sự nhìn nhận sơ bộ về khả năng trả nợ của từng loại khách hàng như sau:

Hình 2.14: So sánh khả năng trả nợ

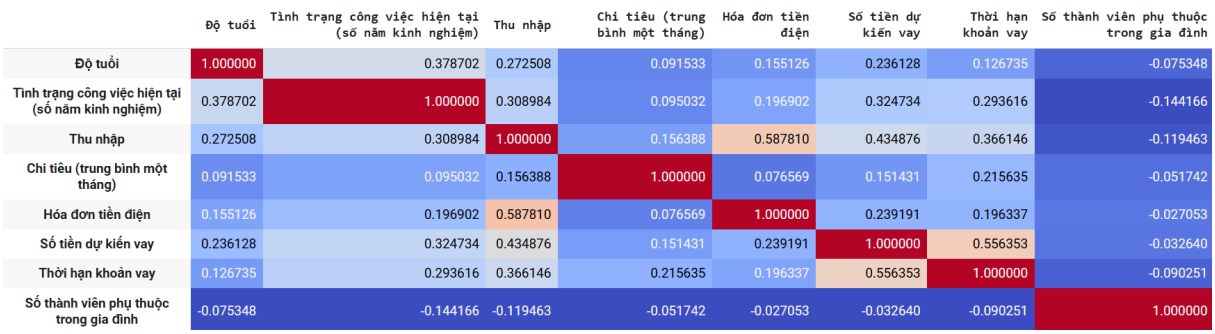
Khách hàng nam có nhu cầu vay vốn cao hơn khách hàng nữ. Khách hàng nữ có nhiều khả năng trả nợ đúng hạn hơn khách hàng nam. Khách hàng nam có tỷ lệ chậm thanh toán cao hơn so với khách hàng nữ. Nhóm khách hàng có tài sản đảm bảo có tỷ lệ trả nợ đúng hạn cao hơn đáng kể so với nhóm không có tài sản đảm bảo. Khách hàng vay với mục đích tiêu dùng sẽ có khả năng trả nợ đúng hạn và trả chậm cao hơn so với các mục đích khác (vì số lượng khách hàng vay tiêu dùng cao hơn đáng kể so với các nhóm khác). Khách hàng có trình độ đại học có khả năng trả nợ đúng hạn cao nhất và khách hàng có trình độ trung học phổ thông có xu hướng trả nợ muộn nhất. Nhóm khách hàng đã lập gia đình có tỷ lệ trả nợ đúng hạn cao nhất và nhóm khách hàng độc thân có khả năng trả nợ chậm cao nhất. Nhóm khách hàng đã có nhà có tỷ lệ trả nợ đúng hạn cao hơn nhóm khách hàng chưa có nhà và khách hàng chưa có nhà có tỷ lệ trả nợ chậm tương đối cao. Nhóm khách hàng có đầy đủ giấy tờ chứng minh nguồn thu nhập có tỷ lệ trả nợ đúng hạn cao hơn nhóm khách hàng không có giấy tờ chứng minh nguồn thu nhập.

Kiểm tra sự mất cân bằng

Thực hiện thống kê, ta thấy được có 1527 trường hợp trả nợ trễ hạn và 1479 trường hợp trả nợ đúng hạn. Vậy có thể kết luận, không có sự mất cân bằng trong bộ dữ liệu.

Sự tương quan của dữ liệu

Ma trận tương quan giúp ta có sự đánh giá cơ bản về sự tương quan của bộ dữ liệu:



	Độ tuổi	Tình trạng công việc hiện tại (số năm kinh nghiệm)	Thu nhập	Chi tiêu (trung bình một tháng)	Hóa đơn tiền điện	Số tiền dự kiến vay	Thời hạn khoản vay	Số thành viên phụ thuộc trong gia đình
Độ tuổi	1.000000	0.378702	0.272508	0.091533	0.155126	0.236128	0.126735	-0.075348
Tình trạng công việc hiện tại (số năm kinh nghiệm)	0.378702	1.000000	0.308984	0.095032	0.196902	0.324734	0.293616	-0.144166
Thu nhập	0.272508	0.308984	1.000000	0.156388	0.587810	0.434876	0.366146	-0.119463
Chi tiêu (trung bình một tháng)	0.091533	0.095032	0.156388	1.000000	0.076569	0.151431	0.215635	-0.051742
Hóa đơn tiền điện	0.155126	0.196902	0.587810	0.076569	1.000000	0.239191	0.196337	-0.027053
Số tiền dự kiến vay	0.236128	0.324734	0.434876	0.151431	0.239191	1.000000	0.556353	-0.032640
Thời hạn khoản vay	0.126735	0.293616	0.366146	0.215635	0.196337	0.556353	1.000000	-0.090251
Số thành viên phụ thuộc trong gia đình	-0.075348	-0.144166	-0.119463	-0.051742	-0.027053	-0.032640	-0.090251	1.000000

Hình 2.15: Ma trận tương quan

2.2 Tiền xử lý dữ liệu

2.2.1 Làm sạch dữ liệu

Để dữ liệu có thể sẵn sàng được xử lí bởi các mô hình máy học, các dữ liệu còn thiếu trước hết phải được bổ sung.

Có tất cả bốn thuộc tính có dữ liệu còn thiếu, và số lượng cùng với tỉ lệ là như sau:

Thuộc tính	**Số lượng**	**Tỉ lệ phần trăm**
Trình độ học vấn	21	0.006986
Tình trạng công việc hiện tại (số năm kinh nghiệm)	16	0.005323
Tình trạng hôn nhân	14	0.004657
Hoá đơn tiền điện	14	0.004657

Bảng 2.2: Các thuộc tính còn thiếu

Các giá trị dữ liệu trên sẽ được bổ sung như sau:

- Với các dữ liệu định tính, thay thế các giá trị còn thiếu bằng với mode,

- Với các dữ liệu định lượng, thay thế các giá trị còn thiếu bằng với trung vị.

Tiếp đến, cần loại bỏ các dữ liệu trùng lặp.

Có tất cả bảy mục dữ liệu bị trùng. Loại bỏ tất cả chúng cho ta một bộ dữ liệu với 2999 mục.

2.2.2 Chuyển đổi dữ liệu

Dữ liệu sau khi làm sạch cần được chuyển đổi sang dạng số để có thể xử lí bởi các mô hình máy học. Ngoài ra, tên các thuộc tính cũng nên được chuyển đổi thành tiếng Việt không dấu.

Những dữ liệu phân loại sẽ được gắn nhãn số để các mô hình học máy có thể học và dự đoán được. Các nhãn sẽ được gắn như sau:

	Number of categorical values	Categorical values
Khả năng trả nợ	2	[0. Trả nợ đúng hạn, 1. Trả nợ trễ hạn]
Giới tính	2	[0. Nữ, 1. Nam]
Tình trạng hôn nhân	4	[1. Có gia đình, 2. Độc thân, 3. Ly hôn, 4. Góa, nan]
Tài sản thế chấp	2	[0. Có tài sản thế chấp, 1. Không có tài sản thế chấp]
Tình trạng sở hữu nhà ở	2	[0. Đã sở hữu nhà ở, 1. Chưa sở hữu nhà ở]
Mục đích vay	6	[1. Tiêu dùng, 4. Mua xe, 6. Đầu tư chứng khoán, 2. Học tập, 3. Sản xuất kinh doanh, 5. Mua nhà]
Chứng minh thu nhập	2	[0. Có giấy tờ chứng minh thu nhập, 1. Không có giấy tờ chứng minh thu nhập]
Trình độ học vấn	4	[2. Thạc sĩ, 3. Đại học, 1. Tiến sĩ, 4. Cấp ba, nan]

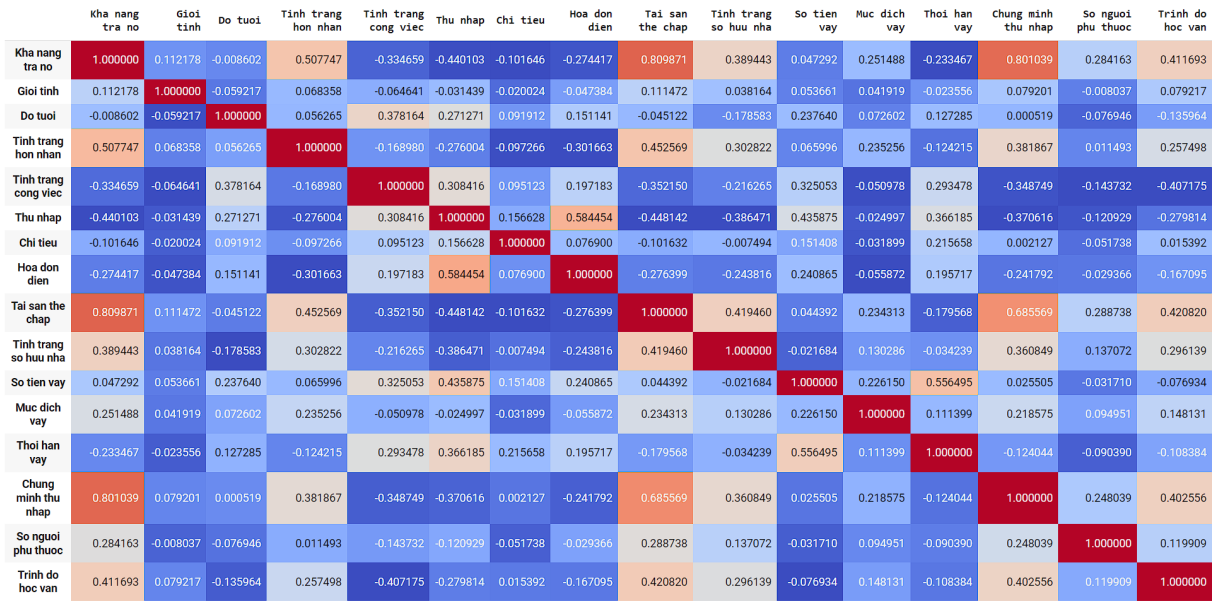
Hình 2.16: Các nhãn của dữ liệu tương ứng

Cuối cùng ta thu được bộ dữ liệu sau khi chuyển đổi:

	Khanangtrano	Gioitinh	Dotuoi	Tinhtranghonnhan	Tinhtrangcongviec	Thunhap	Chitieu	Hoadondien	Taisanthechap	Tinhtrangsohuunha	Sotienvay	Mucdichvay	Thoihanvay	Chungminhthunhap	Songuoiphuthuoc	Trinhdohocvan
0	0	0	37	1	7	30000000	15000000	600000	0	0	250000000	1	12	0	1	2
1	0	0	41	1	8	35000000	19000000	650000	0	0	250000000	1	12	0	2	3
2	0	1	33	1	6	35000000	20000000	600000	0	0	350000000	1	12	0	2	2
3	0	1	36	1	7	35000000	4500000	750000	0	0	300000000	1	12	0	2	1
4	0	1	35	1	5	35000000	4500000	650000	0	0	350000000	1	8	0	3	1

Hình 2.17: Dữ liệu sau khi chuyển đổi

2.2.3 Tương quan giữa các thuộc tính



	Kha nang tra no	Gioi tinh	Do tuoi	Tinh trang hon nhan	Tinh trang cong viec	Thu nhap	Chi tieu	Hoa don dien	Tai san the chap	Tinh trang so huu nha	So tien vay	Muc dich vay	Thoi han vay	Chung minh thu nhap	So nguoi phu thuoc	Trinh do hoc van
Kha nang tra no	1.000000	0.112178	-0.008602	0.507747	-0.334659	-0.440103	-0.101646	-0.274417	0.809871	0.389443	0.047292	0.251488	-0.233467	0.801039	0.284163	0.411693
Gioi tinh	0.112178	1.000000	-0.059217	0.068358	-0.064641	-0.031439	-0.020024	-0.047384	0.111472	0.038164	0.053661	0.041919	-0.023556	0.079201	-0.008037	0.079217
Do tuoi	-0.008602	-0.059217	1.000000	0.056265	0.378164	0.271271	0.091912	0.151141	-0.045122	-0.178583	0.237640	0.072602	0.127285	0.000519	-0.076946	-0.135964
Tinh trang hon nhan	0.507747	0.068358	0.056265	1.000000	-0.168980	-0.276004	-0.097266	-0.301663	0.452569	0.302822	0.065996	0.235256	-0.124215	0.381867	0.011493	0.257498
Tinh trang cong viec	-0.334659	-0.064641	0.378164	-0.168980	1.000000	0.308416	0.095123	0.197183	-0.352150	-0.216265	0.325053	-0.050978	0.293478	-0.348749	-0.143732	-0.407175
Thu nhap	-0.440103	-0.031439	0.271271	-0.276004	0.308416	1.000000	0.156628	0.584454	-0.448142	-0.386471	0.435875	-0.024997	0.366185	-0.370616	-0.120929	-0.279814
Chi tieu	-0.101646	-0.020024	0.091912	-0.097266	0.095123	0.156628	1.000000	0.076900	-0.101632	-0.007494	0.151408	-0.031899	0.215658	0.002127	-0.051738	0.015392
Hoa don dien	-0.274417	-0.047384	0.151141	-0.301663	0.197183	0.584454	0.076900	1.000000	-0.276399	-0.243816	0.240865	-0.055872	0.195717	-0.241792	-0.029366	-0.167095
Tai san the chap	0.809871	0.111472	-0.045122	0.452569	-0.352150	-0.448142	-0.101632	-0.276399	1.000000	0.419460	0.044392	0.234313	-0.179568	0.685569	0.288738	0.420820
Tinh trang so huu nha	0.389443	0.038164	-0.178583	0.302822	-0.216265	-0.386471	-0.007494	-0.243816	0.419460	1.000000	-0.021684	0.130286	-0.034239	0.360849	0.137072	0.296139
So tien vay	0.047292	0.053661	0.237640	0.065996	0.325053	0.435875	0.151408	0.240865	0.044392	-0.021684	1.000000	0.226150	0.556495	0.025505	-0.031710	-0.076934
Muc dich vay	0.251488	0.041919	0.072602	0.235256	-0.050978	-0.024997	-0.031899	-0.055872	0.234313	0.130286	0.226150	1.000000	0.111399	0.218575	0.094951	0.148131
Thoi han vay	-0.233467	-0.023556	0.127285	-0.124215	0.293478	0.366185	0.215658	0.195717	-0.179568	-0.034239	0.556495	0.111399	1.000000	-0.124044	-0.090390	-0.108384
Chung minh thu nhap	0.801039	0.079201	0.000519	0.381867	-0.348749	-0.370616	0.002127	-0.241792	0.685569	0.360849	0.025505	0.218575	-0.124044	1.000000	0.248039	0.402556
So nguoi phu thuoc	0.284163	-0.008037	-0.076946	0.011493	-0.143732	-0.120929	-0.051738	-0.029366	0.288738	0.137072	-0.031710	0.094951	-0.090390	0.248039	1.000000	0.119909
Trinh do hoc van	0.411693	0.079217	-0.135964	0.257498	-0.407175	-0.279814	0.015392	-0.167095	0.420820	0.296139	-0.076934	0.148131	-0.108384	0.402556	0.119909	1.000000

Hình 2.18: Ma trận tương quan giữa các thuộc tính

Các giá trị tương quan này nằm ở mức chấp nhận được, và hợp lý trong bối cảnh kinh tế tài chính.

	Kha nang tra no
Gioi tinh	0.112178
Do tuoi	-0.008602
Tinh trang hon nhan	0.507747
Tinh trang cong viec	-0.334659
Thu nhap	-0.440103
Chi tieu	-0.101646
Hoa don dien	-0.274417
Tai san the chap	0.809871
Tinh trang so huu nha	0.389443
So tien vay	0.047292
Muc dich vay	0.251488
Thoi han vay	-0.233467

Chung minh thu nhap	0.801039
So nguoi phu thuoc	0.284163
Trinh do hoc van	0.411693

Bảng 2.3: Tương quan giữa các biến độc lập với biến mục tiêu

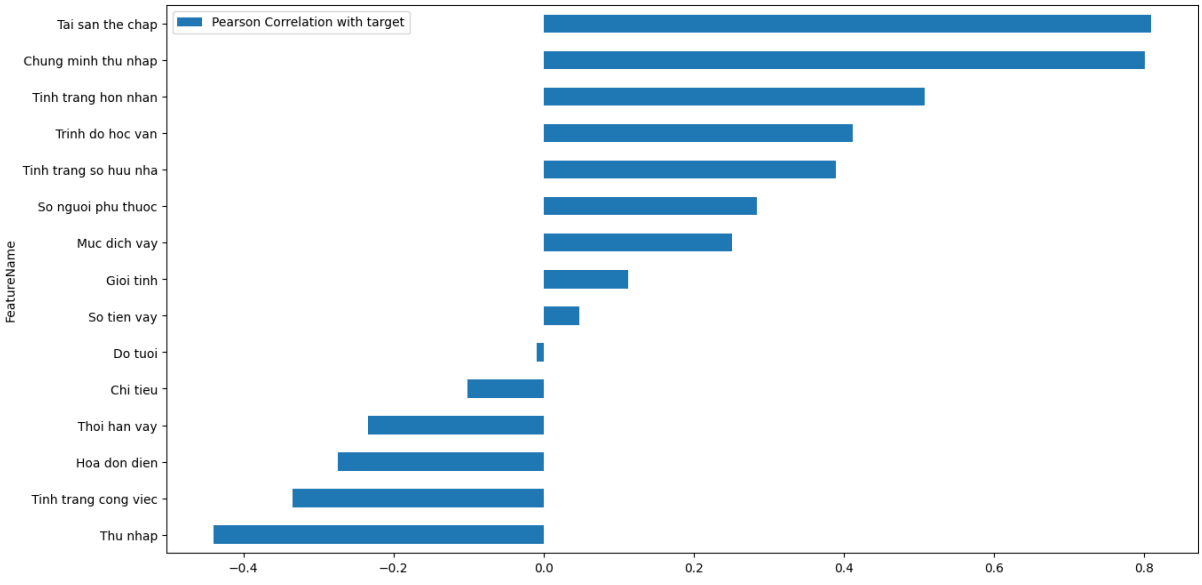
Tương quan giữa các biến độc lập với biến mục tiêu theo sát với những giả định ban đầu và xu hướng đem lại ảnh hưởng tích cực/tiêu cực tới biến mục tiêu.

Chương 3

Xây dựng mô hình và kết quả thực nghiệm

3.1 Lựa chọn thuộc tính

Quay lại hệ số tương quan của từng biến độc lập tới mục tiêu; ta sắp xếp các biến theo chiều giảm dần hệ số tương quan, và vẽ biểu đồ minh hoạ như sau:



Hình 3.1: Biểu đồ minh hoạ hệ số tương quan của từng biến độc lập tới biến mục tiêu

Qua biểu đồ, ta có thể đưa ra các nhận xét sau:

- Các biến "Thu nhap", "Tinh trang cong viec", "Hoa don dien", "Thoi han

vay", "Chi tieu", "Do tuoi" có tương quan âm;

- Các biến “Tai san the chap”, “Chung minh thu nhap”, “Tinh trang hon nhan”, “Trinh do hoc van”, “Tinh trang so huu nha”, “So nguoi phu thuoc”, “Muc dich vay”, “Gioi tinh”, “So tien vay” có tương quan dương.

Trong 15 biến được mô tả trên biểu đồ, ta chọn 5 biến có trị tuyệt đối của độ tương quan lớn nhất (biểu thị cho ảnh hưởng đến kết quả mô hình lớn nhất) để chạy và tối ưu mô hình trên đó; các biến khác khi được sử dụng trong mô hình có thể không đem lại thay đổi đáng kể nào đến kết quả và hiệu năng của mô hình.

```
from sklearn.metrics import roc_curve, auc
def _plot_roc_curve(fpr, tpr, thres, auc):
    plt.figure(figsize = (10, 8))
    plt.plot(fpr, tpr, 'b-', color='darkorange', lw=2, linestyle='--', label='ROC
    curve (area = %0.2f)'%auc)
    plt.plot([0, 1], [0, 1], '--')
    plt.axis([0, 1, 0, 1])
    plt.xlabel('False Positive Rate', color="r")
    plt.ylabel('True Positive Rate', color="g")
    plt.legend(loc='lower right')
    plt.title('ROC Curve')
```

```
target = ['Kha nang tra no']
features = ['Tai san the chap','Chung minh thu nhap','Tinh trang hon nhan','Thu
    nhap','Trinh do hoc van']
```

```
X = df[features]
y = df[target]
```

3.2 Tách dữ liệu huấn luyện

```
# train, test split
from sklearn.model_selection import train_test_split
n_state = 42
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3,
    random_state = n_state)
```

```
X_train.shape, X_test.shape
```

Sau khi chạy, dữ liệu được tách làm 2 phần: `X_train` (2099 dòng), `X_test` (900 dòng).

```
print(np.count_nonzero(y_train == 1))
print(len(y_train) - np.count_nonzero(y_train == 1))
```

Sau khi chạy các đoạn code trên, số mẫu dữ liệu trong class 0 là 1033 (49%), trong khi số mẫu trong class 1 là 1066 (51%). Chênh lệch không quá lớn, bộ dữ liệu không bị mất cân bằng.

3.3 Chuẩn hoá dữ liệu

```
# Using StandardScaler
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()

# Scale train and test set
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.fit_transform(X_test)
```

```
from sklearn import metrics
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score,
    roc_auc_score
from sklearn.metrics import roc_curve, auc
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.metrics import ConfusionMatrixDisplay
```

3.4 Lựa chọn loại mô hình

Ta đưa các thuộc tính quan trọng vào một số mô hình phân loại.

Các thuộc tính chính sử dụng để đánh giá mô hình bao gồm:

- Accuracy;
- Precision;
- Recall;
- F1-score.

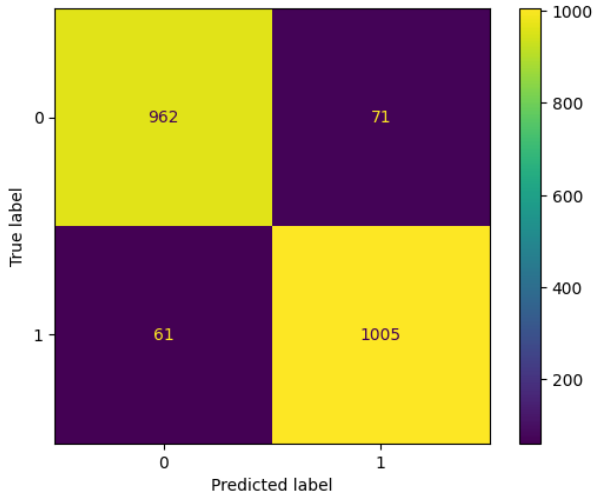
3.4.1 Hồi quy logistics

```
LR_classifier = LogisticRegression(multi_class='multinomial',random_state=n_state)

LR_classifier.fit(X_train, y_train.values.ravel())

y_pred = LR_classifier.predict(X_train)

print('Confusion matrix:')
print(pd.DataFrame(confusion_matrix(y_train,y_pred)),'\n')

print('Classification report:')
print(classification_report(y_train,y_pred))

print('Logistic Regression accuracy: ', round(accuracy_score(y_train, y_pred),4))
```

```
Confusion matrix:
     0     1
0  962    71
1   61  1005

Classification report:
              precision    recall  f1-score   support

           0       0.94      0.93      0.94      1033
           1       0.93      0.94      0.94      1066

    accuracy                           0.94      2099
   macro avg       0.94      0.94      0.94      2099
weighted avg       0.94      0.94      0.94      2099

Logistic Regression accuracy:  0.9371
```

Hình 3.2: Kết quả khi chạy trên X_train

Tiếp theo, trực quan hoá ma trận nhầm lẫn cho tập dữ liệu `X_train`:

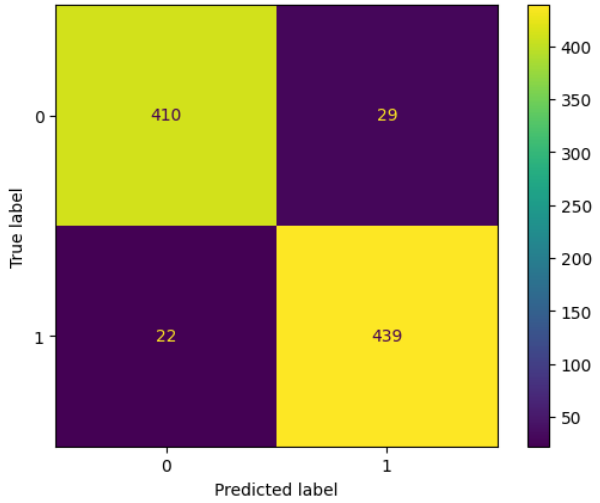


Hình 3.3: Trực quan hoá ma trận nhầm lẫn ở tập dữ liệu X_train

Sau đó thực hiện tương tự với `X_test`.

```
Confusion matrix:
     0    1
0  410   29
1   22  439

Classification report:
              precision    recall  f1-score   support

           0       0.95      0.93      0.94       439
           1       0.94      0.95      0.95       461

    accuracy                           0.94       900
   macro avg       0.94      0.94      0.94       900
weighted avg       0.94      0.94      0.94       900

Logistic Regression accuracy:  0.9433
```

Hình 3.4: Kết quả chạy mô hình trên X_test

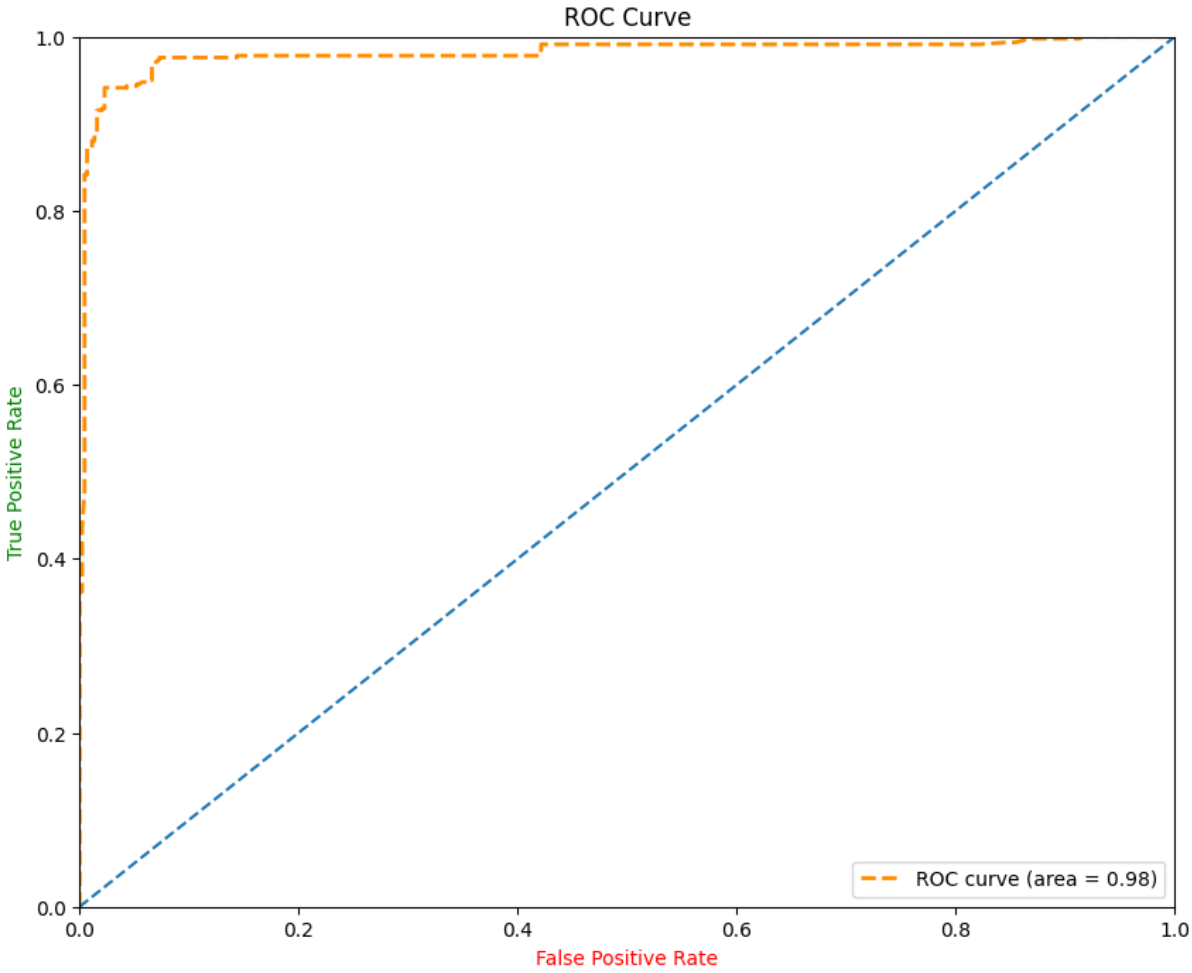


Hình 3.5: Ma trận nhầm lẫn đối với X_test

Để đánh giá ảnh hưởng của các biến độc lập đến biến mục tiêu, ta tiến hành tính hệ số tương quan giữa từng biến độc lập đến biến mục tiêu, và thu được kết quả như sau:

Biến độc lập	**Hệ số tương quan**
Chung minh thu nhap	0.932668
Tai san the chap	0.820317
Tinh trang hon nhan	0.487112
Thu nhap	-0.122048
Trinh do hoc van	0.016964

Bảng 3.1: Hệ số tương quan trong mô hình hồi quy logistics



Hình 3.6: Đường cong ROC trong mô hình hồi quy logistics

Sau khi chạy mô hình, ta có thể thu về tổng quan quá trình chạy mô hình.

```
Optimization terminated successfully.
         Current function value: 0.174795
         Iterations 8
                           Logit Regression Results
==============================================================================
Dep. Variable:        Kha nang tra no   No. Observations:                 2099
Model:                          Logit   Df Residuals:                     2094
Method:                           MLE   Df Model:                            4
Date:                Sun, 23 Jun 2024   Pseudo R-squ.:                  0.7478
Time:                        03:59:39   Log-Likelihood:                -366.89
converged:                       True   LL-Null:                       -1454.7
Covariance Type:            nonrobust   LLR p-value:                     0.000
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
x1             1.6458      0.108     15.241      0.000       1.434       1.857
x2             1.8558      0.108     17.238      0.000       1.645       2.067
x3             0.9889      0.113      8.784      0.000       0.768       1.210
x4            -0.2620      0.126     -2.075      0.038      -0.509      -0.015
x5             0.0455      0.108      0.422      0.673      -0.166       0.257
==============================================================================
```

Hình 3.7: Tổng quan mô hình hồi quy logistics

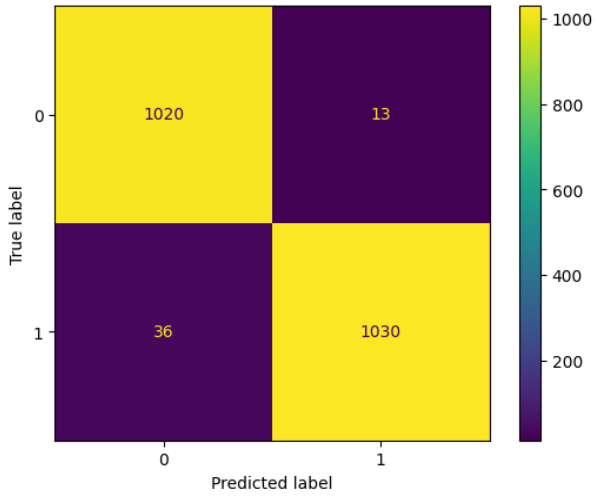
Nhận xét: Ba biến độc lập "Tai san the chap" (x1), "Chung minh thu nhap" (x2), "Tinh trang hon nhan" (x3) có ý nghĩa về thống kê ở mức độ tin cậy 1% và 5%, có ảnh hưởng tích cực đến biến mục tiêu. Biến mục tiêu "Thu nhap" (x4) có ý nghĩa về thống kê ở mức độ tin cậy 5%, có ảnh hưởng tiêu cực tới biến mục tiêu. "Trinh do hoc van" (x5) không có ý nghĩa thống kê ở bất kì mức độ tin cậy nào. Hệ số tương quan Pseudo R-squared là 0.7478, hay 74.78%, nghĩa là mô hình phù hợp để dự đoán khả năng thanh toán của khách hàng trong tập dữ liệu.

3.4.2 Cây quyết định

```
DT_classifier = DecisionTreeClassifier()

DT_classifier.fit(X_train, y_train.values.ravel())

y_pred = DT_classifier.predict(X_train)

print('Confusion matrix:')
print(pd.DataFrame(confusion_matrix(y_train,y_pred)),'\n')

print('Classification report:')
print(classification_report(y_train,y_pred))

print('Decision Tree accuracy: ', accuracy_score(y_train, y_pred))
```

```
Confusion matrix:
      0     1
0  1020    13
1    36  1030

Classification report:
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1033
           1       0.99      0.97      0.98      1066

    accuracy                           0.98      2099
   macro avg       0.98      0.98      0.98      2099
weighted avg       0.98      0.98      0.98      2099

Decision Tree accuracy:  0.9766555502620295
```

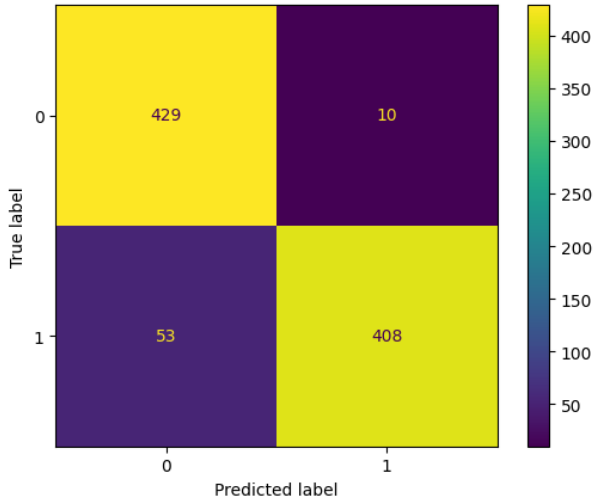
Hình 3.8: Kết quả mô hình cây quyết định trên X_train



Hình 3.9: Ma trận nhầm lẫn đối với X_train

```
Confusion matrix:
     0    1
0  429   10
1   53  408

Classification report:
              precision    recall  f1-score   support

           0       0.89      0.98      0.93       439
           1       0.98      0.89      0.93       461

    accuracy                           0.93       900
   macro avg       0.93      0.93      0.93       900
weighted avg       0.93      0.93      0.93       900

Decision Tree accuracy:  0.93
```

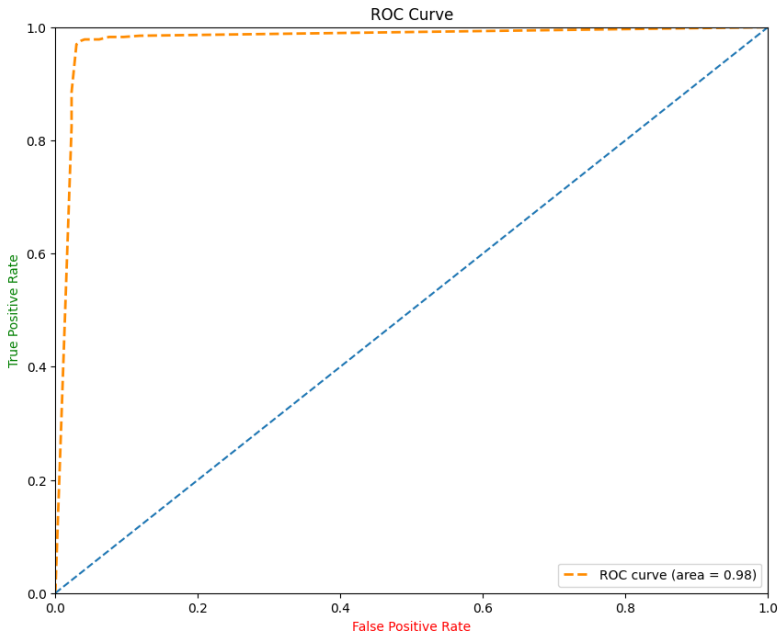
Hình 3.10: Kết quả mô hình cây quyết định trên X_test



Hình 3.11: Ma trận nhầm lẫn đối với X_test

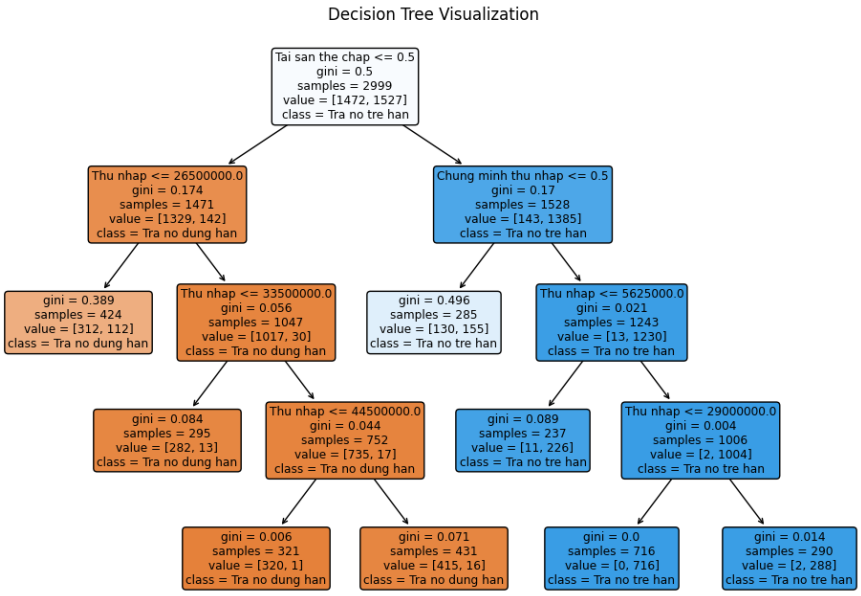
Biến độc lập	**Hệ số tương quan**
Chung minh thu nhap	0.117903
Tai san the chap	0.702779
Tinh trang hon nhan	0.047293
Thu nhap	0.115934
Trinh do hoc van	0.016090

Bảng 3.2: Hệ số tương quan trong mô hình cây quyết định



Hình 3.12: Đường cong ROC trong cây quyết định

Ta có thể trực quan hoá cây quyết định và sử dụng thuật toán gọt cây để cây gọn gàng hơn, giúp ta có cái nhìn trực quan về cách đưa ra quyết định cho các nhân viên tài chính.



Hình 3.13: Cây quyết định đã tối ưu

Theo đó, ta có thể rút ra quy luật sau để hỗ trợ các nhân viên tài chính ra quyết định:

- Đối với khách hàng có tài sản thế chấp và thu nhập hàng tháng lớn hơn 26.5 triệu VNĐ, nên ngay lập tức xét duyệt đề nghị vay.
- Đối với khách hàng không có tài sản thế chấp và chứng minh thu nhập, có thu nhập hàng tháng từ 5.625 đến 29 triệu VNĐ, nên ngay lập tức từ chối đề nghị vay.
- Đối với trường hợp khách hàng không có tài sản thế chấp nhưng có chứng minh thu nhập, nên cẩn thận xem xét đề nghị vay của khách hàng để đưa ra quyết định.

Các siêu tham số có thể được tối ưu với thuật toán GridSearchCV:

```
Best Decision tree Parameters: {'criterion': 'gini', 'max_depth': 8}
Cross-validation accuracy: 0.959
Accuracy: 0.931
Precision: 0.972
Recall: 0.892
F1 score: 0.930
```

Hình 3.14: Kết quả chạy mô hình trên X_test sau khi tối ưu siêu tham số

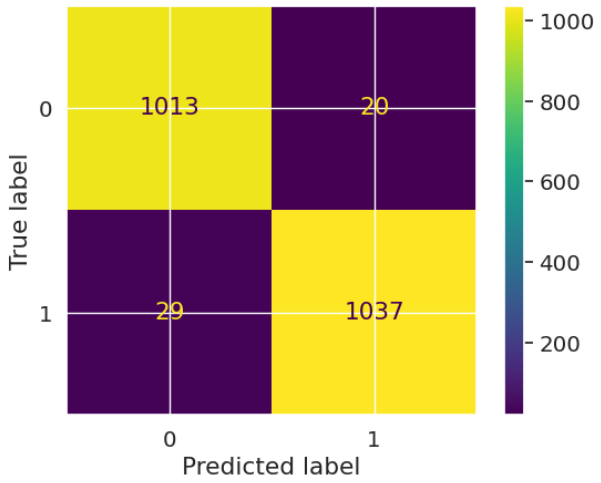
So sánh với mô hình trước khi tối ưu, các chỉ số Accuracy, Precision, Recall, F1-score đều cao hơn, như vậy hiệu quả mô hình đã được cải thiện.

3.4.3 Rừng ngẫu nhiên

```
1  from sklearn.ensemble import RandomForestClassifier
2
3  RF_classifier = RandomForestClassifier()
4
5  RF_classifier.fit(X_train, y_train.values.ravel())
6
7  y_pred = RF_classifier.predict(X_train)
8
9  print('Confusion matrix:')
10 print(pd.DataFrame(confusion_matrix(y_train,y_pred)),'\n')
11
12 print('Classification report:')
13 print(classification_report(y_train,y_pred))
14
15 print('Random Forest accuracy: ', accuracy_score(y_train, y_pred))
```

```
Confusion matrix:
      0     1
0  1013    20
1    29  1037

Classification report:
              precision    recall  f1-score   support

           0       0.97      0.98      0.98      1033
           1       0.98      0.97      0.98      1066

    accuracy                           0.98      2099
   macro avg       0.98      0.98      0.98      2099
weighted avg       0.98      0.98      0.98      2099

Random Forest accuracy:  0.9766555502620295
```

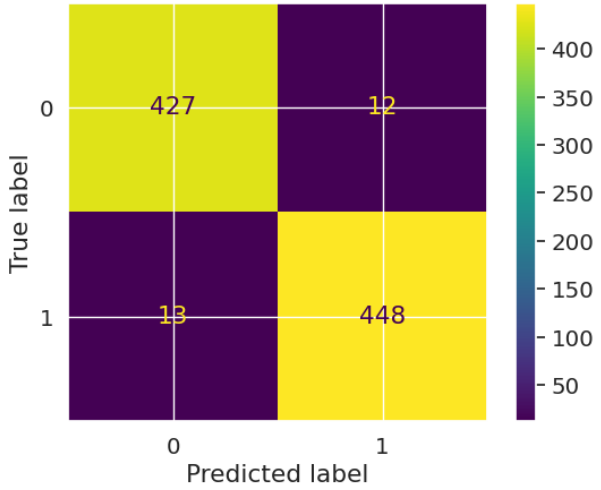
Hình 3.15: Kết quả chạy mô hình Rừng ngẫu nhiên trên X_train



Hình 3.16: Ma trận nhầm lẫn đối với X_train

```
Confusion matrix:
     0    1
0  427   12
1   13  448

Classification report:
              precision    recall  f1-score   support

           0       0.97      0.97      0.97       439
           1       0.97      0.97      0.97       461

    accuracy                           0.97       900
   macro avg       0.97      0.97      0.97       900
weighted avg       0.97      0.97      0.97       900

Random Forest accuracy:  0.9722222222222222
```

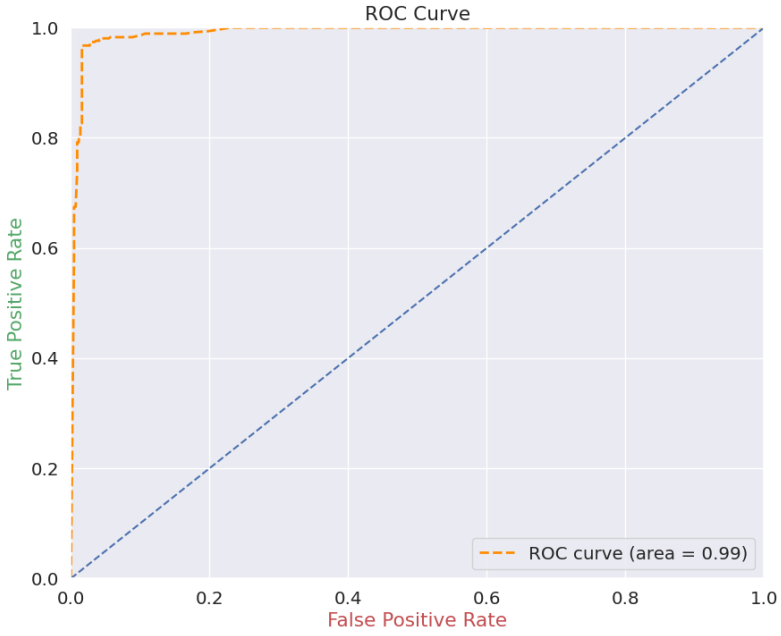
Hình 3.17: Kết quả chạy mô hình Rừng ngẫu nhiên đối với X_test



Hình 3.18: Ma trận nhầm lẫn đối với X_test

Biến độc lập	**Hệ số tương quan**
Chung minh thu nhap	0.377233
Tai san the chap	0.307958
Tinh trang hon nhan	0.142321
Thu nhap	0.146967
Trinh do hoc van	0.025520

Bảng 3.3: Hệ số tương quan trong mô hình rừng ngẫu nhiên



Hình 3.19: Đường cong ROC trong mô hình rừng ngẫu nhiên

3.4.4 XGBOOST

```
XGB_classifier = XGBClassifier()
XGB_classifier.fit(X_train, y_train.values.ravel())

y_pred = XGB_classifier.predict(X_train)
print(confusion_matrix(y_train,y_pred))
print(classification_report(y_train,y_pred))
print('XGBoost accuracy: ', accuracy_score(y_train, y_pred))
```

```
[[1015   18]
 [  32 1034]]
              precision    recall  f1-score   support

           0       0.97      0.98      0.98      1033
           1       0.98      0.97      0.98      1066

    accuracy                           0.98      2099
   macro avg       0.98      0.98      0.98      2099
weighted avg       0.98      0.98      0.98      2099

XGBoost accuracy:  0.9761791329204383
```

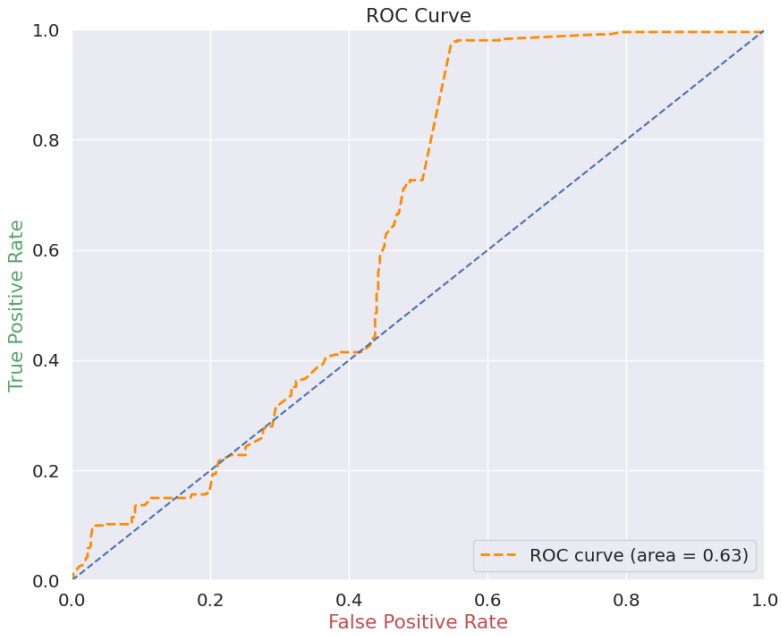
Hình 3.20: Kết quả chạy mô hình XGBOOST trên X_train

```
[[439   0]
 [461   0]]
              precision    recall  f1-score   support

           0       0.49      1.00      0.66       439
           1       0.00      0.00      0.00       461

    accuracy                           0.49       900
   macro avg       0.24      0.50      0.33       900
weighted avg       0.24      0.49      0.32       900

XGBoost accuracy:  0.48777777777777775
```

Hình 3.21: Kết quả chạy mô hình XGBOOST trên X_test

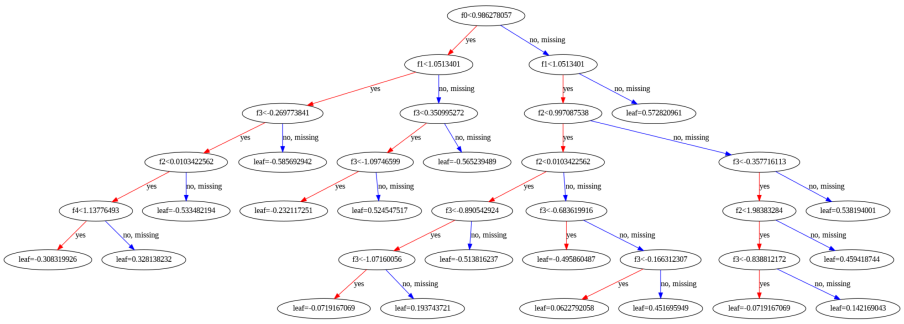
Biến độc lập	Hệ số tương quan
Chung minh thu nhap	0.238928
Tai san the chap	0.711175
Tinh trang hon nhan	0.026359
Thu nhap	0.016651
Trinh do hoc van	0.005986

Bảng 3.4: Hệ số tương quan trong mô hình XGBOOST



Hình 3.22: Đường cong ROC trong mô hình XGBOOST

Ta có thể dựng được cây quyết định cho mô hình như sau:



Hình 3.23: Cây quyết định

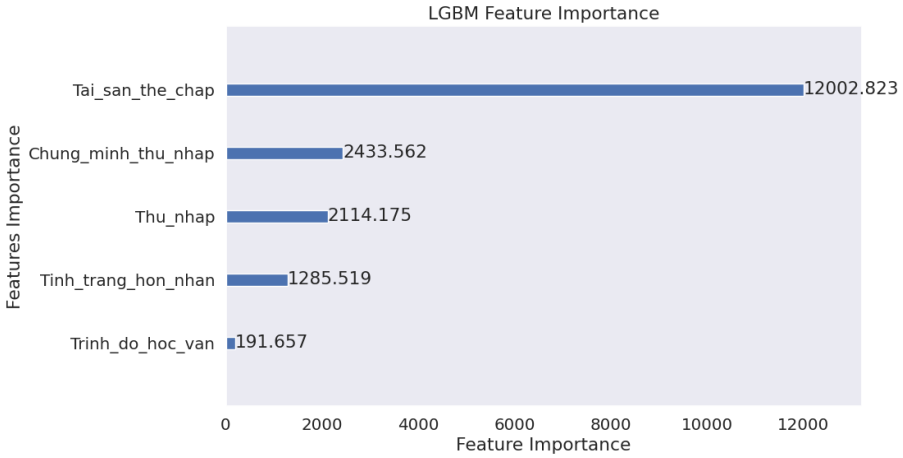
3.4.5 LightGBM

```
# LightGBM Stratified KFold
import lightgbm as lgb
from sklearn.model_selection import KFold, StratifiedKFold

# Initialize the cross-validator
# Choosing StratifiedKFold
sf = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)
train_scores = []  # to store train evaluation scores
test_scores = []  # to store test evaluation scores

# Define the hyperparameters for LightGBM
params = {
    'boosting_type': 'gbdt',
    'objective': 'binary',
    'metric': 'binary_logloss',
    'learning_rate': 0.1,
    'num_leaves': 31,
    'max_depth': -1,
    'random_state': 42
}
# Iterate over the cross-validation splits
for train_idx, test_idx in sf.split(X, y):
  # Split the data into training and test sets
    X_train_1, X_test_1 = X.iloc[train_idx], X.iloc[test_idx]
    y_train_1, y_test_1 = y.iloc[train_idx], y.iloc[test_idx]

    # Train the LightGBM model with selected hyperparameters
    LGBMClassifier = lgb.LGBMClassifier(**params)
    LGBMClassifier.fit(X_train_1, y_train_1)

    # Make predictions on the train data
    y_train_pred = LGBMClassifier.predict(X_train_1)

    # Evaluate the model on the train data
    train_accuracy = accuracy_score(y_train_1, y_train_pred)
    train_precision = precision_score(y_train_1, y_train_pred)
    train_recall = recall_score(y_train_1, y_train_pred)
```

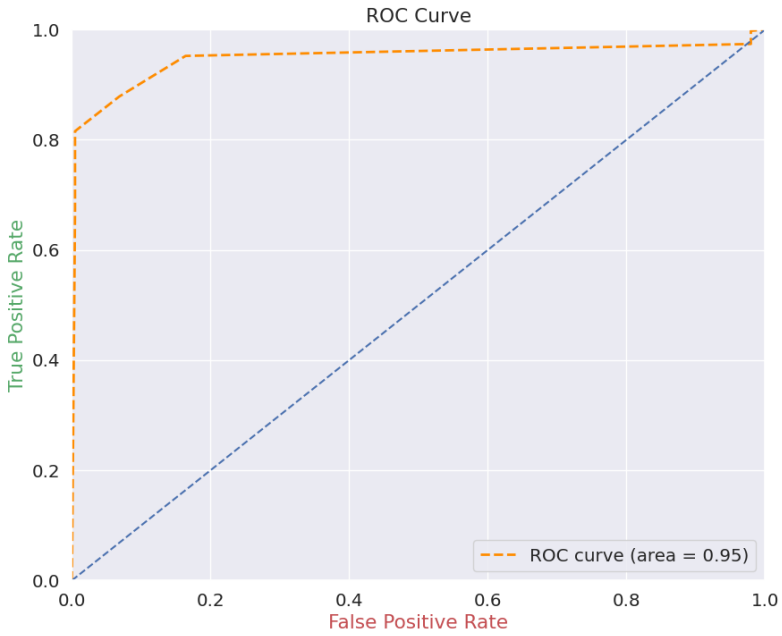
```
38      train_f1 = f1_score(y_train_1, y_train_pred)
39
40      # Store the train evaluation scores
41      train_scores.append({'Accuracy': train_accuracy, 'Precision': train_precision,
        'Recall': train_recall, 'F1-score': train_f1})
42
43      # Make predictions on the test data
44      y_test_pred = LGBMClassifier.predict(X_test_1)
45
46      # Evaluate the model on the test data
47      test_accuracy = accuracy_score(y_test_1, y_test_pred)
48      test_precision = precision_score(y_test_1, y_test_pred)
49      test_recall = recall_score(y_test_1, y_test_pred)
50      test_f1 = f1_score(y_test_1, y_test_pred)
51
52      # Store the test evaluation scores
53      test_scores.append({'Accuracy': test_accuracy, 'Precision': test_precision,
        'Recall': test_recall, 'F1-score': test_f1})
54
55  # Convert the evaluation scores to DataFrame
56  train_scores_df = pd.DataFrame(train_scores)
57  test_scores_df = pd.DataFrame(test_scores)
58
59  # Display the average evaluation scores on train data
60  avg_train_scores = train_scores_df.mean().to_frame(name='Train')
61  print("Average Train Evaluation Scores:")
62  print(avg_train_scores)
63
64  # Display the average evaluation scores on test data
65  avg_test_scores = test_scores_df.mean().to_frame(name='Test')
66  print("Average Test Evaluation Scores:")
67  print(avg_test_scores)
```

```
Average Train Evaluation Scores:
              Train
Accuracy   0.979067
Precision  0.986494
Recall     0.972204
F1-score   0.979294
Average Test Evaluation Scores:
               Test
Accuracy   0.965321
Precision  0.973685
Recall     0.958084
F1-score   0.965653
```

Hình 3.24: Tóm tắt kết quả chạy mô hình



Hình 3.25: Biểu đồ thể hiện hệ số tương quan của các biến độc lập



Hình 3.26: Đường cong ROC ứng với mô hình LightGBM

3.4.6 Naive Bayes

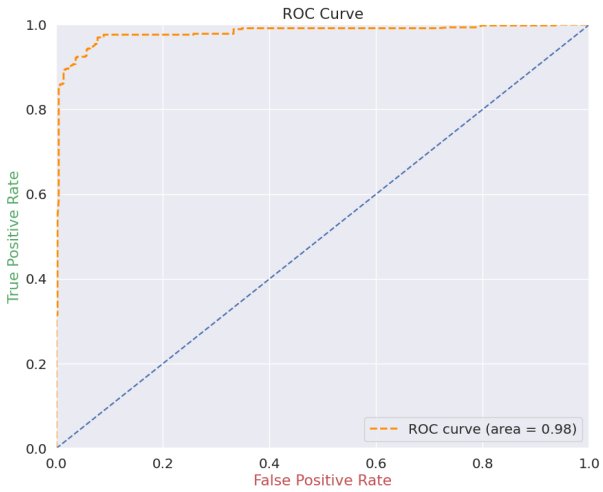
```
from sklearn.naive_bayes import GaussianNB
NB_classifier = GaussianNB()

NB_classifier.fit(X_train, y_train.values.ravel())

y_pred = NB_classifier.predict(X_train)

print('Confusion matrix:')
print(pd.DataFrame(confusion_matrix(y_train,y_pred)),'\n')

print('Classification report:')
print(classification_report(y_train,y_pred))

print('Naive Bayes accuracy: ', accuracy_score(y_train, y_pred))
```

```
Confusion matrix:
     0     1
0  948    85
1   38  1028

Classification report:
              precision    recall  f1-score   support

           0       0.96      0.92      0.94      1033
           1       0.92      0.96      0.94      1066

    accuracy                           0.94      2099
   macro avg       0.94      0.94      0.94      2099
weighted avg       0.94      0.94      0.94      2099

Naive Bayes accuracy:  0.9414006669842783
```

Hình 3.27: Kết quả chạy mô hình Naive Bayes trên X_train

```
Confusion matrix:
     0    1
0  404   35
1   14  447

Classification report:
              precision    recall  f1-score   support

           0       0.97      0.92      0.94       439
           1       0.93      0.97      0.95       461

    accuracy                           0.95       900
   macro avg       0.95      0.94      0.95       900
weighted avg       0.95      0.95      0.95       900

Naive Bayes accuracy:  0.9455555555555556
```

Hình 3.28: Kết quả chạy mô hình Naive Bayes trên X_test



Hình 3.29: Đường cong ROC ứng với mô hình Naive Bayes

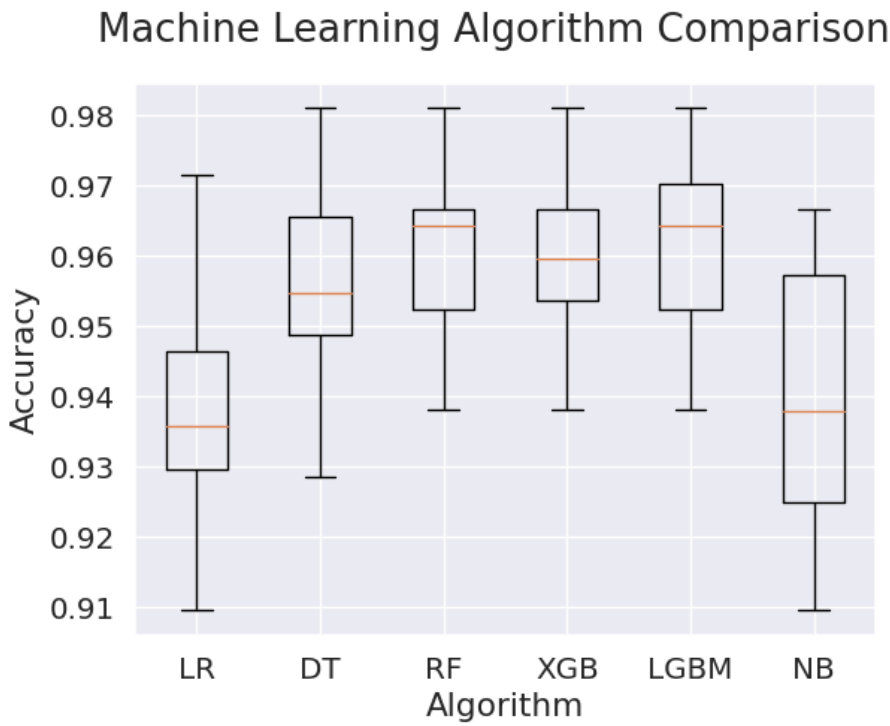
Để đánh giá ý nghĩa của mỗi biến độc lập, ta kiểm tra cách biệt giữa phân phối xác suất có điều kiện của từng biến trên từng class dữ liệu.

```
Feature: Tai san the chap
Class 0 Probability: -0.8222198474605805
Class 1 Probability: 0.7967665125954636

Feature: Chung minh thu nhap
Class 0 Probability: -0.8096539308404713
Class 1 Probability: 0.7845895971465462

Feature: Tinh trang hon nhan
Class 0 Probability: -0.520761699506707
Class 1 Probability: 0.5046405587152136

Feature: Thu nhap
Class 0 Probability: 0.4281006562014238
Class 1 Probability: -0.4148480092458439

Feature: Trinh do hoc van
Class 0 Probability: -0.41790073393955235
Class 1 Probability: 0.40496384442735306
```

Hình 3.30: Phân phối xác suất có điều kiện

3.5 So sánh hiệu quả giữa các mô hình

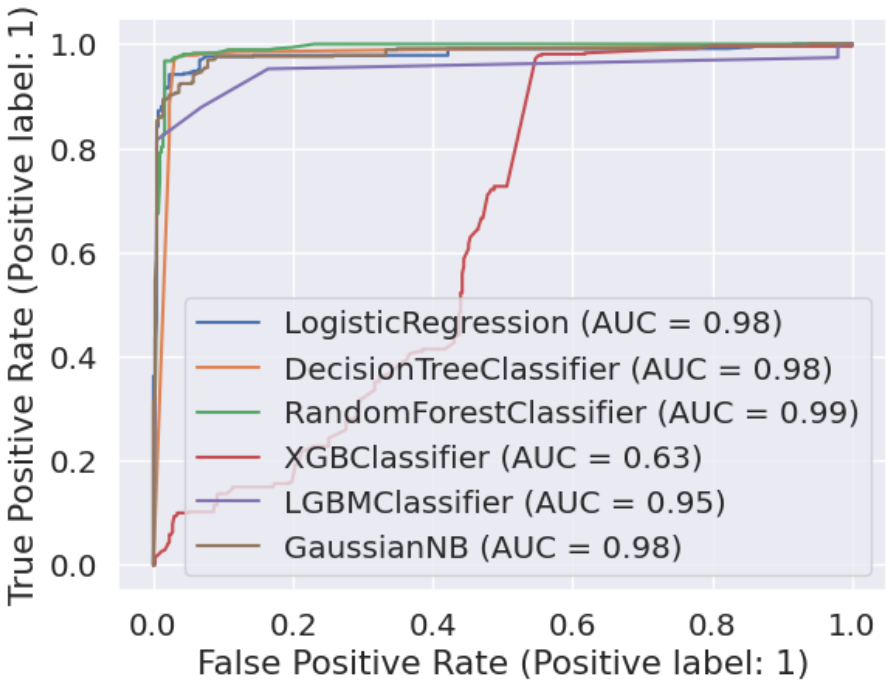
Ta thực hiện cross validation với `kfold` = 10 để tính trung bình độ chính xác từng mô hình.



Hình 3.31: Box plot thể hiện độ chính xác trung bình của mỗi mô hình

Độ chính xác của các mô hình không chênh lệch nhau quá nhiều. Trong đó, mô hình Naive Bayes có độ chính xác thấp nhất; XGBoost và LightGBM có độ chính xác cao nhất, thể hiện sự hiệu quả khi dự đoán khả năng trả nợ không đúng hạn của khách hàng. Các mô hình Cây quyết định và Rừng ngẫu nhiên có độ chính xác khá ổn định và không quá khác nhau, điểm cộng của mô hình Cây quyết định là nó cho phép trực quan hoá quy trình phê duyệt khoản vay để hỗ trợ nhân viên tài chính.

Một cách đánh giá mô hình nữa là so sánh AUC của đường cong ROC của các mô hình.



Hình 3.32: So sánh AUC giữa các mô hình

3.6 Tăng mẫu, giảm mẫu

3.6.1 Tăng mẫu

```
from imblearn.over_sampling import SMOTE
# Upsampling
sm = SMOTE(k_neighbors=5)
X_train_resample, y_train_resample = sm.fit_resample(X_train,y_train)

DT_classifier.fit(X_train_resample, y_train_resample.values.ravel())
DT_classifier.fit(X_train_resample, y_train_resample.values.ravel())

y_pred = DT_classifier.predict(X_train_resample)

print('Confusion matrix:')
print(pd.DataFrame(confusion_matrix(y_train_resample,y_pred)),'\n')

print('Classification report:')
print(classification_report(y_train_resample,y_pred))

```

```
print('Decision Tree accuracy: ', round(accuracy_score(y_train_resample, y_pred),4))
```

```
Confusion matrix:
      0     1
0  1053    13
1    36  1030

Classification report:
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1066
           1       0.99      0.97      0.98      1066

    accuracy                           0.98      2132
   macro avg       0.98      0.98      0.98      2132
weighted avg       0.98      0.98      0.98      2132

Decision Tree accuracy:  0.977
```

Hình 3.33: Kết quả chạy mô hình sau khi tăng mẫu với X_train

```
Confusion matrix:
     0    1
0  429   10
1   53  408

Classification report:
              precision    recall  f1-score   support

           0       0.89      0.98      0.93       439
           1       0.98      0.89      0.93       461

    accuracy                           0.93       900
   macro avg       0.93      0.93      0.93       900
weighted avg       0.93      0.93      0.93       900

Decision Tree accuracy:  0.93
```

Hình 3.34: Kết quả chạy mô hình sau khi tăng mẫu với X_train

Độ chính xác mô hình giảm nhẹ sau khi tăng mẫu, vì thế tăng mẫu là không cần thiết.

3.6.2 Oversampling

```
from imblearn.over_sampling import RandomOverSampler
# Define oversampling strategy
oversample = RandomOverSampler(sampling_strategy='minority')
```

```
4
5 X_train_over, y_train_over = oversample.fit_resample(X_train, y_train)
6
7 print('Predict on over-sampling trainning set')
8 DT_classifier.fit(X_train_over, y_train_over.values.ravel())
9 y_pred_train_over = DT_classifier.predict(X_train_over)
10 print(confusion_matrix(y_train_over,y_pred_train_over))
11 print(classification_report(y_train_over,y_pred_train_over))
12 print('Decision Tree accuracy: ', accuracy_score(y_train_over, y_pred_train_over))
```

```
Predict on over-sampling trainning set
[[1053   13]
 [  36 1030]]
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1066
           1       0.99      0.97      0.98      1066

    accuracy                           0.98      2132
   macro avg       0.98      0.98      0.98      2132
weighted avg       0.98      0.98      0.98      2132

Decision Tree accuracy:  0.9770168855534709
```

Hình 3.35: Kết quả chạy mô hình sau khi áp dụng Oversampling trên X_train

```
Predict on testing set
[[429  10]
 [ 53 408]]
              precision    recall  f1-score   support

           0       0.89      0.98      0.93       439
           1       0.98      0.89      0.93       461

    accuracy                           0.93       900
   macro avg       0.93      0.93      0.93       900
weighted avg       0.93      0.93      0.93       900

Decision Tree accuracy:  0.93
```

Hình 3.36: Kết quả chạy mô hình sau khi áp dụng Oversampling trên X_test

Sau khi áp dụng Oversampling, độ chính xác không được cải thiện so với ban đầu, do đó Oversampling là không cần thiết.

3.6.3 Giảm mẫu

```
from imblearn.under_sampling import RandomUnderSampler
# define undersample strategy
undersample = RandomUnderSampler(sampling_strategy='majority')

# fit and apply the transform
X_train_under, y_train_under = undersample.fit_resample(X, y)

DT_classifier.fit(X_train_under, y_train_under.values.ravel())

y_pred_train_under = DT_classifier.predict(X_train_under)
print(confusion_matrix(y_train_under,y_pred_train_under))
print(classification_report(y_train_under,y_pred_train_under))
print('Decision Tree accuracy: ', accuracy_score(y_train_under, y_pred_train_under))
```

```
[[1454   18]
 [  39 1433]]
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1472
           1       0.99      0.97      0.98      1472

    accuracy                           0.98      2944
   macro avg       0.98      0.98      0.98      2944
weighted avg       0.98      0.98      0.98      2944

Decision Tree accuracy:  0.9806385869565217
```

Hình 3.37: Kết quả chạy mô hình sau khi giảm mẫu trên X_train

```
[[409  30]
 [ 56 405]]
              precision    recall  f1-score   support

           0       0.88      0.93      0.90       439
           1       0.93      0.88      0.90       461

    accuracy                           0.90       900
   macro avg       0.91      0.91      0.90       900
weighted avg       0.91      0.90      0.90       900

Decision Tree accuracy:  0.9044444444444445
```

Hình 3.38: Kết quả chạy mô hình sau khi giảm mẫu trên X_test

Độ chính xác mô hình không cải thiện sau khi giảm mẫu; giảm mẫu trong trường hợp này là không cần thiết.

Chương 4

Kết luận và hướng phát triển đề tài

4.1 Kết luận

- Trong trường hợp dự đoán rủi ro tín dụng, mô hình Cây quyết định có thể giúp xác định các yếu tố quan trọng nhất góp phần nâng cao uy tín tín dụng. Đây là một công cụ mạnh mẽ để dự đoán rủi ro tín dụng vì nó dễ hiểu, linh hoạt và hiệu quả về mặt tính toán, đồng thời có thể cung cấp những hiểu biết sâu sắc có giá trị cho những người ra quyết định. Mô hình này có thể dễ dàng hình dung, giúp ích cho việc truyền đạt kết quả phân tích rủi ro tín dụng tới các bên liên quan để giúp người ra quyết định hiểu được các yếu tố góp phần gây ra rủi ro tín dụng và đưa ra quyết định sáng suốt về việc cho vay. Kết quả và các chỉ số đánh giá của các mô hình machine learning dùng để dự đoán khả năng trả nợ trong báo cáo này cho kết quả rất cao, cho thấy khả năng dự đoán của các mô hình là rất đáng tin cậy. Các mô hình XGBoost, LightGBM và Cây quyết định có kết quả ổn định hơn và hiệu suất dự đoán cao hơn các mô hình khác. Tuy nhiên, mô hình Cây quyết định có thể được hiểu một cách trực quan và giúp nhân viên tín dụng dễ dàng tối ưu hóa quy trình đánh giá khách hàng và đưa ra các quyết định thông minh về cấp vốn vay và quản lý rủi ro tín dụng.

- Trong báo cáo này, chúng em đã trình bày việc sử dụng thuật toán machine learning trên tập dữ liệu mô phỏng để dự đoán khả năng trả nợ. Để đạt được hiệu suất tốt nhất, nghiên cứu này cho thấy rằng việc xử lý trước dữ liệu,

lựa chọn cẩn thận các kỹ thuật để cân bằng tập dữ liệu và thuật toán phân loại đều rất quan trọng. Hồi quy logistic, Cây quyết định, Rừng ngẫu nhiên, XGBoost, LightGBM và Naive Bayes hoạt động khá tốt trên tập dữ liệu này. Trong tương lai, chúng em muốn tiếp tục khám phá các thuật toán học tập và kỹ thuật giảm kích thước phức tạp hơn, lựa chọn các siêu tham số phù hợp để cải thiện hơn nữa hiệu suất của mô hình trong nhiệm vụ dự đoán quan trọng này.

4.2 Hướng phát triển của đề tài

Dựa trên đánh giá mô hình, có thể điều chỉnh các siêu tham số của mô hình machine learning để tối ưu hóa hoặc cải thiện mô hình. Ví dụ: có thể điều chỉnh độ sâu của cây, số lượng mẫu tối thiểu cần thiết để phân chia nút. Việc tinh chỉnh các tham số này sẽ cải thiện khả năng dự đoán của mô hình và giảm thiểu việc khớp quá mức. Sử dụng các kỹ thuật tổng hợp như Đóng bao, Rừng ngẫu nhiên hoặc Tăng cường để kết hợp nhiều cây quyết định. Việc sử dụng các kỹ thuật này làm giảm sự biến động của mô hình và cải thiện khả năng dự đoán trên dữ liệu mới. Ngoài ra, việc xác định một ngưỡng thích hợp để dự đoán rủi ro tín dụng giúp đạt được sự cân bằng giữa việc đưa ra quyết định sai lầm về rủi ro và quyết định sai lầm về khả năng trả nợ của khách hàng. Cuối cùng, chúng em nên nghiên cứu các mô hình khác để tìm ra mô hình tốt nhất cho bài toán phân loại tín dụng.

Tài liệu tham khảo

[1] Jomark Pablo Noriega, Luis Antonio Rivera, and José Alfredo Herrera. Machine learning for credit risk prediction: A systematic literature review. *Data*, 8(11), 2023.