

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM
KHOA CÔNG NGHỆ THÔNG TIN
MÔN HỌC MÁY



HCMUTE

BÁO CÁO ĐỀ TÀI CUỐI KỲ

DỰ ĐOÁN RỦI RO CHO VAY TÍN DỤNG

LỚP HỌC PHẦN: MALE431085_23_2_04CLC

HỌC KỲ II – NĂM HỌC 2023-2024

Sinh viên thực hiện:

- 21110298: Đặng Kim Thành

- 21110175: Nguyễn Văn Hào

Giảng viên hướng dẫn : ThS. Trần Quang Khải

Thành phố Hồ Chí Minh, tháng 5 năm 2024

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP HCM
KHOA CÔNG NGHỆ THÔNG TIN
MÔN HỌC MÁY**

**CỘNG HÒA XÃ HỘI CHỦ
NGHĨA
VIỆT NAM
Độc lập – Tự do – Hạnh phúc**



HCMUTE

BÁO CÁO ĐỀ TÀI CUỐI KỲ

DỰ ĐOÁN RỦI RO CHO VAY TÍN DỤNG

LỚP HỌC PHẦN: MALE431085_23_2_04CLC

HỌC KỲ II – NĂM HỌC 2023-2024

Sinh viên thực hiện:

- 21110298: Đặng Kim Thành

- 21110175: Nguyễn Văn Hào

Giảng viên hướng dẫn : ThS. Trần Quang Khải

Thành phố Hồ Chí Minh, tháng 5 năm 2024

DANH SÁCH NHÓM THỰC HIỆN ĐỀ TÀI
MÔN NHẬP MÔN DỮ LIỆU LỚN

HỌC KỲ II NĂM HỌC 2023-2024

- 1. Giảng viên hướng dẫn:** ThS. Trần Quang Khải
- 2. Mã lớp học:** MALE431085_23_2_04CLC
- 3. Tên đề tài :** Dự đoán rủi ro cho vay tín dụng
- 4. Danh sách nhóm viết tiểu luận cuối kỳ:**

STT	HỌ VÀ TÊN SINH VIÊN	Mã số sinh viên	Tỉ lệ % tham gia
01	Đặng Kim Thành	21110298	100%
02	Nguyễn Văn Hào	21110175	100%

NHẬN XÉT CỦA GV

Giảng viên chấm điểm

LỜI CẢM ƠN

Chúng em xin gửi lời cảm ơn sâu sắc đến Thầy Trần Quang Khải, Thầy đã tận tình giảng dạy trong suốt quá trình tìm hiểu và học tập môn Học máy. Trong quá trình nghiên cứu và học, thầy đã nhiệt huyết giảng dạy và những kiến thức mà thầy đã truyền đạt không chỉ là nền tảng cho quá trình nghiên cứu, thực hiện đồ án mà còn là tư liệu quý báu để chúng em trong quá trình học tập và tìm hiểu sau này cũng như là trên con đường nghề nghiệp phía trước.

Một lần nữa, em xin bày tỏ lòng biết ơn đến Thầy. Xin kính chúc Thầy luôn dồi dào sức khỏe, đạt được nhiều thành công trong công việc.

Thành phố Hồ Chí Minh, tháng 5 năm 2024

KẾ HOẠCH PHÂN CÔNG NHIỆM VỤ
THỰC HIỆN ĐỀ TÀI CUỐI KÌ
HỌC KỲ II NĂM HỌC 2023-2024

1. Mã lớp: MALE431085_23_2_04CLC
2. Tên đề tài: Dự báo rủi ro cho vay tín dụng
3. Bảng phân công nhiệm vụ:

Nội dung hoàn thành	Sinh viên hoàn thành	Tỷ lệ hoàn thành
- Tìm hiểu tập dữ liệu, tiền xử lý dữ liệu và cài đặt các mô hình KNN, AdaBoosting, RandomForest, ANN	Đặng Kim Thành	100%
- Tìm hiểu tập dữ liệu, tiền xử lý dữ liệu và cài đặt các mô hình Logistic Regression, GradientBoosting, Naïve Bayes	Nguyễn Văn Hào	100%

MỤC LỤC



DANH MỤC HÌNH ẢNH	vii
PHẦN MỞ ĐẦU	1
1. Phát biểu về bài toán	1
2. Mục đích thực hiện	1
3. Tổng quan về dữ liệu	2
4. Bố cục đề tài	2
PHẦN NỘI DUNG	3
CHƯƠNG 1: DATASET	3
1.1. Tổng quan về tập dữ liệu	3
1.2. Khám phá tổng quát và định hướng phân tích	3
1.3. Xử lý dữ liệu thiếu và dọn dẹp dữ liệu	5
CHƯƠNG 2. TIỀN XỬ LÝ VÀ CHUẨN HÓA DỮ LIỆU	7
2.1. Các bước phân tích và nhận xét về từng đặc trưng	7
2.1.2. Đặc trưng về sở hữu xe (Car Owners)	8
2.1.3. Đặc trưng tình trạng hôn nhân	9
2.1.4. Đặc trưng nghề nghiệp	10
2.1.5. Tỷ lệ phân bố của biến Thành phố và Bang	11
2.1.6. Đặc trưng bang người dùng sinh sống	13
2.2. Phân tích mối tương quan giữa các biến	17
2.3. Xử lý ngoại lai	19
2.3.1. Xử lý ngoại lai của đặc trưng Tuổi	19
2.3.2. Xử lý ngoại lai của đặc trưng Income	19
2.4. Trực quan cung cấp cái nhìn rõ ràng về sự phân bố của biến mục tiêu, Risk_Flag	20
CHƯƠNG 3: PHÂN TÍCH THỐNG KÊ VÀ KIỂM ĐỊNH GIẢ THUYẾT	22
3.1. Phương pháp thống kê sử dụng	22
3.2. Phân tích và kiểm định giả thuyết	22

3.3. Đánh giá mức độ tin cậy của kết quả	25
CHƯƠNG 4: MÔ HÌNH HÓA VÀ DỰ ĐOÁN	27
4.1. Chuẩn hóa và chuẩn bị dữ liệu huấn luyện	27
4.1.1. Phân tích thành phần chính (PCA) cho CURRENT_JOB_YRS và Experience	27
4.1.2. Chuẩn bị dữ liệu	29
4.2. Mô hình logictis regresion	30
4.2.1. Sơ lược về mô hình	30
4.2.2. Đào tạo và đánh giá mô hình.....	31
4.3. Mô hình KNN	33
4.3.1. Sơ lược về mô hình	33
4.4. Mô hình Random Forest	35
4.4.1. Sơ lược về mô hình	35
4.4.2. Đào tạo và đánh giá mô hình.....	35
4.4.3. Đánh giá kết quả và giải thích mô hình	37
4.5. Mô hình AdaBoosting.....	40
4.5.1. Sơ lược về mô hình	40
4.5.2. Đào tạo và đánh giá mô hình.....	40
4.5.3. Đánh giá kết quả và giải thích mô hình	41
4.6. Mô hình GradientBoosting.....	42
4.6.1. Sơ lược về mô hình	42
4.6.2. Đào tạo và đánh giá mô hình.....	43
4.6.3. Đánh giá kết quả và giải thích mô hình	44
4.7. Mô hình Naive Bayes	45
4.7.1. Sơ lược về mô hình	45
4.7.3. Đánh giá kết quả và giải thích mô hình	47
4.8. Mô hình ANN	49
4.8.1. Sơ lược về mô hình	49
4.8.2. Đào tạo và đánh giá mô hình.....	49
4.8.3. Đánh giá kết quả và giải thích mô hình	53
CHƯƠNG 5. KẾT LUẬN VÀ KIẾN NGHỊ.....	57

5.1. Tóm tắt kết quả nghiên cứu	57
5.2. Kiến nghị dựa trên kết quả	57
5.3. Hướng nghiên cứu và phát triển tiếp theo	57
TÀI LIỆU THAM KHẢO.....	58

DANH MỤC HÌNH ẢNH

Hình 1: Biểu đồ cột giữa đặc trưng House Ownership và Risk Flag	7
Hình 2: Biểu đồ cột giữa đặc trưng Car Ownership và Risk Flag.....	8
Hình 3: Biểu đồ cột giữa đặc trưng Marital Status và Risk Flag	9
Hình 4: Biểu đồ cột giữa đặc trưng Profession và Risk Flag	10
Hình 5: Biểu đồ cột giữa đặc trưng STATE và Risk Flag.....	13
Hình 6: Biểu đồ tần suất về tuổi tác (Age)	14
Hình 7: Biểu đồ tần suất về thu nhập (Income).....	16
Hình 8: Biểu đồ tương quan giữa các biến.....	17
Hình 9: Biểu đồ hộp về đặc trưng tuổi (Age).....	19
Hình 10: Biểu đồ hộp của đặc trưng thu nhập (Income)	20
Hình 11: Biểu đồ cột và biểu đồ tròn về sự phân bố của biến mục tiêu Risk_Flag	20
Hình 12: Kết quả kiểm định khả năng vỡ nợ theo sở hữu xe	23
Hình 13: Kết quả kiểm định khả năng vỡ nợ theo tình trạng hôn nhân	24
Hình 14: Kết quả kiểm định khả năng vỡ nợ theo sở hữu nhà	25
Hình 15: DataFrame đã chuẩn hóa cho PCA	28
Hình 16: DataFrame sau khi áp dụng PCA	28
Hình 17: Nối DataFrame PCA với DataFrame ban đầu.....	29
Hình 18: DataFrame sau khi mã hóa các biến phân loại và loại bỏ cột Id.....	30
Hình 19: Báo cáo các độ đo của Logistic Regression	31
Hình 20: Confusion matrix cho dự đoán của Logistic Regression.....	32
Hình 21. Báo cáo các độ đo của mô hình KNN	34
Hình 22. Mức độ quan trọng của các thuộc tính trong RandomForest	36
Hình 23. Biểu đồ thể hiện mức độ quan trọng của các thuộc tính trong RandomForest ...	37
Hình 24. Báo cáo về các độ đo của mô hình Random Forest.....	38
Hình 25. Confusion matrix cho dự đoán của mô hình Random Forest.....	39
Hình 26: Báo cáo các độ đo của mô hình GradientBoostingClassifier	44
Hình 27: Confusion matrix cho dự đoán của GradientBoostingClassifier.....	44
Hình 28: Báo cáo về các độ đo của GradientBoostingClassifier	47

Hình 29: Confusion matrix cho dự đoán của Naive Bayes	48
Hình 30. Biểu đồ biểu diễn quá trình học của mô hình ANN	53
Hình 31. DataFrame biểu diễn độ chính xác của từng mô hình	55

PHẦN MỞ ĐẦU

1. Phát biểu về bài toán

Học máy - Machine Learning là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc nghiên cứu và xây dựng các kĩ thuật cho phép các hệ thống “học” tự động từ dữ liệu để giải quyết những vấn đề cụ thể.

Và hiện nay với lượng dữ liệu khổng lồ của người dùng Internet cùng với sự bùng nổ của cách mạng phần cứng càng làm cho tốc độ máy tính được cải thiện thì Học máy đang trở thành xu hướng và ngày càng len lỏi sâu vào cuộc sống của chúng ta. Machine learning là một thành phần quan trọng của lĩnh vực khoa học dữ liệu đang phát triển. Thông qua việc sử dụng các phương pháp thống kê, các thuật toán được đào tạo để đưa ra các phân loại hoặc dự đoán và khám phá những thông tin chi tiết từ chính các dự án khai thác dữ liệu.

Thông qua các thông tin chi tiết có được để thúc đẩy việc đưa ra quyết định đối với các ứng dụng và doanh nghiệp, tác động mạnh đến các chỉ số tăng trưởng. Khi dữ liệu lớn tiếp tục nhu cầu mở rộng và phát triển đòi hỏi nhu cầu tuyển dụng các nhà khoa học dữ liệu sẽ tăng lên. Họ sẽ được yêu cầu giúp xác định các câu hỏi kinh doanh có liên quan nhất và dữ liệu để trả lời chúng.

Bài toán của machine learning thường được chia làm hai loại là dự đoán (prediction) và phân loại (classification).

2. Mục đích thực hiện

Thông qua phân tích dữ liệu, đưa ra giả thuyết và thiết kế các mô hình để giải quyết các vấn đề thực tế. Từ đó áp dụng các thuật toán học máy như hồi quy, phân loại và gom cụm, từ đó hiểu rõ hơn về cách mỗi phương pháp hoạt động và khi nào nên áp dụng chúng. Đồng thời, qua các thao tác cũng mang mục đích nắm vững quy trình làm việc trong dự án học máy, từ việc thu thập dữ liệu đến triển khai và đánh giá mô hình.

3. Tổng quan về dữ liệu

Dữ liệu là yếu tố quan trọng nhất trong bất kỳ dự án Machine Learning nào. Trong quá trình này, dữ liệu không chỉ là nguồn thông tin mà còn là "nhiên liệu" cung cấp cho mô hình Machine Learning.

Việc thành công của một dự án Machine Learning phụ thuộc phần lớn vào chất lượng và độ đa dạng của dữ liệu. Trước khi xây dựng mô hình, việc hiểu và tiền xử lý dữ liệu là bước quan trọng, bao gồm làm sạch dữ liệu, khám phá tính chất của nó, chuẩn hóa và mã hóa các biến, xử lý dữ liệu bị thiếu, và phân chia dữ liệu thành các tập huấn luyện, kiểm tra và validation.

Quản lý dữ liệu cũng là một phần không thể thiếu, đảm bảo rằng dữ liệu được bảo mật và toàn vẹn trong suốt quá trình. Tóm lại, dữ liệu là yếu tố quyết định thành công của một dự án Machine Learning, và việc làm việc hiệu quả với dữ liệu là chìa khóa để xây dựng mô hình hiệu suất cao.

4. Bố cục đề tài

Chương 1: Dataset

Chương 2. Tiền xử lý và chuẩn hóa dữ liệu

Chương 3: Phân tích thống kê và kiểm định giả thuyết

Chương 4: Mô hình hóa và dự đoán

Chương 5. Kết luận và kiến nghị

PHẦN NỘI DUNG

CHƯƠNG 1: DATASET

1.1. Tổng quan về tập dữ liệu

Cho vay tài chính là một trong những dịch vụ phổ biến nhất được ngân hàng cung cấp, cho dù đó là vay mua nhà, vay mua ô tô, vay học vấn và v.v. Nhưng đồng thời, có những rủi ro liên quan đến việc cho vay tiền đến khách hàng. Ngân hàng có thể nào tìm ra nguy cơ liên quan đến khách hàng nào đã nộp đơn vay không?

Một tổ chức về cho vay muốn dự đoán ai có thể là những người không thực hiện nghĩa vụ thanh toán cho sản phẩm vay tiêu dùng. Họ có dữ liệu về hành vi lịch sử của khách hàng dựa trên những gì họ đã quan sát được. Do đó, khi họ thu hút khách hàng mới, họ muốn dự đoán ai là người có nguy cơ hơn và ai là người không có nguy cơ.

Link tập dữ liệu:

<https://www.kaggle.com/datasets/subhamjain/loan-prediction-based-on-customer-behavior>

1.2. Khám phá tổng quát và định hướng phân tích

Các nhãn data trong tập dữ liệu:

- ID: Mã số duy nhất của người dùng.
- Income: Thu nhập của người dùng.
- Age: Tuổi của người dùng.
- Experience: Kinh nghiệm chuyên môn của người dùng tính bằng năm.
- Profession: Nghề nghiệp của người dùng.
- Married/Single: Tình trạng hôn nhân của người dùng, có kết hôn hay chưa.
- House_Ownership: Tình trạng sở hữu nhà, có sở hữu nhà, thuê nhà hay không sở hữu nhà.
- Car_Ownership: Người dùng có sở hữu xe hơi hay không.
- STATE: Bang hoặc vùng lãnh thổ cư trú của người dùng.
- CITY: Thành phố hoặc khu vực cư trú của người dùng.
- CURRENT_JOB_YRS: Số năm kinh nghiệm làm việc hiện tại của người dùng.

- **CURRENT_HOUSE_YRS:** Số năm cư trú tại địa chỉ hiện tại của người dùng.
- **Risk_Flag:** Biến mục tiêu, chỉ liệu người dùng đã gặp rủi ro về việc vỡ nợ hay không.

Định hướng phân tích:

Phân tích phân phối biến mục tiêu (Risk_Flag): Có thể xem xét phân phối của biến mục tiêu để hiểu tỷ lệ người dùng đã vỡ nợ so với tỷ lệ không vỡ nợ trong dữ liệu.

Phân tích tương quan: Có thể tiến hành phân tích tương quan giữa các biến đầu vào (Income, Age, Experience, Profession, Marital_Status, House_Ownership, Car_Ownership, STATE, CITY, CURRENT_JOB_YRS, CURRENT_HOUSE_YRS) và biến mục tiêu (Risk_Flag) để xác định mối quan hệ giữa chúng.

Khám phá dữ liệu bất thường: Kiểm tra dữ liệu để phát hiện các giá trị bất thường, dữ liệu còn thiếu, hoặc các điểm ngoại lai có thể ảnh hưởng đến kết quả phân tích.

Mô hình hóa dự đoán: Dựa trên dữ liệu lịch sử và các biến đầu vào, tổ chức có thể xây dựng các mô hình dự đoán để phân loại khách hàng thành hai nhóm: người có nguy cơ vỡ nợ và người không có nguy cơ vỡ nợ. Các mô hình có thể bao gồm học máy phân loại như Random Forest, Logistic Regression, hoặc Support Vector Machines.

Đánh giá mô hình: Tổ chức cần đánh giá hiệu suất của các mô hình dự đoán bằng cách sử dụng các phép đo như độ chính xác, độ nhạy, độ đặc hiệu và F1-score để đảm bảo rằng mô hình có khả năng dự đoán chính xác và hiệu quả.

Tối ưu hóa chiến lược cho vay: Dựa trên kết quả phân tích và dự đoán từ mô hình, tổ chức có thể tối ưu hóa chiến lược cho vay của mình bằng cách tập

trung vào việc cung cấp các sản phẩm vay cho nhóm người không có nguy cơ vỡ nợ và áp đặt các biện pháp hạn chế đối với nhóm có nguy cơ cao hơn.

1.3. Xử lý dữ liệu thiếu và dọn dẹp dữ liệu

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 252000 entries, 0 to 251999
Data columns (total 13 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Id                     252000 non-null  int64
1   Income                 252000 non-null  int64
2   Age                    252000 non-null  int64
3   Experience              252000 non-null  int64
4   Married/Single         252000 non-null  object
5   House_Ownership        252000 non-null  object
6   Car_Ownership          252000 non-null  object
7   Profession              252000 non-null  object
8   CITY                   252000 non-null  object
9   STATE                  252000 non-null  object
10  CURRENT_JOB_YRS        252000 non-null  int64
11  CURRENT_HOUSE_YRS      252000 non-null  int64
12  Risk_Flag              252000 non-null  int64
dtypes: int64(7), object(6)
memory usage: 25.0+ MB
```

DataFrame này có 252.000 dòng, chia thành 13 cột. Mỗi cột được liệt kê cùng với thông tin về số lượng giá trị không rỗng

```
numerical_summary = df.describe() # For numerical variables
numerical_columns_count = numerical_summary.shape[1]
print(f'There are {numerical_columns_count} numeric columns in the dataset')
```

```
categorical_summary = df.describe(include=['O']).T
categorical_columns_count = categorical_summary.shape[0]
print(f'There are {categorical_columns_count} categorical columns in the dataset')
```

Kiểm tra tổng các cột số trong dữ liệu và kiểm tra các loại dữ liệu trong tập dữ liệu.

Kết quả thu được 7 cột dữ liệu số và 6 loại cột trong tập dữ liệu

```
There are 7 numeric columns in the dataset
```

```
There are 6 categorical columns in the dataset
```

Tiếp theo kiểm tra các giá trị NULL trong tập dữ liệu và loại bỏ nó

```
df.isnull().sum()
```

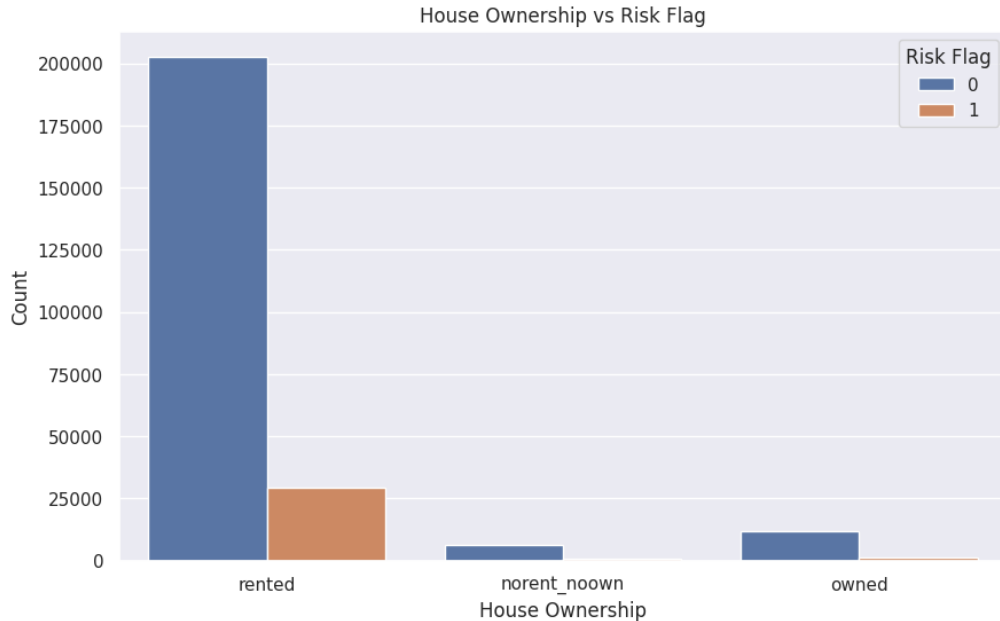
```
Id          0
Income      0
Age         0
Experience  0
Married/Single  0
House_Ownership  0
Car_Ownership  0
Profession  0
CITY        0
STATE       0
CURRENT_JOB_YRS  0
CURRENT_HOUSE_YRS  0
Risk_Flag   0
dtype: int64
```

Không có dữ liệu rỗng nào trong tập dữ liệu này

CHƯƠNG 2. TIỀN XỬ LÝ VÀ CHUẨN HÓA DỮ LIỆU

2.1. Các bước phân tích và nhận xét về từng đặc trưng

2.1.1. Đặc trưng về sở hữu nhà



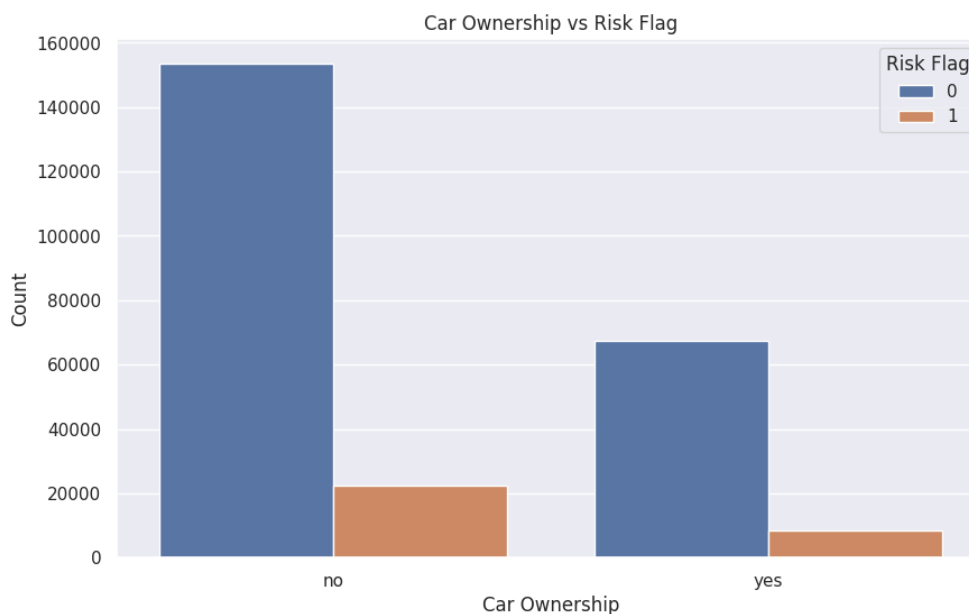
Hình 1. Biểu đồ cột giữa đặc trưng House Ownership và Risk Flag

Phân bố sở hữu nhà: Phần lớn các cá nhân trong tập dữ liệu là người thuê nhà ('người thuê'), đông hơn đáng kể những người sở hữu nhà của họ ('sở hữu') hoặc không có quyền sở hữu ổn định ('norent_noown').

Phân bố rủi ro: Trên tất cả các loại quyền sở hữu nhà, số lượng cá nhân không có rủi ro (Risk Flag = 0) cao hơn số người được gắn cờ là có rủi ro (Risk Flag = 1).

So sánh giữa các rủi ro: Tỷ lệ cá nhân bị gắn cờ rủi ro trong danh mục cho thuê cao hơn so với tỷ lệ trong danh mục sở hữu, cho thấy rằng người thuê nhà có thể gặp phải các yếu tố dẫn đến xếp hạng rủi ro cao hơn. Điều này có thể là do sự bất ổn về tài chính, tình trạng kinh tế thấp hơn hoặc điều kiện sống kém an toàn hơn liên quan đến tài sản thuê.

2.1.2. Đặc trưng về sở hữu xe (Car Owners)



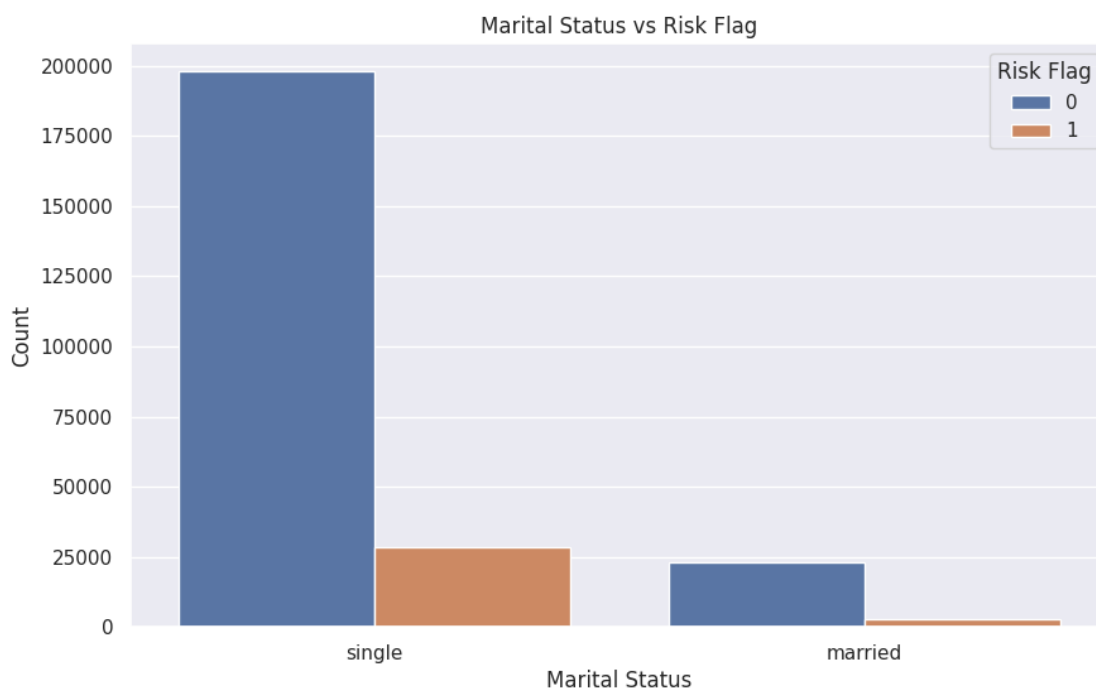
Hình 2: Biểu đồ cột giữa đặc trưng Car Ownership và Risk Flag

Phân phối quyền sở hữu ô tô: Số lượng người không sở hữu ô tô ('no') lớn hơn so với những người có ('yes'). Điều này cho thấy rằng việc sở hữu ô tô không phổ biến trong nhóm dân số được trình bày trong tập dữ liệu.

Phân bố rủi ro: Trong số cả chủ sở hữu ô tô và người không sở hữu ô tô, phần lớn được phân loại là không có rủi ro (Risk Flag = 0). Điều này cho thấy rằng dân số nói chung, bất kể sở hữu ô tô, có xu hướng có mức độ rủi ro thấp.

Những người sở hữu ô tô dường như có tỷ lệ cảnh báo rủi ro thấp hơn so với những người không sở hữu ô tô. Điều này cho thấy rằng việc sở hữu ô tô có thể tương quan với mức độ rủi ro thấp hơn, điều này có thể là do các yếu tố kinh tế xã hội khác nhau như sự ổn định tài chính cao hơn của những người sở hữu ô tô.

2.1.3. Đặc trưng tình trạng hôn nhân



Hình 3: Biểu đồ cột giữa đặc trưng Marital Status và Risk Flag

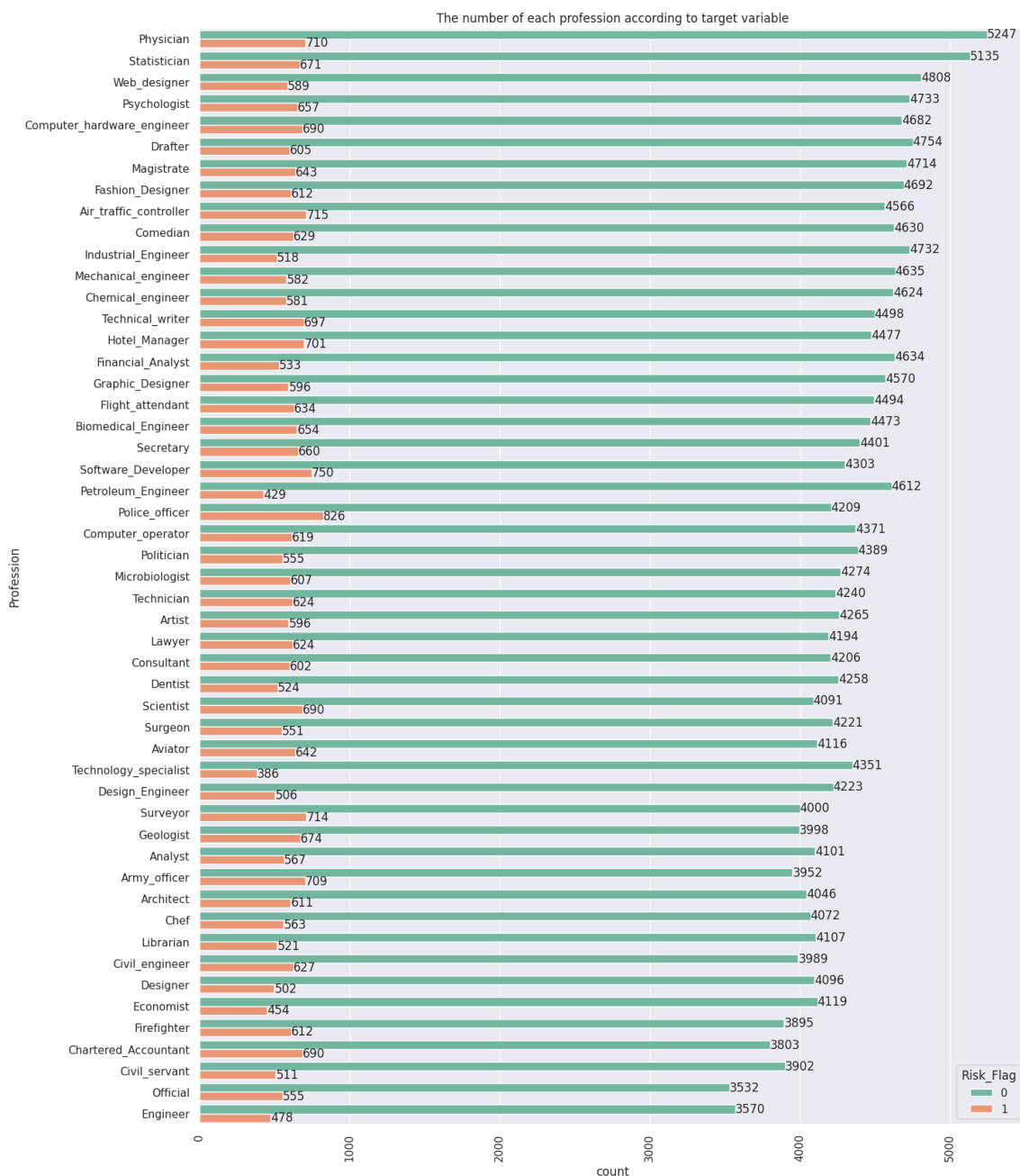
Phân bố tình trạng hôn nhân: Có số lượng cá nhân độc thân cao hơn so với những người đã kết hôn trong tập dữ liệu. Điều này có thể gợi ý một nhóm nhân khẩu học trẻ hơn hoặc một nhóm dân số có xu hướng kết hôn muộn hơn hoặc ít hơn.

Phân bố rủi ro: Tỷ lệ cá nhân có cờ rủi ro là 1 (nguy cơ cao hơn) ở những người độc thân tương đối cao hơn so với những người đã kết hôn. Điều này có thể chỉ ra rằng việc độc thân có liên quan đến nguy cơ rủi ro tăng lên một chút, có thể do các yếu tố như sự ổn định tài chính kém hơn hoặc trách nhiệm xã hội khác biệt so với những người đã kết hôn.

Tổng số người độc thân không có rủi ro cao hơn đáng kể so với số người đã kết hôn, phản ánh số lượng người độc thân cơ bản lớn hơn. Tuy nhiên, tỷ lệ rủi ro và không rủi ro trong nhóm độc thân cũng cao hơn, cho thấy các yếu tố rủi ro cụ thể hoặc khả năng dễ bị tổn thương liên quan đến việc độc thân.

Các cá nhân đã kết hôn có tỷ lệ rủi ro tương đối thấp hơn, có thể do thu nhập tổng hợp, trách nhiệm chung hoặc các yếu tố kinh tế xã hội khác góp phần ổn định và đánh giá rủi ro thấp hơn.

2.1.4. Đặc trưng nghề nghiệp



Hình 4: Biểu đồ cột giữa đặc trưng Profession và Risk Flag

Phân bố nghề nghiệp:

- Bộ dữ liệu bao gồm nhiều ngành nghề khác nhau, với sự khác biệt đáng kể về số lượng giữa các ngành nghề khác nhau.
- Một số ngành nghề như 'Bác sĩ', 'Nhà thống kê' và 'Nhà thiết kế web' phổ biến hơn trong tập dữ liệu.

Phân bố rủi ro:

- Trong hầu hết các ngành nghề, số cá nhân không có rủi ro (Risk Flag = 0) đông hơn đáng kể những người có rủi ro (Risk Flag = 1).
- Một số ngành nghề nhất định như 'Bác sĩ', 'Nhà thiết kế web' và 'Nhà tâm lý học' có số lượng cá nhân không gặp rủi ro đặc biệt cao, cho thấy mức độ rủi ro thấp hơn liên quan đến những nghề nghiệp này.

Hình ảnh trực quan nêu bật các ngành nghề có tỷ lệ rủi ro và không rủi ro khác nhau. Ví dụ: các ngành nghề như 'Kỹ sư dầu khí' và 'Nhà phát triển phần mềm' cho thấy số lượng cá nhân gặp rủi ro cao hơn so với nhiều nghề khác, cho thấy rằng các yếu tố ngành hoặc đặc điểm công việc cụ thể có thể ảnh hưởng đến việc đánh giá rủi ro.

Mặt khác, các ngành nghề như 'Nhà thống kê' và 'Máy tính, kỹ sư phần cứng' cho thấy sự phân bố rủi ro tương đối cân bằng nhưng vẫn có mức độ chi phối rủi ro thấp hơn.

2.1.5. Tỷ lệ phân bố của biến Thành phố và Bang

❖ Địa điểm có rủi ro cao:

Bhubaneswar, Odisha có tỷ lệ rủi ro cao nhất khoảng 32,62%. Điều này có thể chỉ ra các yếu tố địa phương hoặc điều kiện kinh tế góp phần làm tăng mức độ rủi ro của người dân.

Gwalior, Madhya Pradesh và Bettiah, Bihar cũng có tỷ lệ rủi ro cao đáng kể, lần lượt là 27,27% và 26,70%. Những khu vực này có thể có những thách thức kinh tế xã hội cụ thể góp phần làm tăng mức độ rủi ro cao hơn này.

❖ ***Vị trí có rủi ro thấp hơn:***

Ở phía bên kia của quang phổ, các thành phố như Gandhinagar, Gujarat và Dehradun, Uttarakhand có tỷ lệ rủi ro thấp hơn đáng kể, lần lượt khoảng 2,61% và 2,63%. Điều này cho thấy các điều kiện kinh tế hoặc nhân khẩu học ổn định hơn có thể giảm thiểu các yếu tố rủi ro.

❖ ***Những biến đổi về mặt địa lý:***

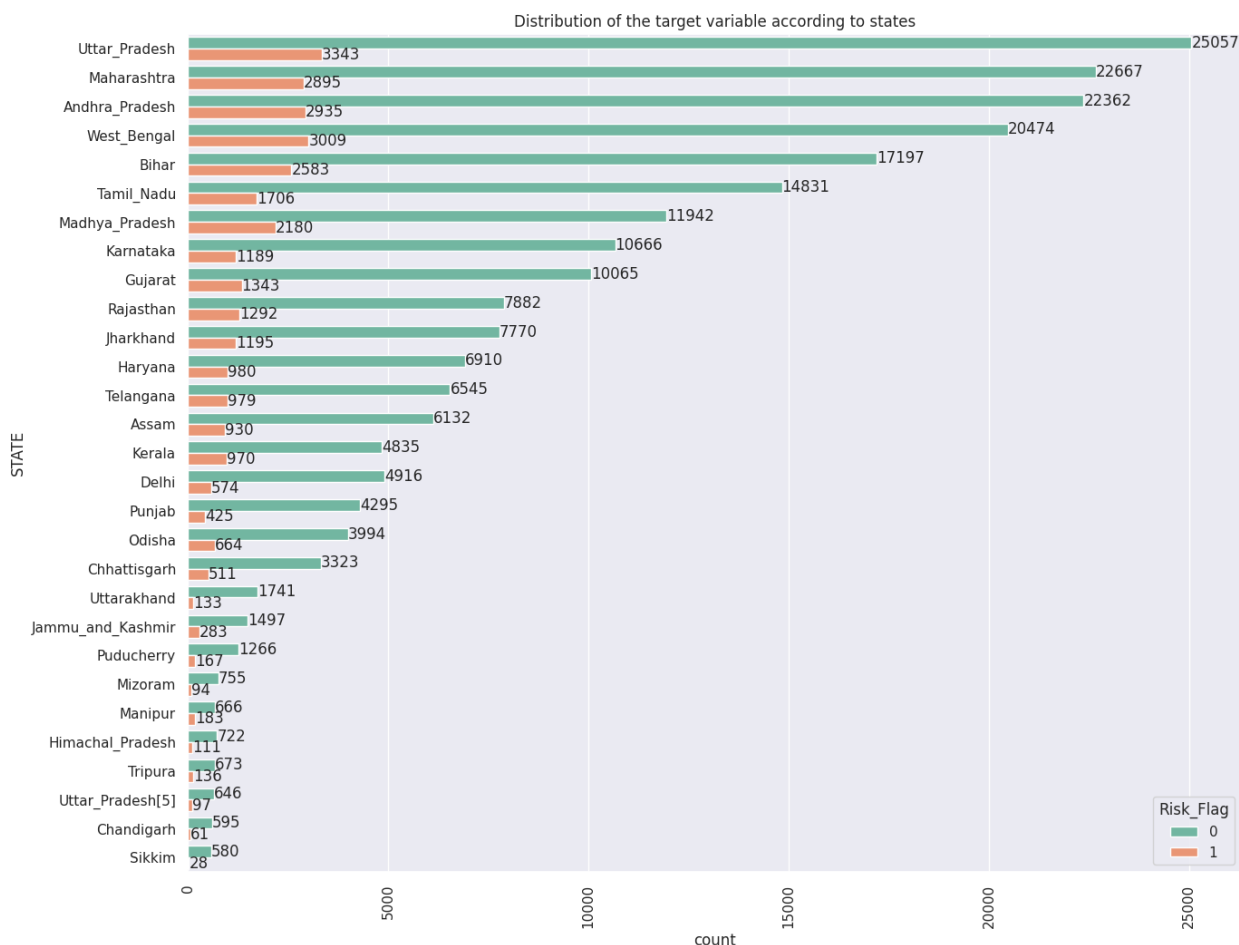
Có sự khác biệt đáng kể về tỷ lệ rủi ro giữa các tiểu bang và thành phố khác nhau. Sự đa dạng này có thể phản ánh các điều kiện kinh tế, cơ hội việc làm hoặc chính sách khu vực khác nhau ảnh hưởng đến sự ổn định tài chính và rủi ro.

❖ ***Ý nghĩa đối với chính sách và kinh doanh:***

Các khu vực có rủi ro cao có thể yêu cầu giáo dục tài chính có mục tiêu, các chương trình phát triển kinh tế hoặc các sản phẩm cho vay và bảo hiểm chuyên biệt để giải quyết các yếu tố rủi ro cơ bản.

Ngược lại, các khu vực có rủi ro thấp hơn có thể là địa điểm hấp dẫn đối với các doanh nghiệp do có cơ sở người tiêu dùng ổn định hơn.

2.1.6. Đặc trưng bang người dùng sinh sống



Hình 5: Biểu đồ cột giữa đặc trưng STATE và Risk Flag

❖ Phân phối theo tiểu bang:

Uttar Pradesh, Maharashtra và Andhra Pradesh có số lượng cá thể cao nhất trong tập dữ liệu, cho thấy các bang này có số lượng đại diện lớn hơn trong dữ liệu.

Các bang nhỏ hơn như Sikkim và Chandigarh có số lượng thấp nhất, phản ánh dân số nhỏ hơn hoặc số lượng đại diện ít hơn trong bộ dữ liệu này.

❖ *Phân phối cờ rủi ro:*

Ở hầu hết mọi tiểu bang, số người không có rủi ro (Cờ rủi ro = 0) đông hơn đáng kể số người có rủi ro (Cờ rủi ro = 1). Điều này cho thấy xu hướng chung là rủi ro thấp ở hầu hết các bang.

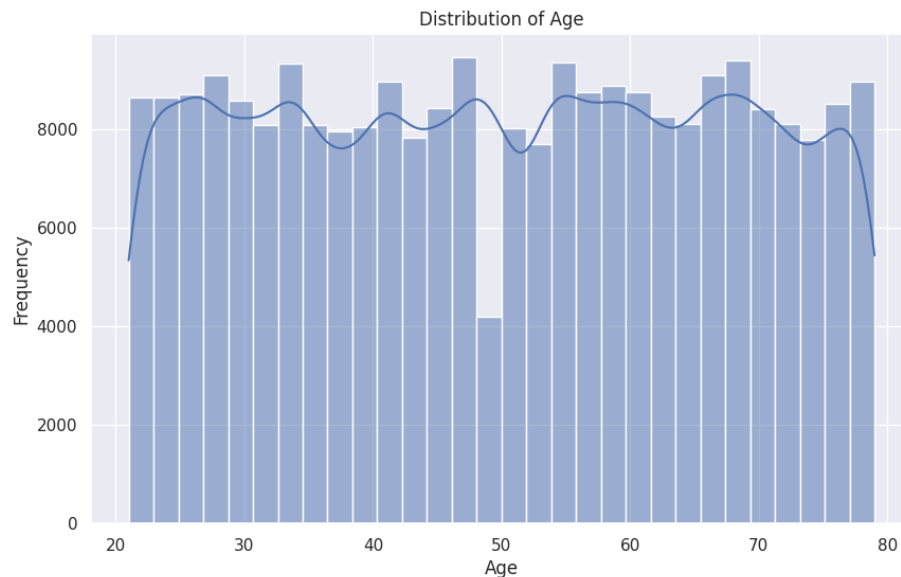
Các bang có số lượng cá nhân gắn cờ rủi ro cao nhất bao gồm Uttar Pradesh, Maharashtra, Andhra Pradesh, Tây Bengal và Bihar. Điều này có thể chỉ ra các yếu tố khu vực cụ thể góp phần tạo nên mức độ rủi ro cao hơn ở những khu vực này.

❖ *Phân tích so sánh:*

Tỷ lệ cờ rủi ro thay đổi tùy theo tiểu bang. Ví dụ, Bihar và Odisha có tỷ lệ cá nhân bị gắn cờ rủi ro tương đối cao hơn so với các bang như Kerala và Delhi. Điều này có thể là do các điều kiện kinh tế xã hội khác nhau, cơ hội việc làm hoặc các chính sách khu vực ảnh hưởng đến sự ổn định tài chính.

Các bang như Kerala, Punjab và Haryana có tỷ lệ rủi ro tương đối thấp hơn, cho thấy điều kiện kinh tế ổn định hơn hoặc các yếu tố giảm thiểu rủi ro tốt hơn.

2.1.7. *Đặc trưng về tuổi tác*



Hình 6: Biểu đồ tần suất về tuổi tác (Age)

❖ *Phân bố theo độ tuổi:*

Sự phân bố độ tuổi dường như tương đối đồng đều giữa các nhóm tuổi khác nhau, cho thấy sự thể hiện độ tuổi đa dạng trong tập dữ liệu.

Có một khoảng cách đáng chú ý ở độ tuổi 50, cho thấy có thể có sự bất thường về dữ liệu hoặc sự thiếu hiểu biết về các cá nhân trong nhóm tuổi này.

❖ *Thời kỳ đỉnh cao:*

Một số nhóm tuổi có tần suất cao hơn một chút, biểu thị các nhóm cá nhân ở những độ tuổi này. Ví dụ: độ tuổi khoảng 30, 40 và 60 dường như có số lượng cao hơn.

❖ *KDE (Ước tính mật độ hạt nhân):*

Dòng KDE cung cấp ước tính mịn về phân bố độ tuổi. Nó khẳng định sự phân bố tương đối đồng đều về độ tuổi với một số biến động nhỏ.

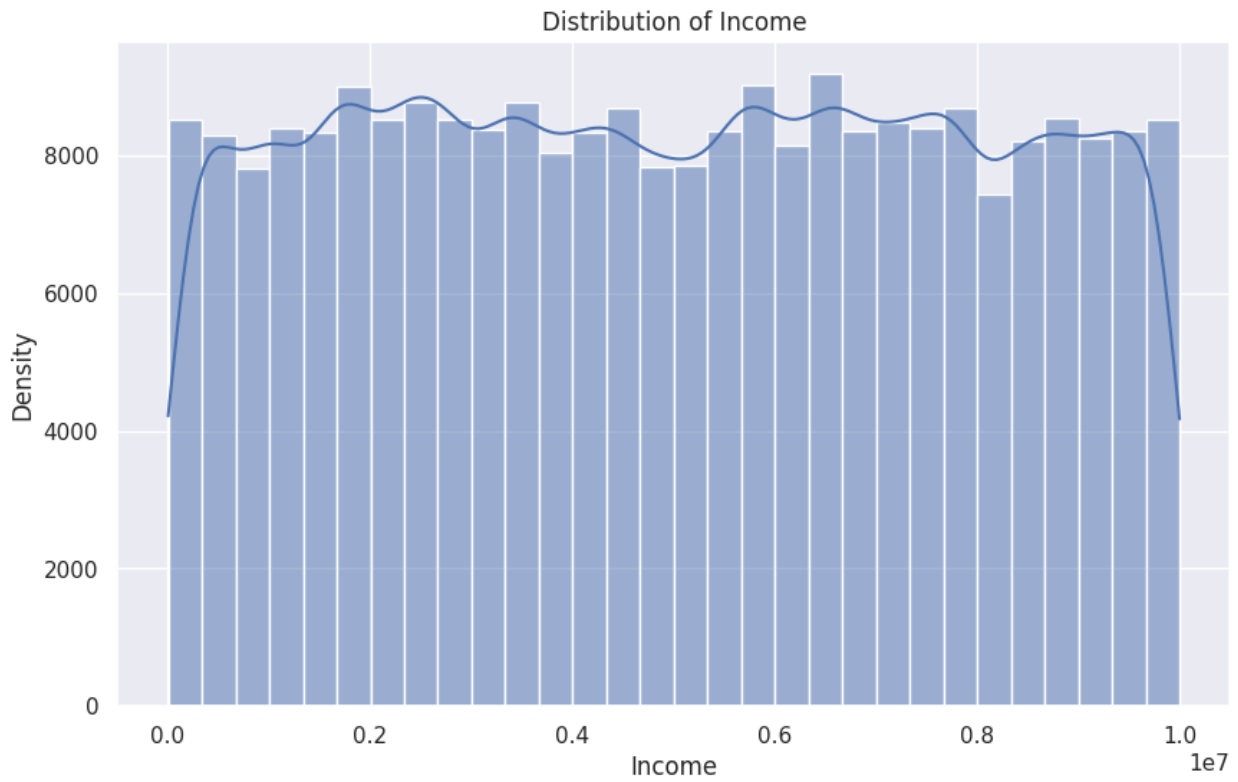
Đường KDE cũng làm nổi bật mức giảm ở độ tuổi 50, phù hợp với khoảng trống quan sát được trong các thanh biểu đồ.

❖ *Quan sát chung:*

Sự phân bố tổng thể cho thấy rằng tập dữ liệu không nghiêng nhiều về một nhóm tuổi cụ thể, khiến nó trở thành một đại diện tốt để phân tích xu hướng ở các độ tuổi khác nhau.

Các khoảng cách ở độ tuổi 50 có thể cần điều tra thêm để hiểu liệu đó có phải là do vấn đề thu thập dữ liệu hay các yếu tố cơ bản khác hay không.

2.1.8. Đặc trưng về thu nhập



Hình 7: Biểu đồ tần suất về thu nhập (Income)

❖ Phân phối thu nhập:

Sự phân bố thu nhập dường như khá đồng đều giữa các mức thu nhập khác nhau, cho thấy rằng tập dữ liệu bao gồm nhiều mức thu nhập khác nhau.

Không có đỉnh hoặc đáy đáng kể trong phân bố, cho thấy dữ liệu thu nhập được trải đều và không bị lệch nhiều về bất kỳ khung thu nhập cụ thể nào.

❖ KDE (Ước tính mật độ hạt nhân):

Đường KDE cung cấp ước tính mịn về phân phối thu nhập. Nó xác nhận sự phân phối tương đối đồng đều với những biến động nhỏ.

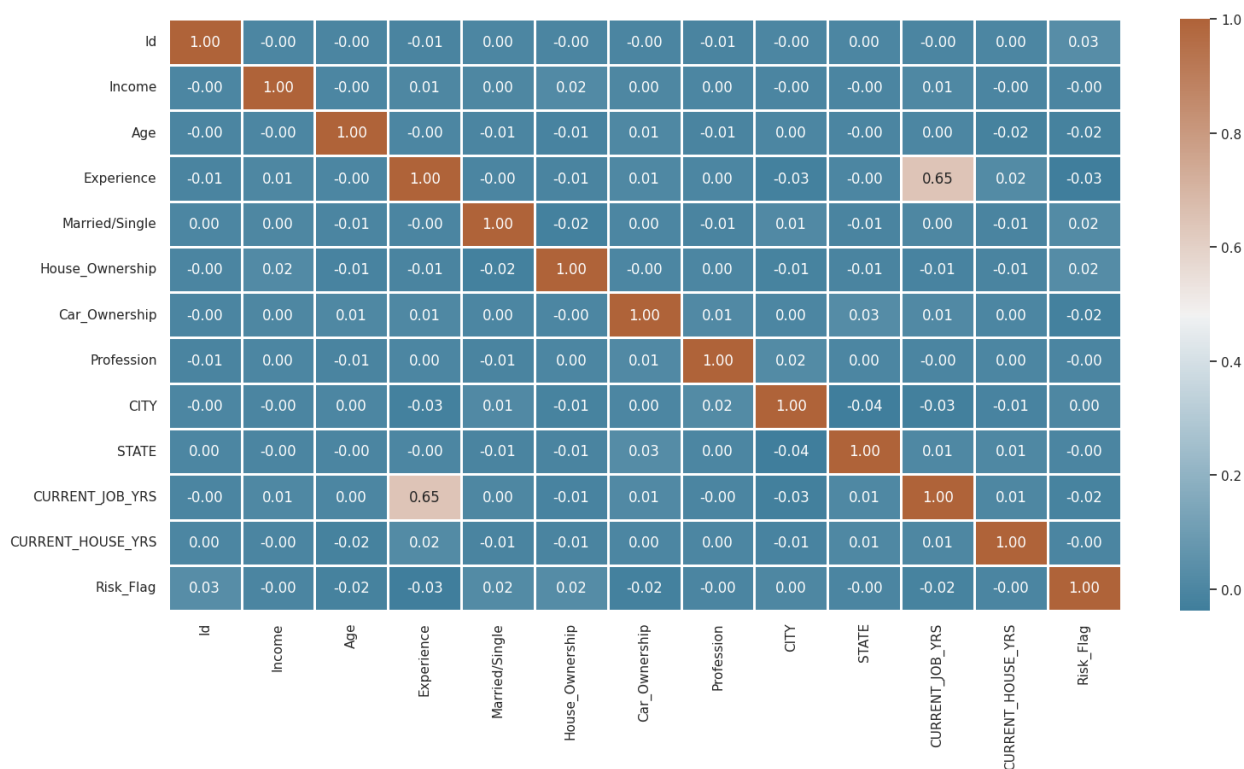
Đường KDE cũng nêu bật sự gia tăng nhẹ về mật độ ở cả đầu dưới và đầu cao hơn của phổ thu nhập, điều này cho thấy sự tập trung nhỏ của các cá nhân có thu nhập rất thấp hoặc rất cao.

❖ Quan sát chung:

Sự phân bố tổng thể cho thấy rằng tập dữ liệu đại diện cho một nhóm cá nhân đa dạng với mức thu nhập khác nhau, khiến nó phù hợp cho các phân tích yêu cầu nhiều dữ liệu thu nhập.

Sự phân bố đồng đều cũng chỉ ra rằng tập dữ liệu không có độ lệch thu nhập đáng kể, điều này có lợi cho việc phân tích và lập mô hình khách quan.

2.2. Phân tích mối tương quan giữa các biến



Hình 8: Biểu đồ tương quan giữa các biến

❖ Quan sát chung:

Hầu hết các biến thể hiện mối tương quan rất thấp hoặc không có mối tương quan với nhau, được biểu thị bằng các giá trị gần bằng 0. Điều này cho thấy các đặc điểm tương đối độc lập với nhau.

❖ ***Tương quan mạnh mẽ:***

Mối tương quan mạnh nhất được quan sát là giữa Kinh nghiệm và CURRENT_JOB_YRS (0,65). Điều này có ý nghĩa vì số năm kinh nghiệm của một cá nhân có thể liên quan chặt chẽ đến số năm họ đã làm trong công việc hiện tại.

Một mối tương quan tương đối cao khác được thấy giữa CURRENT_JOB_YRS và Tuổi (0,20), điều này được mong đợi vì những người lớn tuổi có thể có nhiều năm kinh nghiệm làm việc hơn.

❖ ***Tương quan Risk Flag:***

Risk_Flag cho thấy mối tương quan rất thấp với tất cả các tính năng khác, trong đó cao nhất là mối tương quan yếu 0,03 với Id. Điều này cho thấy rằng các đặc điểm trong tập dữ liệu không ảnh hưởng mạnh đến cờ rủi ro hoặc các tương tác phức tạp hơn có thể ảnh hưởng đến rủi ro không được nắm bắt bằng tuyến tính đơn giản. các mối tương quan.

❖ ***Không có mối tương quan tiêu cực đáng chú ý:***

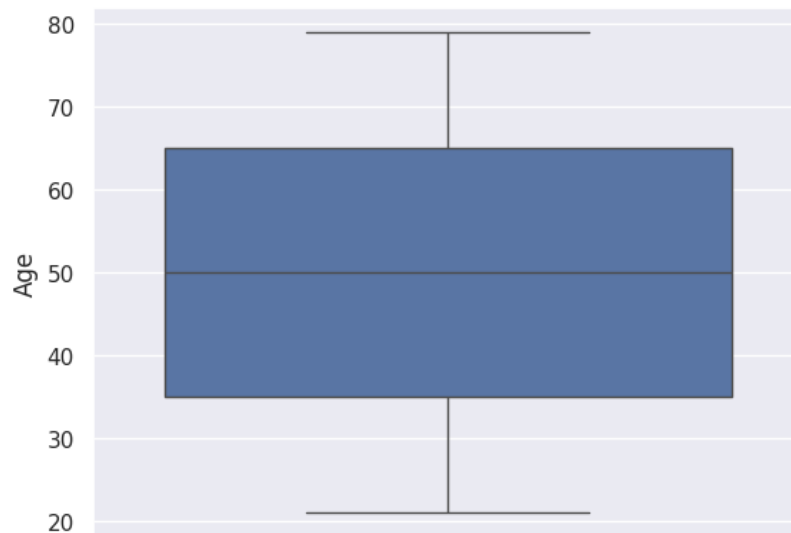
Không có mối tương quan tiêu cực đáng kể nào được quan sát trong tập dữ liệu, chỉ ra rằng các tính năng không có mối quan hệ nghịch đảo với nhau một cách có ý nghĩa.

❖ ***Độc lập của các biến phân loại:***

Các biến phân loại như Married/Single, House_Ownership, Car_Ownership, Profession, CITY, and STATE cho thấy mối tương quan rất thấp với các biến khác, cho thấy tính độc lập của chúng.

2.3. Xử lý ngoại lai

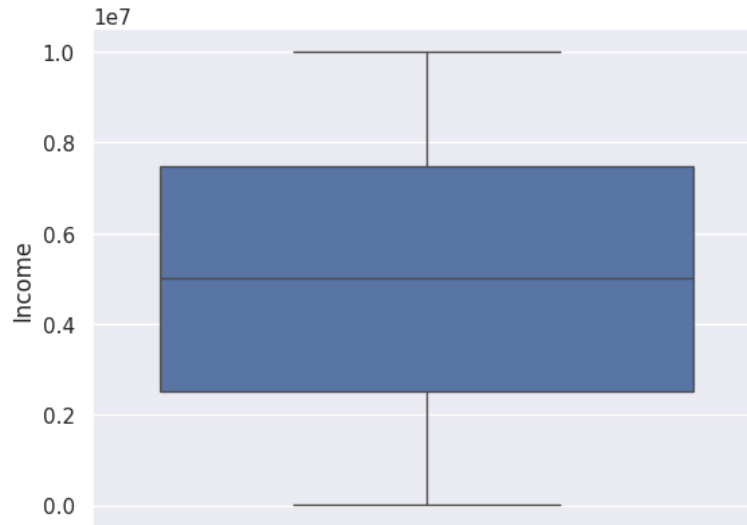
2.3.1. Xử lý ngoại lai của đặc trưng Tuổi



Hình 9: Biểu đồ hộp về đặc trưng tuổi (Age)

Không có điểm dữ liệu nào được biểu diễn như ngoại lệ trên biểu đồ, điều này cho thấy không có giá trị tuổi nào quá cao hoặc quá thấp so với phần còn lại của nhóm dữ liệu.

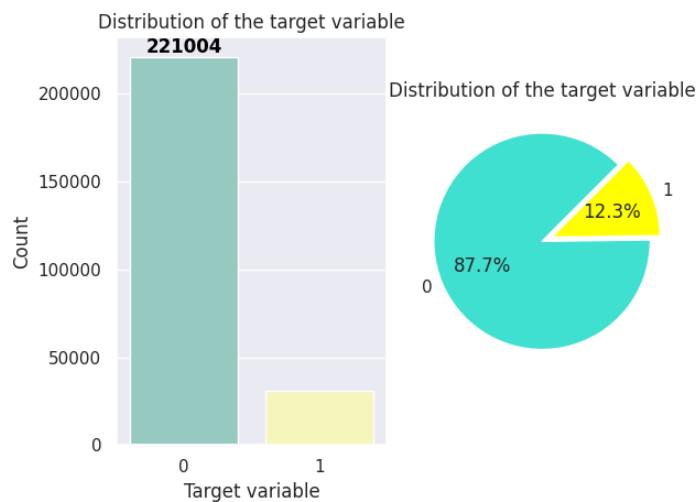
2.3.2. Xử lý ngoại lai của đặc trưng Income



Hình 10: Biểu đồ hộp của đặc trưng thu nhập (Income)

Không có điểm ngoại lai nào được hiển thị, dữ liệu có thể rất đồng đều và không có giá trị nào quá cao hoặc quá thấp so với phần còn lại của tập dữ liệu

2.4. Trực quan cung cấp cái nhìn rõ ràng về sự phân bố của biến mục tiêu, Risk_Flag



Hình 11: Biểu đồ cột và biểu đồ tròn về sự phân bố của biến mục tiêu Risk_Flag

❖ Tóm tắt về trực quan hóa dữ liệu

- Lớp 0 chiếm 88% tập dữ liệu, trong khi lớp 1 chỉ chiếm 12%. Các lớp bị sai lệch rất nhiều, chúng ta cần giải quyết vấn đề này
- Không có ngoại lệ trong tập dữ liệu. Nhưng chúng ta cần chuẩn hóa Age và Income
- Mỗi tương quan chặt chẽ giữa Experience và CURRENT_JOB_YRS. Có thể bỏ một cột trong quá trình lựa chọn tính năng hoặc sử dụng Phân tích thành phần chính (PCA)
- Married/Single House_Ownership Car_Ownership có thể được mã hóa nhị phân hoặc dùng mã hóa one-hot
- Có thể tìm mối liên hệ giữa biến mục tiêu và biến phân loại bằng phép kiểm Chi bình phương

CHƯƠNG 3: PHÂN TÍCH THỐNG KÊ VÀ KIỂM ĐỊNH GIẢ THUYẾT

3.1. Phương pháp thống kê sử dụng

Phương pháp chi bình phương (Chi-Square)

Kiểm định chi bình phương (còn gọi là kiểm định chi bình phương hoặc kiểm định χ^2) là một kiểm định giả thuyết thống kê được sử dụng trong phân tích các bảng dự phòng khi cỡ mẫu lớn. Nói một cách đơn giản hơn, thử nghiệm này chủ yếu được sử dụng để kiểm tra xem hai biến phân loại (hai chiều của bảng dự phòng) có độc lập trong việc ảnh hưởng đến thống kê thử nghiệm hay không (các giá trị trong bảng).

Công thức:

$$\chi^2 = \sum \frac{(O - E)^2}{E}$$

Trong đó:

- O là tần số quan sát, tần số thực sự thu được từ mẫu ngẫu nhiên
- E là tần số kỳ vọng, tần số dự đoán khi giả định hai biến độc lập nhau
- Có hai giả thuyết kiểm định Chi-Square:
- H_0 : hai biến thống kê độc lập với nhau
- H_1 : hai biến thống kê phụ thuộc với nhau

3.2. Phân tích và kiểm định giả thuyết

Giả thuyết kiểm định khả năng vỡ nợ theo sở hữu xe

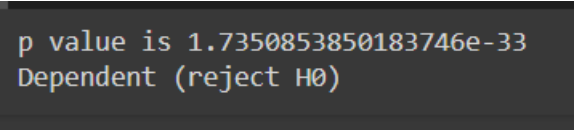
Mục đích: Kiểm định sự khác biệt về khả năng vỡ nợ giữa những người sở hữu xe và những người không sở hữu xe, giúp hiểu rõ hơn về ảnh hưởng của việc có xe đến khả năng vỡ nợ.

Nhận thức vấn đề: Nhận định ban đầu về giả thuyết là người không sở hữu xe có rủi ro vỡ nợ cao hơn. Chi-Square test (kiểm định chi bình phương) được áp dụng để kiểm tra giả thuyết này, có hay không sự khác biệt về khả năng vỡ nợ giữa hai nhóm sở hữu xe và không sở hữu xe.

+ Giả thuyết Không (H_0): Phản ánh giả định rằng không có sự khác biệt về khả năng vỡ nợ giữa hai nhóm, hoặc khả năng vỡ nợ của nhóm không sở hữu xe không cao hơn nhóm sở hữu xe (có khoảng tin cậy là 95% và mức ý nghĩa là 5%).

+ Giả thuyết Ngược (H_1): Có sự khác biệt đáng kể về khả năng vỡ nợ giữa hai nhóm người.

Giả định rằng các phương sai của hai nhóm là không đồng nhất (điều này thường phù hợp hơn khi làm việc với dữ liệu thực tế) để thực hiện việc kiểm định dựa trên các giá trị của hai nhóm mẫu. Ta sẽ đi sâu vào phân tích thống kê đã được tiến hành để so sánh khả năng vỡ nợ giữa người sở hữu xe và không sở hữu xe, sử dụng kiểm định chi bình phương:



p value is 1.7350853850183746e-33
Dependent (reject H_0)

Hình 12: Kết quả kiểm định khả năng vỡ nợ theo sở hữu xe

Nhận xét: Kết quả của kiểm định Chi-Square cho thấy sự chênh lệch đáng kể, với p-value < mức ý nghĩa, bác bỏ H_0 cho thấy sự khác biệt về sở hữu xe giữa hai nhóm là có ý nghĩa thống kê.

Giả thuyết kiểm định khả năng vỡ nợ theo trình trạng hôn nhân

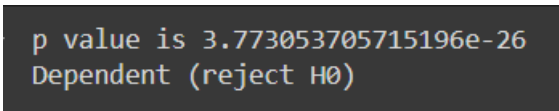
Mục đích: Kiểm định sự khác biệt về khả năng vỡ nợ giữa những người độc thân và đã kết hôn, giúp hiểu rõ hơn về ảnh hưởng của tình trạng hôn nhân đến khả năng vỡ nợ.

Nhận thức vấn đề: Nhận định ban đầu về giả thuyết là người độc thân có rủi ro vỡ nợ cao hơn. Chi-Square test (kiểm định chi bình phương) được áp dụng để kiểm tra giả thuyết này, có hay không sự khác biệt về khả năng vỡ nợ giữa hai nhóm độc thân và đã kết hôn.

+ Giả thuyết Không (H_0): Phản ánh giả định rằng không có sự khác biệt về khả năng vỡ nợ giữa hai nhóm, hoặc khả năng vỡ nợ của nhóm độc thân không cao hơn nhóm đã kết hôn (có khoảng tin cậy là 95% và mức ý nghĩa là 5%).

+ Giả thuyết Nghịch (H_1): Có sự khác biệt đáng kể về khả năng vỡ nợ giữa hai nhóm người.

Giả định rằng các phương sai của hai nhóm là không đồng nhất (điều này thường phù hợp hơn khi làm việc với dữ liệu thực tế) để thực hiện việc kiểm định dựa trên các giá trị của hai nhóm mẫu. Ta sẽ đi sâu vào phân tích thống kê đã được tiến hành để so sánh khả năng vỡ nợ giữa người độc thân và đã kết hôn, sử dụng kiểm định chi bình phương:



p value is 3.773053705715196e-26
Dependent (reject H_0)

Hình 13: Kết quả kiểm định khả năng vỡ nợ theo tình trạng hôn nhân

Nhận xét: Kết quả của kiểm định Chi-Square cho thấy sự chênh lệch đáng kể, với $p\text{-value} < \text{mức ý nghĩ}$, bác bỏ H_0 cho thấy sự khác biệt về tình trạng hôn nhân giữa hai nhóm là có ý nghĩa thống kê.

Giả thuyết kiểm định khả năng vỡ nợ theo sở hữu nhà

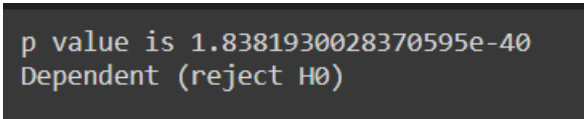
Mục đích: Kiểm định sự khác biệt về khả năng vỡ nợ giữa những người sở hữu nhà, người thuê nhà, người không có nhà và không thuê nhà, giúp hiểu rõ hơn về ảnh hưởng của việc sở hữu nhà đến khả năng vỡ nợ.

Nhận thức vấn đề: Nhận định ban đầu về giả thuyết là người thuê nhà có rủi ro vỡ nợ cao hơn. Chi-Square test (kiểm định chi bình phương) được áp dụng để kiểm tra giả thuyết này, có hay không sự khác biệt về khả năng vỡ nợ giữa các nhóm người thuê nhà, người sở hữu nhà và người không thuê không sở hữu.

+ Giả thuyết Không (H_0): Phản ánh giả định rằng không có sự khác biệt về khả năng vỡ nợ giữa các nhóm, hoặc khả năng vỡ nợ của nhóm thuê nhà không cao hơn nhóm còn lại (có khoảng tin cậy là 95% và mức ý nghĩa là 5%).

+ Giả thuyết Nghịch (H_1): Có sự khác biệt đáng kể về khả năng vỡ nợ giữa các nhóm người.

Giả định rằng các phương sai của hai nhóm là không đồng nhất (điều này thường phù hợp hơn khi làm việc với dữ liệu thực tế) để thực hiện việc kiểm định dựa trên các giá trị của hai nhóm mẫu. Ta sẽ đi sâu vào phân tích thống kê đã được tiến hành để so sánh khả năng vỡ nợ giữa những người sở hữu nhà, người thuê nhà, người không có nhà và không thuê nhà, sử dụng kiểm định chi bình phương:



p value is 1.8381930028370595e-40
Dependent (reject H_0)

Hình 14: Kết quả kiểm định khả năng vỡ nợ theo sở hữu nhà

Nhận xét: Kết quả của kiểm định Chi-Square cho thấy sự chênh lệch đáng kể, với p-value < mức ý nghĩa, bác bỏ H_0 cho thấy sự khác biệt về việc sở hữu nhà giữa các nhóm là có ý nghĩa thống kê.

3.3. Đánh giá mức độ tin cậy của kết quả

Việc kiểm định thống kê đã cung cấp bằng chứng rõ ràng về sự khác biệt trong mối quan hệ giữa các nhóm người:

Nhóm người theo sở hữu nhà: có vẻ như những người không thuê nhà hoặc không sở hữu và thuê có tỷ lệ vỡ nợ cao hơn so với những người sở hữu nhà. Điều này có thể phản ánh khả năng tài chính ổn định hơn của những người sở hữu nhà

Nhóm người theo sở hữu xe: có thể thấy rằng người không sở hữu xe có tỷ lệ vỡ nợ cao hơn người sở hữu xe.

Nhóm người theo tình trạng hôn nhân: số liệu cho thấy có sự khác biệt rõ khi người độc thân có tỷ lệ vỡ nợ cao hơn người đã kết hôn.

⇒ Các kết quả có độ tin cậy cao. Giả thuyết kiểm định cho thấy rằng các yếu tố như sở hữu nhà, tình trạng hôn nhân, và sở hữu xe có thể có ảnh hưởng đáng kể đến khả năng vỡ nợ của một cá nhân. Việc phân tích sâu hơn các yếu tố này có thể giúp các tổ chức tài chính đưa ra quyết định vay vốn một cách chính xác hơn và giảm thiểu rủi ro vỡ nợ.

CHƯƠNG 4: MÔ HÌNH HÓA VÀ DỰ ĐOÁN

4.1. Chuẩn hóa và chuẩn bị dữ liệu huấn luyện

4.1.1. Phân tích thành phần chính (PCA) cho CURRENT_JOB_YRS và Experience

PCA (Principal Component Analysis) là một kỹ thuật giảm số chiều phổ biến trong học máy và phân tích dữ liệu. PCA được sử dụng để giảm số lượng các biến số trong dữ liệu xuống thành một tập hợp các biến số nhỏ hơn mà vẫn giữ lại được phần lớn thông tin của dữ liệu gốc. Mục tiêu chính của PCA là giảm số chiều dữ liệu, loại bỏ nhiễu và giúp việc trực quan hóa dữ liệu dễ dàng hơn.

```
features = ["CURRENT_JOB_YRS", "Experience"]

df_for_pca = df[features]
scaled_df_for_pca = (df_for_pca -
df_for_pca.mean(axis=0))/df_for_pca.std()
scaled_df_for_pca
```

	CURRENT_JOB_YRS	Experience
0	-0.914129	-1.180230
1	0.731035	-0.014067
2	-0.639935	-1.013635
3	-1.188323	-1.346825
4	-0.914129	0.152528
...
251995	-0.091547	0.485718
251996	-0.091547	-0.014067
251997	0.182647	-0.513851
251998	-1.736711	-1.680014
251999	0.182647	1.152097

252000 rows × 2 columns

Hình 15: DataFrame đã chuẩn hóa cho PCA

	PC1	PC2
0	-1.480935	-0.188162
1	0.506973	-0.526866
2	-1.169251	-0.264246
3	-1.792620	-0.112078
4	-0.538533	0.754240

Hình 16: DataFrame sau khi áp dụng PCA

```
df1 = pd.concat([df, df_pca], axis=1)
df1.head()
```

	Id	Income	Age	Experience	Married/Single	House_Ownership	Car_Ownership	Profession	CITY	STATE	CURRENT_JOB_YRS	CURRENT_HOUSE_YRS	Risk_Flag	PC1	PC2
0	1	1303834	23	3	single	rented	no	Mechanical_Engineer	Rewa	Madhya_Pradesh	3	13	0	-1.480935	-0.188162
1	2	7574516	40	10	single	rented	no	Software_Developer	Parbhani	Maharashtra	9	13	0	0.506973	-0.526866
2	3	3991815	66	4	married	rented	no	Technical_writer	Alappuzha	Kerala	4	10	0	-1.169251	-0.264246
3	4	6256451	41	2	single	rented	yes	Software_Developer	Bhubaneswar	Odisha	2	12	1	-1.792620	-0.112078
4	5	5768871	47	11	single	rented	no	Civil_servant	Tiruchirappalli[10]	Tamil_Nadu	3	14	1	-0.538533	0.754240

Hình 17: Nối DataFrame PCA với DataFrame ban đầu

4.1.2. Chuẩn bị dữ liệu

4.1.2.1. Mã hóa nhãn cho các biến phân loại

Các mô hình học máy thường yêu cầu đầu vào là các giá trị số và không thể làm việc trực tiếp với các giá trị phân loại dạng chữ hoặc ký hiệu, do đó ta cần đưa về dạng số.

```
features =
['Married/Single', 'Car_Ownership', 'Profession', 'CITY', 'STATE']
label_encoder = LabelEncoder()

for col in features:
    df1[col] = label_encoder.fit_transform(df1[col])

df2 = pd.get_dummies(df1, columns = ["House_Ownership"])
```

4.1.2.2. Loại bỏ cột dư thừa Id

```
df2.drop(["Id"], axis=1, inplace=True)
df2
```

	Income	Age	Experience	Married/Single	Car_Ownership	Profession	CITY	STATE	CURRENT_30B_YRS	CURRENT_HOUSE_YRS	Risk_Flag	PC1	PC2	House_Ownership_norent_noown	House_Ownership_owed	House_Ownership_rented
0	1303834	23	3	1	0	33	251	13	3	13	0	-1.480935	-0.188162	False	False	True
1	7574516	40	10	1	0	43	227	14	9	13	0	0.506973	-0.526866	False	False	True
2	3991815	66	4	0	0	47	8	12	4	10	0	-1.169251	-0.264246	False	False	True
3	6256451	41	2	1	1	43	54	17	2	12	1	-1.792620	-0.112078	False	False	True
4	5768871	47	11	1	0	11	296	22	3	14	1	-0.538533	0.754240	False	False	True
...
251995	8154883	43	13	1	0	45	162	28	6	11	0	0.278721	0.408188	False	False	True
251996	2843572	26	10	1	0	3	251	13	6	11	0	-0.074680	0.054787	False	False	True
251997	4522448	46	7	1	0	17	144	14	7	12	0	-0.234197	-0.492498	False	False	True
251998	6507128	45	0	1	0	27	233	18	0	10	0	-2.415990	0.040091	False	False	True
251999	9070230	70	17	1	0	44	26	22	7	11	0	0.943806	0.685504	False	False	True

252000 rows x 16 columns

Hình 18: DataFrame sau khi mã hóa các biến phân loại và loại bỏ cột Id

1.1.1. Chia dữ liệu

1.1.1.1. Loại bỏ thuộc tính đích và chia tập dữ liệu

```
X = df2.drop(['Risk_Flag'], axis=1)
y = df2.Risk_Flag
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =
0.3, random_state = 0)
```

1.1.1.2. Xử lý mất cân bằng dữ liệu

Khi dữ liệu huấn luyện bị mất cân bằng, tức là số lượng các mẫu trong các lớp không đều nhau, mô hình học máy có thể gặp nhiều vấn đề dẫn đến hiệu suất kém. Kỹ thuật SMOTE (Synthetic Minority Over-sampling Technique) là một phương pháp tạo ra các mẫu tổng hợp mới từ các mẫu hiện có của lớp thiểu số, giúp nâng cao hiệu suất và độ tin cậy của mô hình trong các ứng dụng thực tế.

```
sm = SMOTE(random_state = 500)
X_res, y_res = sm.fit_resample(X_train, y_train)
```

4.2. Mô hình logictis regresion

4.2.1. Sơ lược về mô hình

Logistic Regression là thuật toán học máy được sử dụng cho bài toán phân loại nhị phân (chỉ có 2 lớp). Khác với hồi quy tuyến tính, Logistic Regression xây dựng một hàm

sigmoid để ước tính xác suất của một mẫu dữ liệu thuộc về lớp nào. Thuật toán này có ưu điểm là dễ hiểu, dễ triển khai và hiệu quả với dữ liệu tuyến tính. Tuy nhiên, Logistic Regression chỉ xử lý tốt dữ liệu tuyến tính và cần chuẩn hóa dữ liệu đầu vào trước khi sử dụng.

4.2.2. Đào tạo và đánh giá mô hình

```
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from imblearn.over_sampling import SMOTE

# Chuẩn hóa dữ liệu
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_res)
X_test_scaled = scaler.transform(X_test)

model = LogisticRegression(max_iter = 500000, C = 0.01, penalty = 'l2',
solver = 'sag')
model.fit(X_train_scaled, y_res)
y_pred = model.predict(X_test_scaled)
accuracy = model.score(X_test_scaled, y_test)
accuracy

print(classification_report(y_test, y_pred))
```

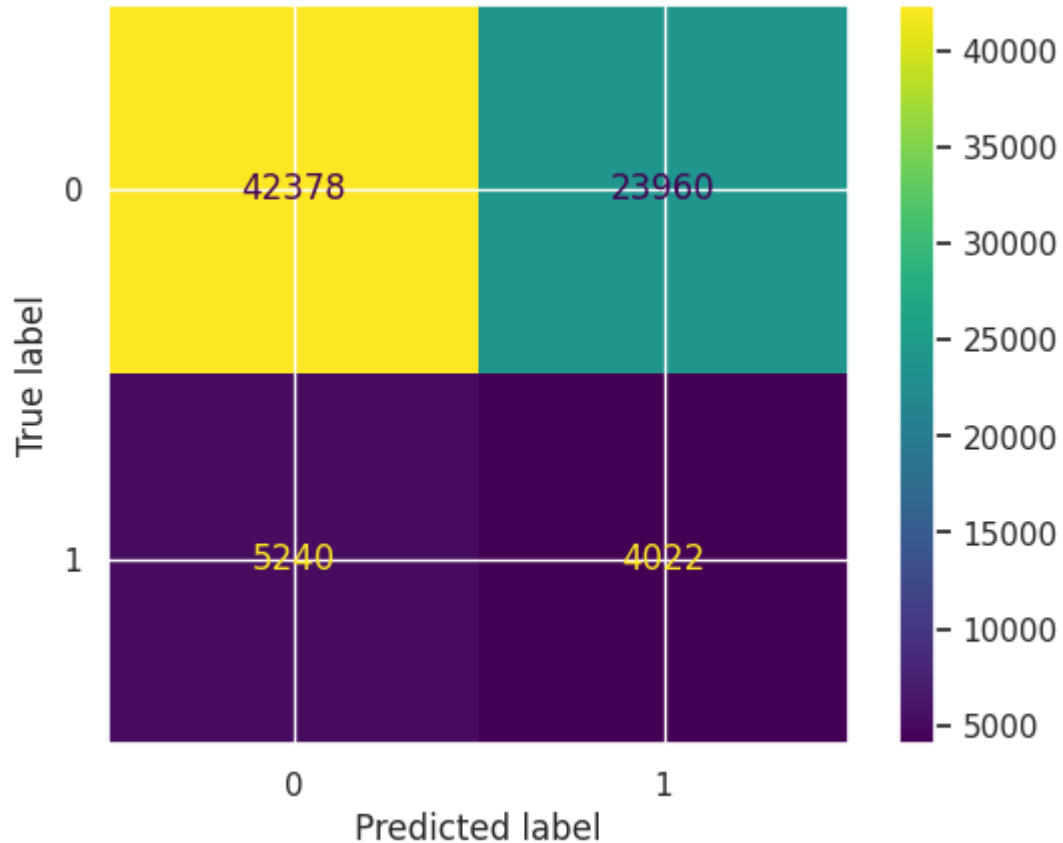
	precision	recall	f1-score	support
0	0.89	0.64	0.74	66338
1	0.14	0.44	0.22	9262
accuracy			0.61	75600
macro avg	0.52	0.54	0.48	75600
weighted avg	0.80	0.61	0.68	75600

Hình 19: Báo cáo các độ đo của Logistic Regression

4.2.3. Đánh giá kết quả và giải thích mô hình

```
from sklearn.metrics import ConfusionMatrixDisplay
import matplotlib.pyplot as plt
%matplotlib inline
```

```
con_max = ConfusionMatrixDisplay.from_predictions(y_test, y_pred)
plt.show()
```



Hình 20: Confusion matrix cho dự đoán của Logistic Regression

- **Lớp 0 (Không vỡ nợ):**
 - **Precision (0.89):** Mô hình có độ chính xác cao khi dự đoán khách hàng không vỡ nợ, với 89% dự đoán là chính xác.
 - **Recall (0.64):** Tuy nhiên, chỉ có 64% khách hàng thực sự không vỡ nợ được mô hình nhận diện chính xác.
 - **F1-score (0.74):** Điểm F1 cho thấy mô hình có hiệu suất tốt nhưng vẫn còn không gian để cải thiện, đặc biệt là về mặt recall.
- **Lớp 1 (Vỡ nợ):**

- **Precision thấp (0.14):** Chỉ có 14% trong số những dự đoán vỡ nợ của mô hình là chính xác, cho thấy nhiều báo động giả.
- **Recall (0.44):** Mô hình có khả năng nhận diện 44% khách hàng thực sự vỡ nợ.
- **F1-score (0.22):** Điểm F1 thấp cho thấy mô hình không hiệu quả trong việc dự đoán lớp vỡ nợ, chủ yếu do precision rất thấp.
- **Accuracy tổng thể (0.61):** Mô hình chính xác 61% trên toàn bộ dữ liệu kiểm tra.

Nhận xét chung:

- Mô hình này có vẻ tốt hơn trong việc nhận diện khách hàng không vỡ nợ so với việc nhận diện khách hàng vỡ nợ.
- Precision thấp cho lớp vỡ nợ có thể dẫn đến việc từ chối tín dụng cho nhiều khách hàng tiềm năng không vỡ nợ.
- Recall tương đối thấp cho cả hai lớp cho thấy mô hình có thể bỏ qua một số trường hợp thực sự vỡ nợ hoặc không vỡ nợ.

4.3. Mô hình KNN

4.3.1. Sơ lược về mô hình

KNN (viết tắt của K-Nearest Neighbors, hay thuật toán k láng giềng gần nhất) là thuật toán học máy đơn giản và linh hoạt, có thể được sử dụng cho cả bài toán phân loại và hồi quy. KNN hoạt động bằng cách dự đoán dựa trên các điểm gần nhất (k điểm) trong tập huấn luyện, sau đó dự đoán nhãn (phân loại) hoặc giá trị trung bình (hồi quy) cho mẫu dữ liệu mới dựa trên nhãn hoặc giá trị trung bình của k láng giềng gần nhất. Ưu điểm của KNN là dễ hiểu, không cần giả định về phân bố dữ liệu và hiệu quả với nhiều dạng dữ liệu. Tuy nhiên, KNN cũng có một số nhược điểm như: nhạy cảm với nhiễu trong dữ liệu, cần chọn giá trị k phù hợp và tốn thời gian tính toán khi xử lý tập dữ liệu lớn.

4.3.2. Đào tạo và đánh giá mô hình

```
from sklearn.neighbors import KNeighborsClassifier
```

```

param_grid = {
    'n_neighbors': [3, 5, 7, 9],
    'p': [1, 2],
    'metric': ['minkowski', 'euclidean']
}

knn = KNeighborsClassifier()
grid_search = GridSearchCV(estimator=knn, param_grid=param_grid, cv=5,
verbose=3, scoring='accuracy')
grid_search.fit(X_res, y_res)

best_params = grid_search.best_params_
best_score = grid_search.best_score_

best_params, best_score
knn_model = KNeighborsClassifier(n_neighbors=9, metric='minkowski', p=1)
knn_model.fit(X_res, y_res)
y_pred = knn_model.predict(X_test)
accuracy = knn_model.score(X_test, y_test)
accuracy

```

4.3.3. Đánh giá kết quả và giải thích mô hình

```
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.94	0.90	0.92	66338
1	0.46	0.61	0.53	9262
accuracy			0.87	75600
macro avg	0.70	0.75	0.72	75600
weighted avg	0.88	0.87	0.87	75600

Hình 21. Báo cáo các độ đo của mô hình KNN

Nhận xét:

- Độ chính xác tổng thể cao: Mô hình KNN đạt độ chính xác tổng thể 87%, cho thấy mô hình phân loại đúng phần lớn các trường hợp.
- Precision cho lớp 1 thấp: Precision của lớp 1 (các trường hợp có rủi ro) là 0.46, nghĩa là chỉ 46% trong số các dự đoán dương tính là đúng. Điều này cho thấy mô hình có nhiều dự đoán dương tính sai.
- Recall cho lớp 1 tốt hơn: Recall của lớp 1 là 0.61, nghĩa là mô hình tìm được 61% các trường hợp có rủi ro thực sự. Điều này tốt hơn so với mô hình AdaBoost trước đó.
- F1-Score của lớp 1: F1-Score của lớp 1 là 0.53, cho thấy mô hình KNN có sự cân bằng giữa Precision và Recall tốt hơn so với AdaBoost.
- Macro và Weighted Averages: Các chỉ số trung bình (macro và weighted) cho thấy hiệu suất tổng thể của mô hình là ổn định và đáng tin cậy, đặc biệt với weighted avg do sự mất cân bằng giữa hai lớp.

4.4. Mô hình Random Forest

4.4.1. Sơ lược về mô hình

Random Forest là một phương pháp ensemble học máy mạnh mẽ dựa trên việc kết hợp nhiều cây quyết định độc lập. Mỗi cây được xây dựng trên một tập con ngẫu nhiên của dữ liệu, và các cây được huấn luyện để giảm thiểu phương sai và ngăn ngừa overfitting. Khi dự đoán, Random Forest lấy trung bình (trong trường hợp hồi quy) hoặc bỏ phiếu đa số (trong trường hợp phân loại) từ các cây riêng lẻ. Điều này giúp Random Forest có khả năng dự đoán mạnh mẽ và ổn định, nhưng mô hình có thể trở nên phức tạp và khó giải thích.

4.4.2. Đào tạo và đánh giá mô hình

Tạo mô hình RandomForestClassifier và huấn luyện, sau đó xuất ra mức độ quan trọng của thuộc tính

```
from sklearn.ensemble import RandomForestClassifier
```

```

gr_rf_model = RandomForestClassifier(class_weight="balanced",
random_state=101, max_depth=12, n_estimators= 200, min_samples_split = 3)
gr_rf_model.fit(X_train,y_train)

rf_feature_imp = pd.DataFrame(index = X.columns, data =
gr_rf_model.feature_importances_,
                             columns = ["Feature
Importance"]).sort_values("Feature Importance", ascending = False)
rf_feature_imp

```

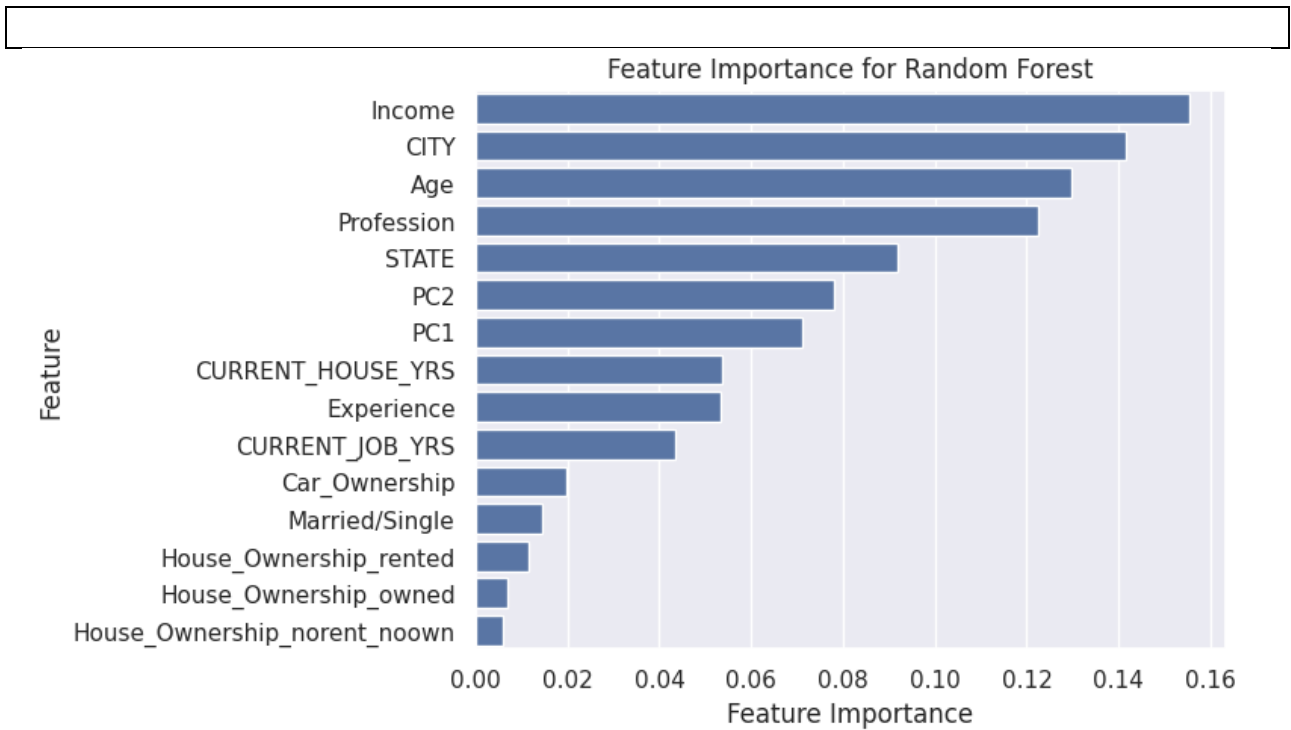
	Feature Importance
Income	0.156703
CITY	0.139608
Age	0.131028
Profession	0.121232
STATE	0.093531
PC2	0.076709
PC1	0.070287
CURRENT_HOUSE_YRS	0.054881
Experience	0.052617
CURRENT_JOB_YRS	0.046180
Car_Ownership	0.019076
Married/Single	0.014020
House_Ownership_rented	0.011038
House_Ownership_owned	0.006772
House_Ownership_norent_noown	0.006316

Hình 22. Mức độ quan trọng của các thuộc tính trong RandomForest

```

ax = sns.barplot(x=rf_feature_imp["Feature Importance"],
y=rf_feature_imp.index)
plt.ylabel("Feature")
plt.title("Feature Importance for Random Forest")
plt.show()

```



Hình 23. Biểu đồ thể hiện mức độ quan trọng của các thuộc tính trong RandomForest

Từ biểu đồ trên, chọn các thuộc tính có mức quan trọng cao và thực hiện huấn luyện lại

```
new_df = df2[["Income", "CITY", "Age", "Profession", "STATE", "PC2",
              "PC1", "CURRENT_HOUSE_YRS", "Risk_Flag"]]
X_new = new_df.drop('Risk_Flag',axis=1)
y_new = new_df['Risk_Flag']
X_train_new, X_test_new, y_train_new, y_test_new =
train_test_split(X_new, y_new, test_size = 0.2, stratify=y, random_state
= 101)

rf_model = RandomForestClassifier(class_weight="balanced",
random_state=101)
rf_model.fit(X_train_new,y_train_new)
y_pred_new = rf_model.predict(X_test_new)
accuracy = rf_model.score(X_test_new, y_test_new)
accuracy
```

4.4.3. Đánh giá kết quả và giải thích mô hình

```
print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
0	0.97	0.91	0.94	44201
1	0.56	0.77	0.64	6199
accuracy			0.90	50400
macro avg	0.76	0.84	0.79	50400
weighted avg	0.91	0.90	0.90	50400

Hình 24. Báo cáo về các độ đo của mô hình Random Forest

```
plt.figure(figsize=(10,6))

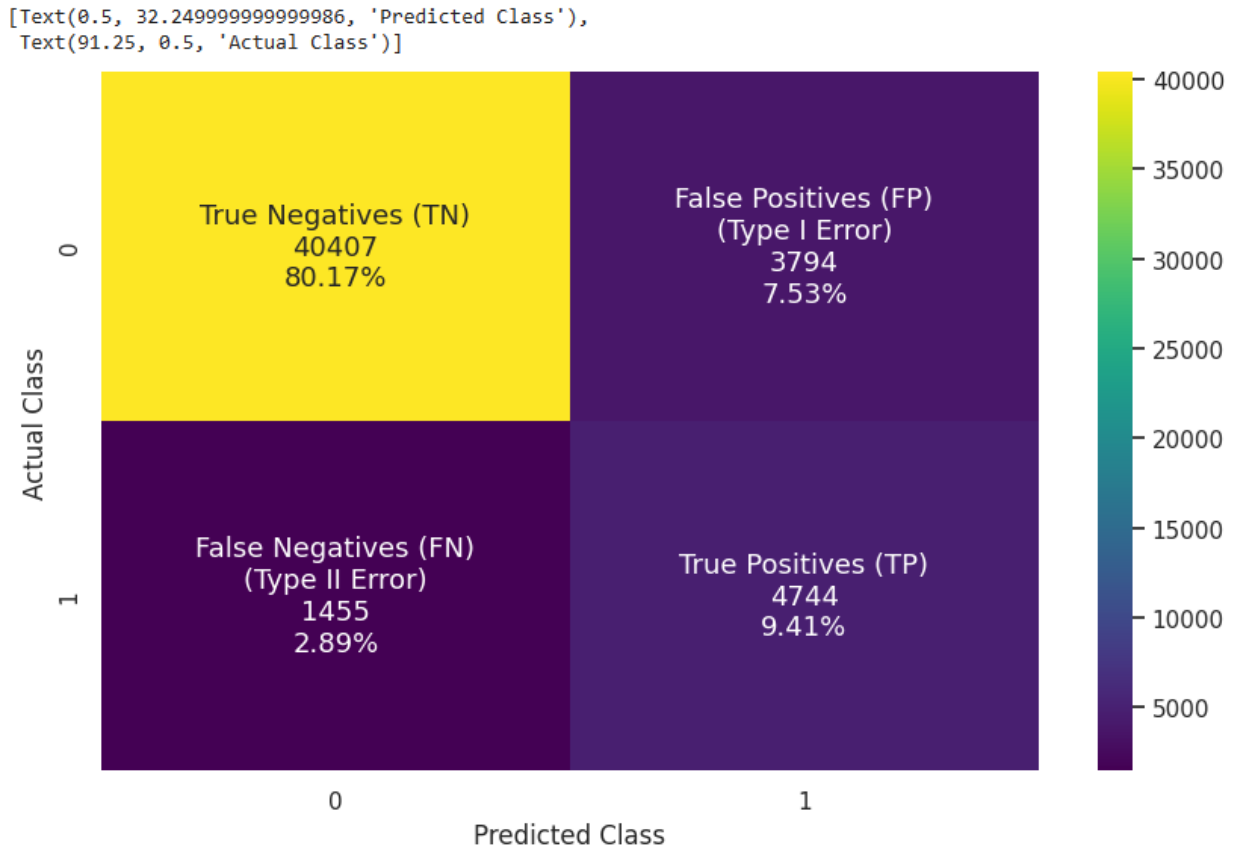
y_pred_new = rf_model.predict(X_test_new)
cf_matrix = confusion_matrix(y_test_new, y_pred_new)

group_names = ['True Negatives (TN)', 'False Positives (FP)\n(Type I
Error)',
               'False Negatives (FN)\n(Type II Error)', 'True Positives
(TP)']
group_counts = ['{0:0.0f}'.format(value) for value in cf_matrix.flatten()]
group_percentages = ['{0:.2%}'.format(value) for value in
cf_matrix.flatten()/np.sum(cf_matrix)]

labels = [f'{v1}\n{v2}\n{v3}' for v1, v2, v3 in zip(group_names,
group_counts, group_percentages)]

labels = np.asarray(labels).reshape(2, 2)

ax = sns.heatmap(cf_matrix, annot=labels, fmt="", annot_kws={'size': 13},
cmap='viridis')
ax.set(xlabel='Predicted Class', ylabel = 'Actual Class')
```

Hình 25. Confusion matrix cho dự đoán của mô hình Random Forest

Nhận xét

- True Negatives (TN):

- Số lượng: 40,407

- Tỷ lệ: 80.17%

- Đây là số lượng mẫu thuộc lớp 0 (âm tính) mà mô hình dự đoán chính xác là âm tính. Con số này khá lớn, cho thấy mô hình có khả năng phát hiện các trường hợp âm tính khá tốt.

- False Positives (FP) (Type I Error):

- Số lượng: 3,794

- Tỷ lệ: 7.53%

- Đây là số lượng mẫu thuộc lớp 0 nhưng mô hình lại dự đoán là lớp 1 (dương tính). Tỷ lệ này không quá cao, nhưng vẫn cần được cải thiện để giảm thiểu các dự đoán sai lệch.

- False Negatives (FN) (Type II Error):

- Số lượng: 1,455

- Tỷ lệ: 2.89%

- Đây là số lượng mẫu thuộc lớp 1 (dương tính) nhưng mô hình lại dự đoán là lớp 0. Tỷ lệ này khá thấp, cho thấy mô hình có khả năng phát hiện các trường hợp dương tính tốt, nhưng vẫn còn một số trường hợp bị bỏ sót.

- **True Positives (TP):**

- Số lượng: 4,744

- Tỷ lệ: 9.41%

- Đây là số lượng mẫu thuộc lớp 1 mà mô hình dự đoán chính xác là dương tính. Tỷ lệ này thể hiện khả năng của mô hình trong việc nhận diện các trường hợp dương tính.

Tổng quan:

- Mô hình Random Forest này có độ chính xác tương đối cao trong việc nhận diện các trường hợp âm tính (TN) và dương tính (TP).

- Số lượng FP và FN không quá cao, nhưng vẫn cần được chú ý để cải thiện hơn nữa.

- Để cải thiện mô hình, nên xem xét việc cân bằng lại dữ liệu (data balancing).

4.5. Mô hình AdaBoosting

4.5.1. Sơ lược về mô hình

AdaBoost (Adaptive Boosting) là một thuật toán boosting phổ biến trong học máy, hoạt động bằng cách kết hợp nhiều mô hình yếu (weak learners) để tạo ra một mô hình mạnh (strong learner). AdaBoost huấn luyện từng mô hình yếu tuần tự, và mỗi mô hình tập trung vào những mẫu mà các mô hình trước đó phân loại sai. Trọng số của các mẫu khó phân loại sẽ tăng lên, giúp mô hình cuối cùng có hiệu suất cao hơn. AdaBoost thường sử dụng cây quyết định đơn giản như các mô hình yếu và nổi bật ở khả năng tăng độ chính xác mà không dễ bị overfitting. Tuy nhiên, AdaBoosting cũng có thể overfitting nếu không chọn tham số phù hợp.

4.5.2. Đào tạo và đánh giá mô hình

```
from sklearn.ensemble import AdaBoostClassifier

param_grid = {
```

```

    'n_estimators': [50, 100, 150],
    'learning_rate': [0.01, 0.1, 1]
}

grid_search =
GridSearchCV(estimator=AdaBoostClassifier(random_state=15000),
param_grid=param_grid, cv=5, scoring='accuracy', verbose=2, n_jobs=-1)
grid_search.fit(X_res, y_res)

best_params = grid_search.best_params_
best_score = grid_search.best_score_

print(f"Best Parameters: {best_params}")
print(f"Best Cross-Validation Score: {best_score}")

best_model = grid_search.best_estimator_
y_pred_best = best_model.predict(X_test)

accuracy_best = accuracy_score(y_test, y_pred_best)
precision_best = precision_score(y_test, y_pred_best)
recall_best = recall_score(y_test, y_pred_best)
f1_best = f1_score(y_test, y_pred_best)
roc_auc_best = roc_auc_score(y_test, y_pred_best)

print(f"Test Set Accuracy: {accuracy_best}")
print(f"Test Set Precision: {precision_best}")
print(f"Test Set Recall: {recall_best}")
print(f"Test Set F1-score: {f1_best}")
print(f"Test Set ROC-AUC: {roc_auc_best}")

print("Confusion Matrix for Best Model:")
print(confusion_matrix(y_test, y_pred_best))
print("\nClassification Report for Best Model:")
print(classification_report(y_test, y_pred_best))

```

4.5.3. Đánh giá kết quả và giải thích mô hình

```

Fitting 5 folds for each of 9 candidates, totalling 45 fits
Best Parameters: {'learning_rate': 1, 'n_estimators': 150}
Best Cross-Validation Score: 0.6007558992415213
Test Set Accuracy: 0.5802910052910053
Test Set Precision: 0.14856409935556528
Test Set Recall: 0.5127402288922479
Test Set F1-score: 0.23037741340836326
Test Set ROC-AUC: 0.5512312799922664
Confusion Matrix for Best Model:
[[39121 27217]
 [ 4513  4749]]

```

Classification Report for Best Model:

	precision	recall	f1-score	support
0	0.90	0.59	0.71	66338
1	0.15	0.51	0.23	9262
accuracy			0.58	75600
macro avg	0.52	0.55	0.47	75600
weighted avg	0.80	0.58	0.65	75600

Nhận xét:

- **Độ chính xác (Accuracy):** Mô hình đạt độ chính xác tổng thể là 58%, thấp hơn so với kỳ vọng cho một mô hình dự đoán rủi ro.
- **Precision và Recall (lớp 1):** Precision cho lớp 1 (các trường hợp có rủi ro) là 0.1486, nghĩa là mô hình chỉ đúng 14.86% khi dự đoán là có rủi ro. Recall là 0.5127, nghĩa là mô hình tìm được 51.27% các trường hợp có rủi ro thực sự.
- **F1-Score (lớp 1):** F1-Score cho lớp 1 là 0.2304, cho thấy sự cân bằng giữa Precision và Recall không tốt.
- **ROC-AUC:** Điểm ROC-AUC là 0.5512, cho thấy mô hình chỉ hơi tốt hơn một chút so với dự đoán ngẫu nhiên.

4.6. Mô hình GradientBoosting

4.6.1. Sơ lược về mô hình

Gradient Boosting là một phương pháp ensemble khác cũng dựa trên boosting, nhưng sử dụng gradient descent để tối ưu hóa mô hình. Gradient Boosting xây dựng các mô hình yếu tuần tự, mỗi mô hình mới cố gắng sửa lỗi của mô hình trước đó bằng cách giảm gradient của hàm mất mát. Điều này giúp mô hình học tập theo từng bước nhỏ và liên tục cải thiện hiệu suất. Gradient Boosting nổi bật với khả năng xử lý tốt các dữ liệu phi tuyến và độ chính xác cao, nhưng đòi hỏi thời gian huấn luyện lâu và có nguy cơ overfitting nếu không được điều chỉnh cẩn thận.

4.6.2. Đào tạo và đánh giá mô hình

```
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import GradientBoostingClassifier

param_grid = {
    'n_estimators': [50, 100],
    'learning_rate': [0.01, 0.1],
    'max_depth': [3, 5]
}

model = GradientBoostingClassifier()

grid_search = GridSearchCV(model, param_grid, cv=3, verbose=2)

grid_search.fit(X_res, y_res)

best_params = grid_search.best_params_

best_model = GradientBoostingClassifier(**best_params)

best_model.fit(X_res, y_res)

accuracy = best_model.score(X_test, y_test)
accuracy

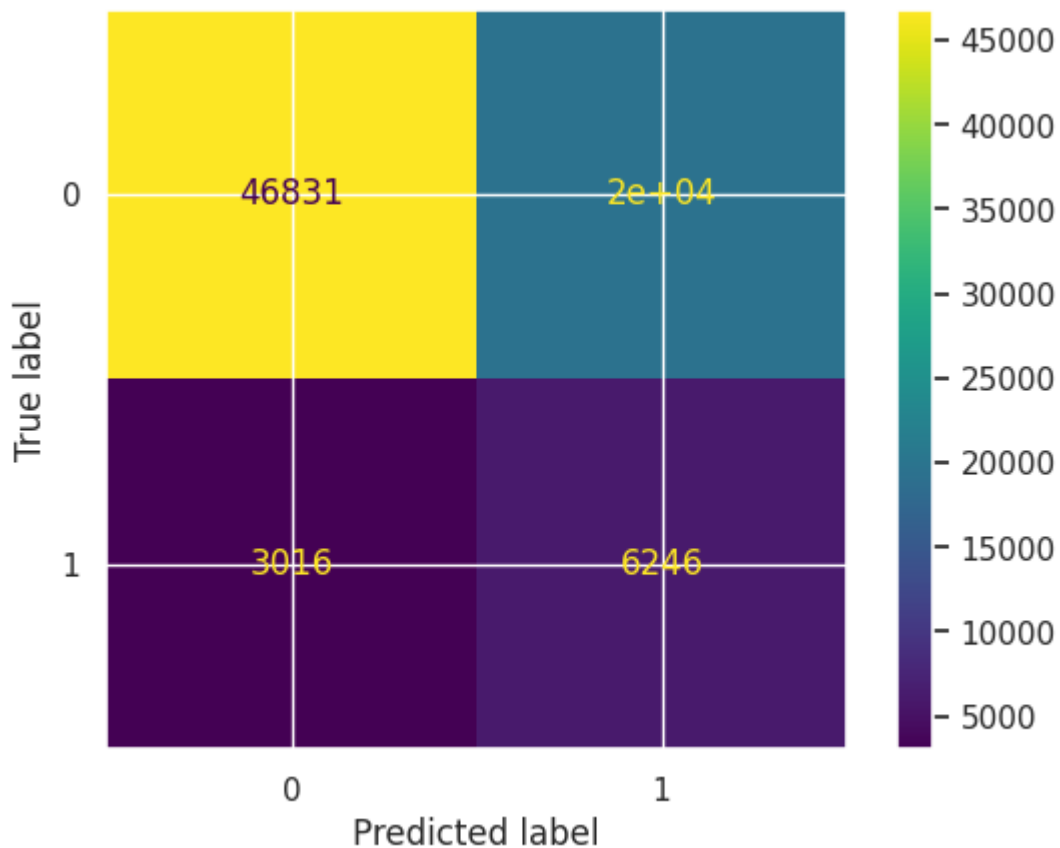
y_pred = best_model.predict(X_test)
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.94	0.71	0.81	66338
1	0.24	0.67	0.36	9262
accuracy			0.70	75600
macro avg	0.59	0.69	0.58	75600
weighted avg	0.85	0.70	0.75	75600

Hình 26: Báo cáo các độ đo của mô hình *GradientBoostingClassifier*

4.6.3. Đánh giá kết quả và giải thích mô hình

```
con_max = ConfusionMatrixDisplay.from_predictions(y_test, y_pred)
plt.show()
```



Hình 27: Confusion matrix cho dự đoán của *GradientBoostingClassifier*

- **Lớp 0 (Không vỡ nợ):**
 - **Precision cao (0.94):** Khi mô hình dự đoán một khách hàng không vỡ nợ, nó chính xác đến 94%.
 - **Recall thấp hơn (0.71):** Chỉ có 71% của tổng số khách hàng thực sự không vỡ nợ được mô hình dự đoán chính xác.
 - **F1-score (0.81):** Điểm số này cho thấy một sự cân bằng giữa precision và recall, nhưng vẫn có không gian để cải thiện, đặc biệt là về mặt recall.
- **Lớp 1 (Vỡ nợ):**
 - **Precision thấp (0.24):** Chỉ có 24% trong số những dự đoán vỡ nợ của mô hình là chính xác.
 - **Recall cao hơn (0.67):** Mô hình có khả năng nhận diện 67% của những trường hợp thực sự vỡ nợ.
 - **F1-score thấp (0.36):** Điểm số này cho thấy mô hình không hiệu quả lắm trong việc dự đoán lớp vỡ nợ, chủ yếu do precision thấp.
- **Accuracy tổng thể (0.70):** Mô hình chính xác 70% trên toàn bộ dữ liệu kiểm tra.

Nhận xét chung:

- Mô hình có vẻ tốt trong việc nhận diện khách hàng không vỡ nợ nhưng không hiệu quả trong việc nhận diện khách hàng vỡ nợ.
- Precision thấp cho lớp 1 có thể dẫn đến việc từ chối tín dụng cho nhiều khách hàng tiềm năng không vỡ nợ.

4.7. Mô hình Naive Bayes

4.7.1. Sơ lược về mô hình

Naive Bayes là một thuật toán học máy phổ biến được sử dụng cho các bài toán phân loại. Thuật toán này dựa trên nguyên tắc của định lý Bayes, và giả định "ngây thơ" (naive) rằng các đặc trưng là độc lập với nhau. Nguyên lý hoạt động của Naive Bayes là tính toán xác suất của mỗi lớp dựa trên các đặc trưng. Mặc dù giả định "ngây thơ" thường không đúng trong thực tế, nhưng nó giúp giảm độ phức tạp của mô hình và tăng hiệu suất

tính toán. Naive Bayes có các biến thể như Gaussian Naive Bayes, Multinomial Naive Bayes và Bernoulli Naive Bayes, phù hợp với các loại dữ liệu khác nhau như dữ liệu số, văn bản, hoặc dữ liệu nhị phân. Mặc dù đơn giản và dễ hiểu, Naive Bayes có thể không hiệu quả nếu các đặc trưng không độc lập với nhau, và có thể không hoạt động tốt trên dữ liệu có phân phối không chuẩn. Tuy nhiên, với dữ liệu lớn và có số chiều lớn, Naive Bayes thường cho kết quả tốt.

4.7.2. Đào tạo và đánh giá mô hình

```
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.feature_selection import SelectKBest, f_classif

pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('feature_selection', SelectKBest(f_classif)),
    ('classifier', GaussianNB(var_smoothing=1e-8))
])

# Huấn luyện mô hình với pipeline
pipeline.fit(X_res, y_res)

# Dự đoán trên tập kiểm tra và tính toán độ chính xác
y_pred = pipeline.predict(X_test)
accuracy = pipeline.score(X_test, y_test)

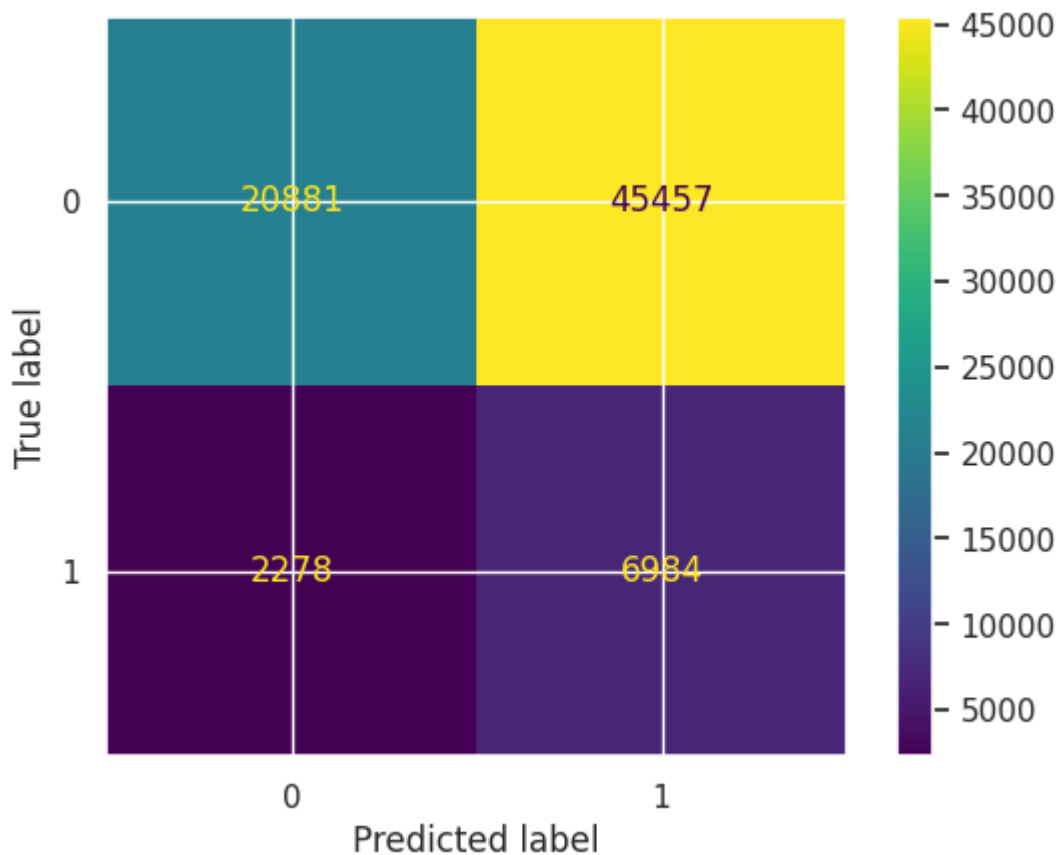
print("Accuracy:", accuracy)
print(classification_report(y_test, y_pred))
```


Accuracy: 0.36858465608465607				
	precision	recall	f1-score	support
0	0.90	0.31	0.47	66338
1	0.13	0.75	0.23	9262
accuracy			0.37	75600
macro avg			0.52	75600
weighted avg			0.81	75600

Hình 28: Báo cáo về các độ đo của GradientBoostingClassifier

4.7.3. Đánh giá kết quả và giải thích mô hình

```
con_max = ConfusionMatrixDisplay.from_predictions(y_test, y_pred)
plt.show()
```



Hình 29: Confusion matrix cho dự đoán của Naive Bayes

- **Lớp 0 (Không vỡ nợ):**
 - **Precision cao (0.90):** Khi mô hình dự đoán một khách hàng không vỡ nợ chính xác đến 90%
 - **Recall thấp (0.31):** Chỉ có 31% của tổng số khách hàng thực sự không vỡ nợ được mô hình dự đoán chính xác.
 - **F1-score thấp (0.47):** Điểm số này cho thấy mô hình không hiệu quả trong việc dự đoán lớp không vỡ nợ, chủ yếu do recall thấp.
- **Lớp 1 (Vỡ nợ):**
 - **Precision thấp (0.13):** Chỉ có 13% trong số những dự đoán vỡ nợ của mô hình là chính xác.
 - **Recall cao (0.75):** Mô hình có khả năng nhận diện 75% của những trường hợp thực sự vỡ nợ.

- **F1-score thấp (0.23):** Điểm số này cho thấy mô hình không hiệu quả lắm trong việc dự đoán lớp vỡ nợ, chủ yếu do precision rất thấp.
- **Accuracy tổng thể (0.37):** Mô hình chính xác 37% trên toàn bộ dữ liệu kiểm tra.

Nhận xét chung:

- Mô hình này có vẻ không cân bằng giữa hai lớp. Mặc dù có precision cao cho lớp không vỡ nợ, nhưng recall thấp cho thấy mô hình bỏ qua nhiều trường hợp thực sự không vỡ nợ.
- Đối với lớp vỡ nợ, mô hình có recall cao nhưng precision rất thấp, điều này có nghĩa là mô hình dự đoán quá nhiều trường hợp là vỡ nợ khi chúng không phải, dẫn đến nhiều báo động giả.
- Độ chính xác tổng thể thấp cho thấy mô hình này có thể không phù hợp để đưa ra quyết định về việc cấp tín dụng.

4.8. Mô hình ANN

4.8.1. Sơ lược về mô hình

Artificial Neural Networks (ANN) là một mô hình học máy phỏng theo cấu trúc và chức năng của mạng nơron sinh học, bao gồm các tầng nơron liên kết với nhau. Mỗi nơron nhận đầu vào, thực hiện một phép tính và truyền kết quả đến các nơron khác ở tầng tiếp theo. ANN có khả năng học các mô hình phức tạp và phi tuyến từ dữ liệu, được sử dụng rộng rãi trong nhiều lĩnh vực như nhận dạng hình ảnh, xử lý ngôn ngữ tự nhiên và dự đoán chuỗi thời gian. Tuy nhiên, ANN đòi hỏi lượng dữ liệu lớn để huấn luyện, cần nhiều tài nguyên tính toán và dễ bị overfitting nếu không được điều chỉnh đúng cách.

4.8.2. Đào tạo và đánh giá mô hình

```
from sklearn.preprocessing import OneHotEncoder, StandardScaler,
QuantileTransformer
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import callbacks
import joblib
from sklearn.compose import ColumnTransformer
```

```

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.callbacks import EarlyStopping

from sklearn.compose import ColumnTransformer
tr1 = ColumnTransformer([
    ('ohe', OneHotEncoder(sparse_output=False, handle_unknown='ignore'),
[3, 4, 5, 6, 7, 8, 11]),
    ('std', StandardScaler(), [0]),
    ('qtl_trsf', QuantileTransformer(output_distribution="normal"), [1, 2,
9, 10])
])

X_train_Scaled = tr1.fit_transform(X_train)
X_test_Scaled = tr1.transform(X_test)

ann_model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(256, activation='elu',
kernel_initializer='he_normal', input_shape=(X_train_Scaled.shape[1],)),
    tf.keras.layers.Dense(128, activation='elu',
kernel_initializer='he_normal'),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Dense(64, activation='elu',
kernel_initializer='he_normal'),
    tf.keras.layers.Dense(32, activation='elu',
kernel_initializer='he_normal'),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Dense(8, activation='elu',
kernel_initializer='he_normal'),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
ann_model.summary()

```

Model: "sequential_1"

Layer (type)	Output Shape	Param #
dense_7 (Dense)	(None, 256)	147456
dense_8 (Dense)	(None, 128)	32896
batch_normalization_3 (Batch Normalization)	(None, 128)	512
dense_9 (Dense)	(None, 64)	8256
dense_10 (Dense)	(None, 32)	2080
batch_normalization_4 (Batch Normalization)	(None, 32)	128
dense_11 (Dense)	(None, 8)	264
batch_normalization_5 (Batch Normalization)	(None, 8)	32
dense_12 (Dense)	(None, 1)	9

=====
Total params: 191633 (748.57 KB)
Trainable params: 191297 (747.25 KB)
Non-trainable params: 336 (1.31 KB)
=====

```
# Compile the model
ann_model.compile(optimizer='Nadam',
                  loss='binary_crossentropy',
                  metrics=['accuracy'])

import time
n_epochs = 23
new_training = 1
time_chay = 0
if new_training:
    model_saver =
keras.callbacks.ModelCheckpoint('/content/drive/MyDrive/Colab
Notebooks/LoanPrediction_Model/ann_model.h5',monitor='val_accuracy',
save_best_only=True)
    early_stopper = keras.callbacks.EarlyStopping(monitor='val_loss',
patience=5)
```

```

performance_sched =
keras.callbacks.ReduceLROnPlateau(monitor='val_loss', patience=2,
factor=0.2)
#with tf.device('/gpu:0'):
start_time = time.time()
history = ann_model.fit(X_train_Scaled, y_train, epochs=n_epochs,
                        batch_size=32,
                        validation_data=(X_test_Scaled, y_test),
                        callbacks=[model_saver, early_stopper,
performance_sched], verbose=1)
end_time = time.time()
time_chay = end_time - start_time
print("Thời gian chạy: ", time_chay)
history = history.history
joblib.dump(history, '/content/drive/MyDrive/Colab
Notebooks/LoanPrediction_Model/history_ann_model')
else:
    try:
        ann_model = keras.models.load_model('/content/drive/MyDrive/Colab
Notebooks/LoanPrediction_Model/ann_model.h5')
        history = joblib.load('/content/drive/MyDrive/Colab
Notebooks/LoanPrediction_Model/history_ann_model')
    except Exception as e:
        history = joblib.load('/content/drive/MyDrive/Colab
Notebooks/LoanPrediction_Model/history_ann_model')

history.keys()
print("Train_Accuracy:", history['accuracy'][-1])
print("Val_Accuracy:", history['val_accuracy'][-1])
#endregion

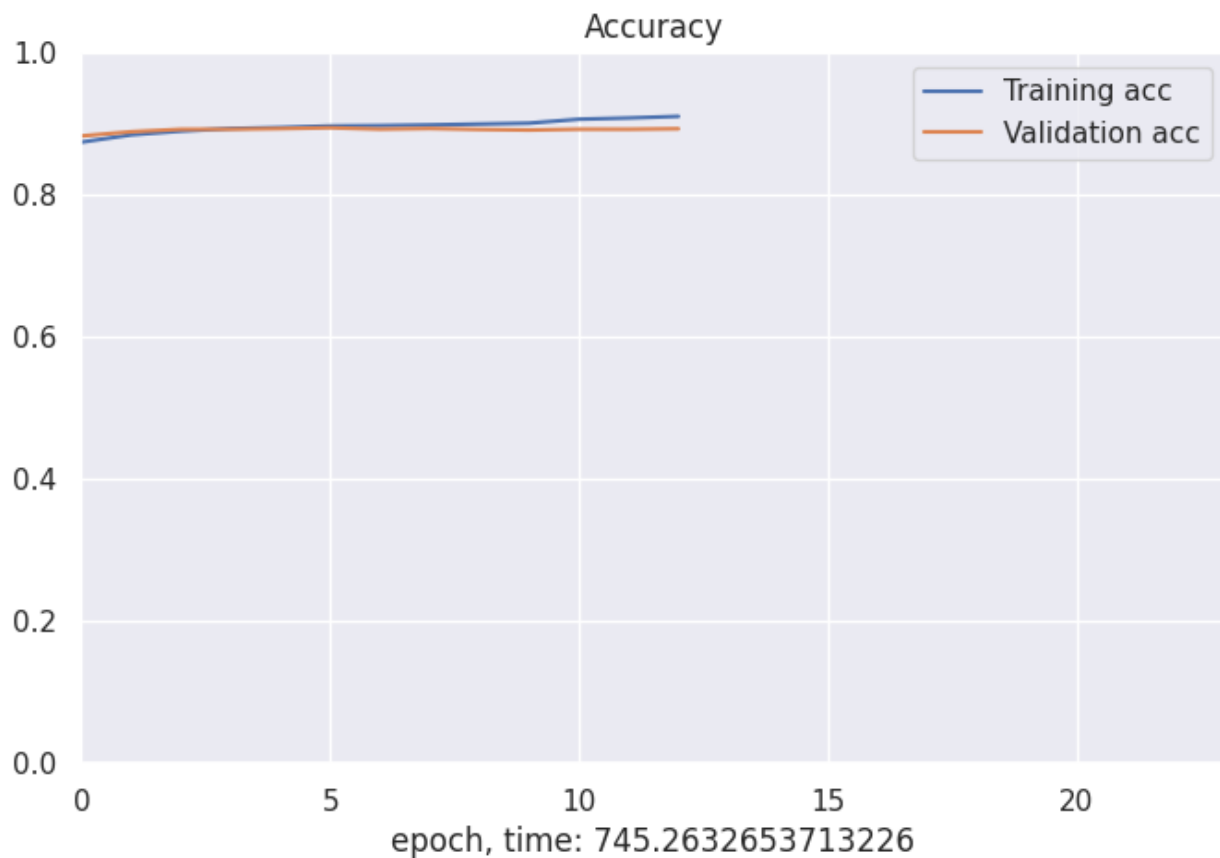
```

```

Epoch 1/23
5509/5513 [=====>.] - ETA: 0s - loss: 0.3447 - accuracy: 0.8744/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWar
ning: saving_api.save_model(
5513/5513 [=====] - 61s 10ms/step - loss: 0.3446 - accuracy: 0.8744 - val_loss: 0.2972 - val_accuracy: 0.8832 - lr: 0.0010
Epoch 2/23
5513/5513 [=====] - 65s 12ms/step - loss: 0.2815 - accuracy: 0.8847 - val_loss: 0.2630 - val_accuracy: 0.8891 - lr: 0.0010
Epoch 3/23
5513/5513 [=====] - 52s 9ms/step - loss: 0.2547 - accuracy: 0.8900 - val_loss: 0.2488 - val_accuracy: 0.8926 - lr: 0.0010
Epoch 4/23
5513/5513 [=====] - 57s 10ms/step - loss: 0.2381 - accuracy: 0.8937 - val_loss: 0.2453 - val_accuracy: 0.8927 - lr: 0.0010
Epoch 5/23
5513/5513 [=====] - 58s 11ms/step - loss: 0.2265 - accuracy: 0.8951 - val_loss: 0.2371 - val_accuracy: 0.8935 - lr: 0.0010
Epoch 6/23
5513/5513 [=====] - 58s 11ms/step - loss: 0.2162 - accuracy: 0.8973 - val_loss: 0.2363 - val_accuracy: 0.8947 - lr: 0.0010
Epoch 7/23
5513/5513 [=====] - 54s 10ms/step - loss: 0.2082 - accuracy: 0.8977 - val_loss: 0.2387 - val_accuracy: 0.8928 - lr: 0.0010
Epoch 8/23
5513/5513 [=====] - 54s 10ms/step - loss: 0.2015 - accuracy: 0.8987 - val_loss: 0.2338 - val_accuracy: 0.8937 - lr: 0.0010
Epoch 9/23
5513/5513 [=====] - 54s 10ms/step - loss: 0.1945 - accuracy: 0.9000 - val_loss: 0.2359 - val_accuracy: 0.8923 - lr: 0.0010
Epoch 10/23
5513/5513 [=====] - 59s 11ms/step - loss: 0.1890 - accuracy: 0.9012 - val_loss: 0.2419 - val_accuracy: 0.8915 - lr: 0.0010
Epoch 11/23
5513/5513 [=====] - 59s 11ms/step - loss: 0.1696 - accuracy: 0.9067 - val_loss: 0.2391 - val_accuracy: 0.8927 - lr: 2.0000e-04
Epoch 12/23
5513/5513 [=====] - 59s 11ms/step - loss: 0.1643 - accuracy: 0.9085 - val_loss: 0.2450 - val_accuracy: 0.8926 - lr: 2.0000e-04
Epoch 13/23
5513/5513 [=====] - 54s 10ms/step - loss: 0.1596 - accuracy: 0.9108 - val_loss: 0.2456 - val_accuracy: 0.8933 - lr: 4.0000e-05
Thời gian chạy: 745.2632653713226
Train_Accuracy: 0.9107823371887207
Val_Accuracy: 0.8933465480804443

```

```
# %% Plot learning curves
import pandas as pd
pd.DataFrame({'Training acc':history['accuracy'],'Validation acc':history['val_accuracy']}).plot(figsize=(8, 5))
plt.grid(True)
plt.xlim(0, n_epochs)
plt.ylim(0, 1)
plt.xlabel(f'epoch, time: {time_chay}')
plt.xticks(np.arange(0,n_epochs+1,5))
plt.savefig("/content/drive/MyDrive/Colab Notebooks/LoanPrediction_Model/ann_model")
plt.title("Accuracy")
plt.show()
```



Hình 30. Biểu đồ biểu diễn quá trình học của mô hình ANN

4.8.3. Đánh giá kết quả và giải thích mô hình

Hiệu suất Mô hình:

- Training Accuracy: 0.9108
- Validation Accuracy: 0.8933

Quá trình Huấn luyện:

- Loss và Accuracy cải thiện qua các epoch:
 - Ban đầu, loss giảm và accuracy tăng đáng kể.
 - Từ epoch thứ 7 trở đi, các chỉ số không cải thiện đáng kể và có dấu hiệu dao động nhẹ, cho thấy mô hình có thể đã đạt tới mức bão hòa.

Đánh giá:

- Training và Validation Accuracy đều cao: Điều này cho thấy mô hình có khả năng học tốt từ dữ liệu huấn luyện và tổng quát tốt trên dữ liệu kiểm tra.
- Không có dấu hiệu Overfitting rõ ràng: Accuracy trên tập huấn luyện và kiểm tra không chênh lệch quá lớn, cho thấy mô hình không bị overfit.

Tổng kết độ chính xác của các mô hình:

```
data = {  
    'Model': ['Logistic Regression', 'KNN', 'Random Forest', 'AdaBoost',  
             'Gradient Boosting', 'Naive Bayes', 'ANN'],  
    'Accuracy': [accuracy_logis, accuracy_knn, accuracy_rf, accuracy_ada,  
                 accuracy_gra, accuracy_naive, accuracy_ann]  
}  
  
df = pd.DataFrame(data)  
df
```


	Model	Accuracy
0	Logistic Regression	0.61
1	KNN	0.87
2	Random Forest	0.90
3	AdaBoost	0.58
4	Gradient Boosting	0.70
5	Naive Bayes	0.37
6	ANN	0.89

Hình 31. DataFrame biểu diễn độ chính xác của từng mô hình

Nhận xét tổng:

- Logistic Regression: Mô hình Logistic Regression có độ chính xác là 61.32%, cải thiện so với lần trước nhưng vẫn thấp hơn nhiều so với các mô hình khác như KNN, Random Forest, và ANN. Điều này cho thấy Logistic Regression vẫn không phải là lựa chọn tốt nhất cho bài toán này.
- K-Nearest Neighbors (KNN): KNN đạt độ chính xác 86.60%, một trong những mô hình có hiệu suất tốt. KNN hoạt động tốt trong bài toán này do có thể tận dụng được tính tương đồng của các điểm dữ liệu gần nhau.
- Random Forest: Random Forest đạt độ chính xác cao nhất (89.59%), chỉ thua kém một chút so với ANN. Đây là mô hình rất mạnh trong việc dự đoán, đặc biệt là với dữ liệu phức tạp.
- AdaBoost: AdaBoost có độ chính xác là 58.03%, vẫn thấp hơn các mô hình khác. AdaBoost có thể không phù hợp với dữ liệu này, có thể do dữ liệu có nhiều nhiễu.
- Gradient Boosting: Gradient Boosting đạt độ chính xác 70.21%, cao hơn AdaBoost và Logistic Regression nhưng vẫn thấp hơn KNN và Random Forest. Gradient Boosting có tiềm năng tốt nhưng có thể cần thêm tinh chỉnh.

- Naive Bayes: Naive Bayes có độ chính xác thấp nhất (36.86%). Điều này có thể do giả định độc lập của các biến không phù hợp với dữ liệu thực tế trong bài toán này.
- Artificial Neural Network (ANN): ANN có độ chính xác cao thứ hai (89.21%), gần bằng với Random Forest. ANN có khả năng học các mẫu phức tạp trong dữ liệu, cho thấy tiềm năng mạnh mẽ trong việc dự đoán tỷ lệ vỡ nợ.

Kết luận:

- Random Forest và ANN là hai mô hình có hiệu suất tốt nhất, phù hợp cho bài toán dự đoán tỷ lệ vỡ nợ của người dân.
- KNN cũng là một mô hình tiềm năng với độ chính xác cao.
- Các mô hình như Logistic Regression, AdaBoost, và Naive Bayes có độ chính xác thấp hơn và có thể không phù hợp cho bài toán này.
- Gradient Boosting cần thêm tinh chỉnh để đạt được hiệu suất tốt hơn.

CHƯƠNG 5. KẾT LUẬN VÀ KIẾN NGHỊ

5.1. Tóm tắt kết quả nghiên cứu

Dữ liệu: Sử dụng dữ liệu lịch sử về hành vi của khách hàng để xây dựng mô hình dự đoán.

Mô hình học máy: Áp dụng các mô hình học máy như Random Forest, Logistic Regression, AdaBoosting, Navie Bayes, ANN để dự đoán khả năng vỡ nợ của khách hàng mới.

Đánh giá mô hình: Sử dụng các phương pháp đánh giá như precision, recall, F1-score và ROC-AUC để đánh giá hiệu suất của mô hình trên tập dữ liệu kiểm tra.

Kết quả: Hiểu được và phân tích được sự phù hợp và không phù hợp của các loại mô hình khác nhau, tối ưu chỉ số accuracy đến mức tối đa

5.2. Kiến nghị dựa trên kết quả

Tích hợp mô hình vào quy trình quyết định: tích hợp mô hình dự đoán vào quy trình xác định rủi ro và xác nhận khách hàng mới. Điều này giúp tổ chức nhanh chóng xác định khách hàng có rủi ro cao hơn và đưa ra biện pháp phòng tránh kịp thời.

Cập nhật và tinh chỉnh định kỳ: đề xuất cập nhật định kỳ mô hình dự đoán dựa trên dữ liệu mới và tinh chỉnh mô hình để duy trì hiệu suất cao.

5.3. Hướng nghiên cứu và phát triển tiếp theo

Nâng cao hiệu suất mô hình: tiếp tục nghiên cứu và thử nghiệm các phương pháp mới để cải thiện hiệu suất của mô hình, bao gồm việc sử dụng các thuật toán học máy tiên tiến và tối ưu hóa siêu tham số.

Mở rộng dữ liệu: nghiên cứu việc tích hợp thêm các thông tin mới vào dữ liệu, như dữ liệu tín dụng từ các tổ chức bên ngoài để cải thiện dự đoán về khả năng vỡ nợ.

Nghiên cứu về giải pháp phòng tránh: tìm hiểu và phát triển các giải pháp phòng tránh để giảm thiểu rủi ro về vỡ nợ từ nhóm khách hàng có khả năng cao.

TÀI LIỆU THAM KHẢO

- [1]. ***Loan Default Prediction Based on Customer Behavior Using Xgboost And SMOTE ENN*** - Mehul Wankhede (Truy cập ngày 10/05/2024)

Link: <https://medium.com/@mswankhede7/loan-prediction-based-on-customer-behavior-using-xgboost-and-smote-enn-983b35380af8>

- [2]. ***Loan Prediction Based on Customer Behavior*** - SUBHAM SURANA (Truy cập ngày 12/05/2024)

Link: <https://www.kaggle.com/datasets/subhamjain/loan-prediction-based-on-customer-behavior/data>

- [3]. ***Chi-squared test*** - From Wikipedia, the free encyclopedia (Truy cập ngày 09/05/2024)

Link: https://en.wikipedia.org/wiki/Chi-squared_test

- [4]. ***Google Colab*** - Google

Link: <https://colab.research.google.com/>