BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC XÂY DỰNG

BÁO CÁO TỔNG KẾT

ĐỀ TÀI NGHIÊN CỨU KHOA HỌC SINH VIÊN NĂM HỌC 2020 – 2021

Nghiên cứu xây dựng mô hình giải quyết bài toán "đánh giá tín dụng cá nhân"

Mã số: CNTT-2020-07

Sinh viên thực hiện:

Đào Việt Cường 64CS2 28264

Lý Hải Yến 64CS2 1557164

Giáo viên hướng dẫn:

ThS. Hoàng Nam Thắng

KS. Nguyễn Đình Quý

Hà Nội, 07/2021

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC XÂY DỰNG

BÁO CÁO TỔNG KẾT

ĐỀ TÀI NGHIÊN CỨU KHOA HỌC SINH VIÊN NĂM HỌC 2020 – 2021

Nghiên cứu xây dựng mô hình giải quyết bài toán "đánh giá tín dụng cá nhân"

Mã số: CNTT-2020-07

Sinh viên thực hiện:

Đào Việt Cường 64CS2 28264

Lý Hải Yến 64CS2 1557164

Giáo viên hướng dẫn:

ThS. Hoàng Nam Thắng

KS. Nguyễn Đình Quý

Cán bộ hướng dẫn **Sinh viên trưởng nhóm**

Hà Nội, 07/2021

MỤC LỤC

MỤC LỤC HÌNH VẼ

MỤC LỤC BẢNG BIỂU

DANH MỤC CÁC CHỮ CÁI VIẾT TẮT

ANN – Mạng thần kinh nhân tạo (Artificial Neural Networks)

BCBS - Ủy ban Basel về Giám sát Ngân hàng (The Basel Committee on Banking Supervision).

BIS - Ngân hàng Thanh toán Quốc tế (Bank for International Settlements).

BP – Truyền ngược (Back-Propagation).

CAR - tỷ lệ an toàn vốn (Capital adequacy ratio).

ClassRBM – Máy Boltzmann phân loại bị hạn chế (Classification restricted Boltzmann machine).

CNTT - Công nghệ thông tin.

DNN – Mạng nơron sâu (Deep Neural Network).

DTC - Decision Tree Cassification.

EAD - Tổng dư nợ tại thời điểm khách hàng không trả được nợ (Exposure at default).

EL - Tổn thất dự kiến (Expected loss).

IRB - Tiếp cận dựa trên đánh giá nội bộ (Internal Ratings-Based).

LGBM – LightGBM.

LGD - Tỉ trọng tổn thất ước tính (Loss given default).

LR - Logistic Regression.

LPD – tổng lượng có khả năng vỡ nợ thấp (low-default portfolio).

M - Thời gian đáo hạn hiệu quả (Effective maturity).

MLE - Maximum Likelihood Estimation.

MSE - Mean Squared Error.

MSLE - Mean squared logarithmic error.

OCC – phân loại một lớp (one-class classification)

PD - Xác suất vỡ nợ (Probability of default)

RMSE - Root Mean Squared Error.

RMSLE - Root mean squared logarithmic error.

RWA - Tổng tài sản có trọng số rủi ro (risk weighted assets)

SMC - Hệ số tương tự ngẫu nhiên (Simple Matching Coefficient)

STD - Độ lệch chuẩn (standard deviation).

SVM – Máy vectơ hỗ trợ (Support Vector Machines)

SMOTE – Kỹ thuật lấy mẫu quá mức cho nhóm thiểu số tổng hợp (Synthetic Minority Oversampling Technique)

LỜI MỞ ĐẦU

Đề tài này thuộc danh mục đề tài Nghiên cứu khoa học Sinh viên năm 2019-2020, mã số CNTT-2020-07 theo Quyết định số 1354/QĐ-ĐHXD ngày 11 tháng 11 năm 2019 của Hiệu trưởng trường Đại học Xây dựng. Tên đề tài "Nghiên cứu xây dựng mô hình giải quyết bài toán đánh giá tín dụng cá nhân", mã số CNTT-2020-07.

ĐẶT VẤN ĐỀ

Đối với các tổ chức tài chính và nền kinh tế nói chung, vai trò của việc chấm điểm tín dụng trong các quyết định cho vay có vai trò rất lớn. Mô hình đánh giá điểm tín dụng chính xác và hoạt động tốt cho phép bên cho vay kiểm soát mức độ rủi ro của họ thông qua việc phân bổ tín dụng có chọn lọc dựa trên phân tích thống kê dữ liệu lịch sử khách hàng. Nghiên cứu này xác định và nghiên cứu một số thách thức cụ thể xảy ra trong quá trình phát triển mô hình đánh giá tín dụng.

NỘI DUNG NGHIÊN CỨU

Tìm hiểu và nắm bắt lý thuyết cơ bản về khai phá dữ liệu và học máy, tìm hiểu về mô hình học máy cây quyết định dựa trên phân loại và hồi quy, áp dụng vào vấn đề đánh giá tín dụng cá nhân.

CHƯƠNG 1: Giới thiệu đề tài nghiên cứu

Bao gồm giới thiệu khái quát về vấn đề cần nghiên cứu, những đóng góp của đề tài, các đề tài cùng phạm vi và những thách thức của bài toán chấm điểm tín dụng cá nhân.

CHƯƠNG 2: Các khái niệm và kỹ thuật cơ bản

Những vấn đề và phương pháp giải quyết trên các loại dữ liệu khác nhau. Một số mô hình được sử dụng trong đánh giá tín dụng cá nhân.

CHƯƠNG 3: Thực nghiệm và đánh giá kết quả nghiên cứu

Cách cài đặt ngôn ngữ python và các thư viện cần thiết cho việc khai phá dữ liệu, áp dụng để đánh giá tín dụng cá nhân rồi đưa ra nhận xét, đánh giá.

CHƯƠNG 1: Giới thiệu đề tài nghiên cứu

1.1. Khái quát

Ngân hàng đóng một vai trò quan trọng trong nền kinh tế thị trường. Họ quyết định ai có thể nhận được nguồn tài chính và theo những điều khoản nào và có thể đưa ra hoặc phá vỡ các quyết định đầu tư. Để thị trường và xã hội hoạt động, các cá nhân và công ty cần tiếp cận tín dụng.

Các mô hình chấm điểm tín dụng, phỏng đoán xác suất vỡ nợ, là phương pháp mà các ngân hàng sử dụng để xác định xem có nên cấp một khoản vay hay không. Trong đề tài này nhóm xây dựng các mô hình học máy dự đoán xác suất ai đó sẽ gặp khó khăn về tài chính trong hai năm tới. Đầu vào là dữ liệu lấy từ một cuộc thi dự đoán trên Kaggle có tên "Give Me Some Credit"[1], dữ liệu lịch sử được cung cấp trên 250.000 người vay, gồm 2 file train và test dưới định dạng csv. Nội dung các feature của bộ dữ liệu được tổng hợp trong bảng sau:

Bảng 1- 1: Ý nghĩa của các feature trong tập dữ liệu

Tên biến	***Miêu tả***	***Dạng***
Thời gian trễ hạn nghiêm trọng trong 2 năm (SeriousDlqin2yrs)	Người trải qua vi phạm 90 ngày quá hạn hoặc tệ hơn	Y/N
Sử dụng quay vòng của hạn mức tín dụng không có bảo đảm(RevolvingUtilizationOfUnsecuredLines)	Tổng số dư trên thẻ tín dụng và hạn mức tín dụng cá nhân ngoại trừ bất động sản và không có nợ trả góp như khoản vay mua ô tô chia cho tổng hạn mức tín dụng	Phần trăm
Tuổi(age)	Tuổi của người vay tính theo năm	integer
Số thời gian 30-59 ngày quá hạn không tệ hơn(NumberOfTime30-59DaysPastDueNotWorse)	Số lần người vay đã quá hạn 30-59 ngày nhưng không tệ hơn trong 2 năm gần đây.	integer
Tỷ lệ nợ(DebtRatio)	Thanh toán nợ hàng tháng, cấp dưỡng, chi phí sinh hoạt chia cho tổng thu nhập hàng tháng	Phần trăm
Thu nhập hàng tháng(MonthlyIncome)	Thu nhập hàng tháng	real

Số lượng hạn mức tín dụng mở và các khoản cho vay(NumberOfOpenCreditLinesAndLoans)	Số khoản vay Mở (trả góp như khoản vay mua ô tô hoặc thế chấp) và Số lượng tín dụng (ví dụ: thẻ tín dụng)	integer
Số lần trễ 90 ngày(NumberOfTimes90DaysLate)	Số lần người vay đã quá hạn từ 90 ngày trở lên.	integer
Đánh số các khoản vay hoặc dòng bất động sản (NumberRealEstateLoansOrLines)	Số lượng các khoản vay thế chấp và bất động sản bao gồm hạn mức tín dụng vốn chủ sở hữu nhà	integer
Số thời gian 60-89 ngày quá hạn không tệ hơn(NumberOfTime60-89DaysPastDueNotWorse)	Số lần khách hàng vay đã được 60-89 ngày quá hạn nhưng không tồi tệ hơn trong vòng 2 năm trở lại đây	integer
Số người phụ thuộc(NumberOfDependents)	Số người phụ thuộc trong gia đình không bao gồm bản thân họ (vợ / chồng, con cái, v.v.)	integer

Trong phạm vi nghiên cứu, nhóm sẽ chỉ sử dụng mô hình học máy học có giám sát, với đầu ra cho trước trong tập huấn luyện là biến "Thời gian trễ hạn nghiêm trọng trong 2 năm" và đầu vào là các biến còn lại. Nhiệm vụ của mô hình sẽ phải dự đoán xác suất của biến "Thời gian trễ hạn nghiêm trọng trong 2 năm" trong tập kiểm tra, hay nói cách khác là dự đoán xác suất suất một người nào đó có thể không trả được khoản vay trong vòng hai năm tới. Các công việc cụ thể trong đề tài này sẽ được nêu trong mục dưới đây.

1.2. Đóng góp của đề tài

Trong nghiên cứu này, nhóm giải quyết một số vấn đề chính trong lĩnh vực tính điểm tín dụng và học máy. Trong các chương tiếp theo, nhóm thực hiện:

(i) Nêu ra các kỹ thuật phân loại và ứng dụng của chúng vào việc chấm điểm tín dụng.

(ii) Sử dụng các kỹ thuật để xử lí các loại dữ liệu bị mất, dữ liệu bị lỗi do các nguyên nhân khách quan.

(iii) Nghiên cứu và giải quyết các vấn đề trong bộ dữ liệu thực, đặc việc là việc mất cân bằng giữa hai nhóm đối tượng, cụ thể là nhóm đối tượng có khả năng vỡ nợ thấp hơn rất nhiều so với nhóm không có khả năng vỡ nợ.

(iv) Xây dựng các mô hình học máy để giải quyết bài toàn và khảo sát các mô hình để đánh giá, lựa chọn mô hình tốt.

1.3. Tình hình các nghiên cứu cùng phạm vi.

Mạng nơ-ron ngày càng trở nên phổ biến với các nhà nghiên cứu trong những năm gần đây. Li và cộng sự. (2002)[2] đề xuất một mô hình dựa trên thuật toán Truyền ngược (BP) để xác định chủ nợ tốt hay xấu. Hu và Tang (2006)[3] đã đề xuất một đánh giá rủi ro tín dụng được biết dựa trên mạng nơ-ron nhân tạo (ANN), đo lường điểm tín dụng của bên đi vay. Các ứng viên phù hợp nhất cho mô hình này là các ngân hàng thương mại có dữ liệu chưa đầy đủ. Dima và cộng sự (2010)[4] đề xuất mô hình ANN để đánh giá rủi ro tín dụng doanh nghiệp nhằm phân loại nhóm đối tượng nợ tốt và xấu. Trong tài liệu của mình, họ đánh giá rủi ro của việc công ty vỡ nợ dựa trên mẫu quốc tế gồm 3.000 công ty đăng ký tín dụng tại một ngân hàng quốc tế hoạt động ở Romania. Mẫu bao gồm tình hình dân số chung của các công ty ở Romania. Dựa trên lịch sử tín dụng trong quá khứ, họ đã chia các công ty thành bảy loại. Họ thực hiện ước tính của mình bằng cách sử dụng hồi quy logit và sau đó là ANN và so sánh kết quả với Standard & Poor's.

Tomczak và Zieba (2014)[5] trong nghiên cứu của họ, đã đề xuất một kỹ thuật học máy mới sử dụng Máy Boltzmann phân loại bị hạ chế (ClassRBM) để xây dựng bảng điểm tín dụng. Bảng điểm là mô hình đơn giản nhất để giải thích và có thể dễ dàng áp dụng cho bất kỳ hệ thống ngân hàng nào. Không giống như các phương pháp tiêu chuẩn, cách tiếp cận của họ sử dụng trình phân loại mạnh mẽ để giải quyết vấn đề phân phối lớp không đồng đều và xây dựng một mô hình tính điểm cực kỳ dễ hiểu và dễ áp dụng. Baesens và cộng sự (2003)[6] đã phân tích ba tập dữ liệu thực tế và trình bày kết quả. Phân tích được thực hiện bằng kỹ thuật trích xuất quy tắc từ mạng nơron. Họ kết luận rằng kỹ thuật trích xuất quy tắc thần kinh có thể được sử dụng để phân tích rủi ro tín dụng. Có thể thấy, các nhà nghiên cứu đang chuyển sang các hệ thống lai với mạng nơ-ron. Huang và cộng sự. (2005)[7] đề xuất phân loại các đối tượng vay vốn ngân hàng thương mại nhà nước sử dụng mạng nơron mờ.

Ngoài các phương pháp tiếp cận dựa trên mạng nơ-ron và SVM, một số kỹ thuật phân loại khác được đề xuất để đánh giá mức độ uy tín. Mặc dù không phải là một mô hình phân loại phổ biến cho điểm tín dụng, nhưng cách tiếp cận Naive Bayes cũng đã được đề xuất. Vedala và Kumar (2012)[8] đề xuất xếp hạng Naive Bayes để xếp hạng uy tín. Xếp hạng này chủ yếu được thực hiện trên các nền tảng cho vay điện

tử sử dụng mạng xã hội để mở rộng cơ sở dữ liệu của nó. Okesola và cộng sự (2017)[9] cũng đã nghiên cứu mô hình phân loại Naive Bayes về mức độ uy tín. Các biến đầu vào trong phương pháp này là các chỉ tiêu nhân khẩu học và vật chất. Một cách tiếp cận hiện đại để xác định mức độ tín nhiệm là phương pháp cây quyết định (Hand và cộng sự, 1997[10]). Szwabe và Misiorek (2018)[11] đã đề xuất mô hình cây quyết định để ra quyết định tín dụng.

Bayraci và Susuz (2019)[12] trong nghiên cứu của họ đã áp dụng Mạng thần kinh sâu (DNN) với nhiều lớp ẩn để đánh giá hồ sơ rủi ro của khách hàng cho vay trên bộ dữ liệu lấy từ một ngân hàng thương mại Thổ Nhĩ Kỳ. Họ đã so sánh khả năng dự đoán của phương pháp học sâu với Hồi quy logistic (LR), Cây quyết định, Naïve Bayes và Máy vectơ hỗ trợ (SVM). Kết quả chỉ ra rằng, mô hình DNN cải thiện hiệu suất của hệ thống chấm điểm tín dụng về độ chính xác cân bằng khi so sánh với các mô hình khác.

Trong đề tài nghiên cứu này, nhóm sẽ xây dựng ba mô hình để giải quyết bài toán đánh giá tín dụng cá nhân, đó là: Cây quyết định, hồi quy logistic và LightGBM. Trong chương 2, nhóm sẽ đưa ra khái niệm của ba mô hình, đồng thời sẽ giải thích lí do để chọn các mô hình đó.

1.4. Thách thức trong việc chấm điểm tín dụng

1.4.1. Vấn đề tổng lượng có khả năng vỡ nợ thấp.

Ở một số giai đoạn nhất định của chu kỳ kinh tế, số lượng người vỡ nợ có thể rất thấp, điều này làm phức tạp quá trình lập mô hình. Dữ liệu không cân bằng đề cập đến tình huống trong đó một lớp được đại diện ít hơn so với lớp kia. Trong tính điểm tín dụng, dữ liệu mất cân bằng là khá phổ biến do việc thiếu thống kê của người vỡ nợ và đây được gọi là vấn đề tổng lượng có khả năng vỡ nợ thấp. Trong bối cảnh mất cân bằng lớp dữ liệu, tổng lượng có khả năng vỡ nợ thấp được coi là một trường hợp hiếm hoi tuyệt đối.

1.4.2. Chấm điểm hành vi

Chấm điểm hành vi, được sử dụng sau khi tín dụng đã được cấp và ước tính khả năng vỡ nợ của khách hàng hiện tại trong một khoảng thời gian nhất định. Chấm điểm hành vi cho phép bên cho vay thường xuyên theo dõi khách hàng và giúp điều phối việc ra quyết định ở cấp độ khách hàng. Dữ liệu được sử dụng để điều chỉnh mô hình cho nhiệm vụ này dựa trên hiệu suất hoàn trả khoản vay của khách hàng và trạng thái tốt / xấu của họ vào một số ngày sau đó. Để có lợi nhuận, ngân hàng phải dự đoán

chính xác khả năng vỡ nợ của khách hàng trong các khoảng thời gian khác nhau (1 tháng, 3 tháng, 6 tháng, v.v.). Sau đó, những khách hàng có nguy cơ vỡ nợ cao có thể được gắn cờ cho phép ngân hàng thực hiện các hành động thích hợp để bảo vệ hoặc hạn chế tổn thất của bản thân.

Chấm điểm hành vi được các tổ chức sử dụng để hướng dẫn các quyết định cho vay đối với các khách hàng thân thiết trong: chiến lược quản lý hạn mức tín dụng; quản lý việc thu hồi và xử lý nợ; giữ chân khách hàng sinh lời trong tương lai; dự đoán các tài khoản có khả năng đóng hoặc tất toán sớm; cung cấp các sản phẩm tài chính mới; đưa ra lãi suất mới; quản lý tài khoản không hoạt động; tối ưu hóa hoạt động tiếp thị qua điện thoại; và dự đoán hoạt động gian lận.[10],[13],[14],[15],[16]

Hoàn cảnh tài chính của khách hàng có thể thay đổi theo thời gian và do đó, chúng được theo dõi và quản lý liên tục. Hệ thống chấm điểm hành vi đầu tiên để dự đoán rủi ro tín dụng của khách hàng hiện tại được Fair Isaac Inc. cho Wells Fargo phát triển vào năm 1975. Mô hình hành vi đã phát triển để ảnh hưởng đến các quyết định trong toàn bộ chu kỳ tín dụng. Ví dụ: mô hình sử dụng cho các sản phẩm thẻ tín dụng cố gắng dự đoán mức độ hoạt động trong tương lai để hỗ trợ các chiến lược duy trì và khích lệ tài chính. Mô hình quản lý tài khoản được bên cho vay sử dụng trong suốt thời gian tồn tại của tài khoản để dự đoán rủi ro vỡ nợ tại một thời điểm nhất định (ví dụ: hàng tháng, quý, năm). Điều này cho phép bên cho vay đặt giới hạn cho vay đối với các quyết định cho vay nạp tiền và thực hiện các biện pháp thích hợp để ngăn chặn các tài khoản xấu, mất mát. Thông tin như vậy cũng có giá trị đối với bộ phận tiếp thị của bên cho vay khi lựa chọn những khách hàng sinh lời cho các sản phẩm bổ sung hoặc quyết định mức độ khích lệ tài chính việc sử dụng tài khoản tăng lên.

Nói chung, có hai cách tiếp cận để chấm điểm hành vi: các kỹ thuật sử dụng các đặc điểm tĩnh về hiệu suất trong quá khứ của khách hàng; và các kỹ thuật kết hợp các khía cạnh động.

Tổng kết chương.

Như vậy trong chương thứ nhất, nhóm đã khái quát được bài toán đánh giá tín dụng cá nhân, chỉ ra những yêu cầu mà nghiên cứu cần đạt được. Ngoài ra còn đưa được những đề tài khác trong cùng phạm vi và những mô hình mà họ áp dụng trong bài toán đánh giá tín dụng cá nhân. Từ đó nêu ra những thách thức gặp phải khi giải

quyết bài toán. Trong chương tiếp theo sẽ là các khái niệm và kỹ thuật cơ bản mà nhóm sử dụng để xây dựng mô hình đánh giá tín dụng cá nhân trong nghiên cứu này.

CHƯƠNG 2: Các khái niệm và kỹ thuật cơ bản

2.1. Machine learning

Học máy là việc trích xuất kiến thức từ dữ liệu. Nó là một lĩnh vực nghiên cứu ở giao điểm của thống kê, trí tuệ nhân tạo và khoa học máy tính và còn được gọi là phân tích dự đoán hoặc học thống kê. Việc áp dụng các phương pháp học máy trong những năm gần đây đã trở nên phổ biến trong cuộc sống hàng ngày. Từ các đề xuất tự động về bộ phim nên xem, món ăn cần đặt hoặc sản phẩm cần mua, mạng xã hội trực tuyến được cá nhân hóa và nhận ra bạn bè trong ảnh của bạn, nhiều trang web và thiết bị hiện đại có các thuật toán máy học làm cốt lõi của chúng. Khi bạn nhìn vào một trang web phức tạp như Facebook, Amazon hoặc Netflix, rất có thể mọi phần của trang web đều chứa nhiều mô hình học máy.[17]

Trong đánh giá tín dụng cá nhân cũng cần đến các mô hình học máy để dự đoán dựa trên một tập dữ liệu của các khách hàng cho vay và phân loại khách hàng có khả năng vỡ nợ hay không có khả năng vỡ, vậy trước khi đánh giá tín dụng cá nhân sử dụng phương pháp nào và những bất cập của phương pháp truyền thống đấy sẽ được nói trong mục 2.2 sau đây.

2.2. Điểm tín dụng cá nhân

2.1.1. Chấm điểm tín dụng định lượng

Một thành viên của nhóm tốt được coi là có khả năng hoàn trả khoản vay của họ. Một thành viên của nhóm xấu được coi là có khả năng không trả được nợ. Mô hình đánh giá tín dụng bao gồm một tập hợp các đặc điểm được sử dụng để quy định điểm tín dụng đối với một khách hàng chỉ ra mức độ rủi ro của họ.[18] Điểm tín dụng này sau đó có thể được so sánh với một ngưỡng để đưa ra quyết định cho vay.

Dựa trên cả nhiệm vụ và dữ liệu được sử dụng, tính điểm tín dụng theo truyền thống được chia thành hai loại lớn[19]:

1) Điểm ứng dụng, được sử dụng tại thời điểm đơn xin tín dụng được thực hiện và ước tính khả năng vỡ nợ của bên đi vay trong một khoảng thời gian nhất định. Dữ liệu được sử dụng để điều chỉnh mô hình cho yêu cầu này thường bao gồm thông tin tài chính và nhân khẩu học về một mẫu của người đi vay trước đó cùng với tình trạng tốt/xấu của họ vào ngày hôm sau.

2) Chấm điểm hành vi, được sử dụng sau khi tín dụng đã được cấp và ước tính khả năng vỡ nợ của khách hàng hiện tại trong một khoảng thời gian nhất định. Chấm điểm hành vi cho phép bên cho vay thường xuyên theo dõi khách hàng và giúp điều phối việc ra quyết định ở cấp độ khách hàng. Dữ liệu được sử dụng để điều chỉnh mô hình cho yêu cầu này dựa trên khả năng chi trả khoản vay của khách hàng và tình trạng tốt / xấu của họ vào một số ngày sau.

Để có lợi nhuận, ngân hàng phải dự đoán chính xác khả năng vỡ nợ của khách hàng trong các khoảng thời gian khác nhau (1 tháng, 3 tháng, 6 tháng, ...). Sau đó, những khách hàng có nguy cơ vỡ nợ cao có thể được gắn cờ cho phép ngân hàng thực hiện các hành động thích hợp để bảo vệ hoặc hạn chế tổn thất.

2.2.1. Hoàn cảnh

Trong ngân hàng bán lẻ, trước khi sử dụng hệ thống chấm điểm tín dụng tự động, rủi ro tín dụng của người yêu cầu được đánh giá theo cách chủ quan dựa trên kinh nghiệm của người bảo lãnh. Thông thường, thông tin về khách hàng được thu thập thông qua các mối quan hệ cá nhân giữa khách hàng và nhân viên tại bên cho vay, điều này hạn chế sự linh động của khách hàng giữa các bên cho vay.[20] Cho vay thường là một quá trình phán đoán trong đó người bảo lãnh (thường là giám đốc ngân hàng) đánh giá các đơn đăng ký dựa trên các tiêu chí được gọi là 5C:

1) Nhân vật (Character) - người đi vay hoặc bất kỳ người nào trong gia đình của họ có được tổ chức biết đến không?
2) Vốn (Capital) - số tiền gửi mà bên đi vay đề nghị là bao nhiêu và số tiền vay được yêu cầu là bao nhiêu?
3) Tài sản thế chấp (Collateral) – người yêu cầu phát hành tín dụng đưa ra tài sản thế chấp là gì?
4) Năng lực (Capacity) - khả năng trả nợ của người yêu cầu là bao nhiêu?
5) Hoàn cảnh (Condition) – hoàn cảnh chung của nền kinh tế hiện nay là gì?

Rõ ràng, quy trình như vậy có một số thiếu sót, đặc biệt là về tính nhất quán và độ tin cậy - nói một cách dễ hiểu là chất lượng của các quyết định cấp tín dụng. Tác giả của bài báo “Modelling consumer credit risk” [21] đã liệt kê ra những thiếu sót chính sau đây: (i) những quyết định như vậy chắc chắn bị ảnh hưởng bởi những thay đổi hàng ngày trong tâm trạng của người quản lý ngân hàng; (ii) các quyết định không phải lúc nào cũng có thể lặp lại vì các nhà quản lý khác nhau không phải lúc nào cũng đưa ra các quyết định giống nhau; (iii) không có quy luật chung của quá trình đưa ra

quyết định, gây khó khăn trong quá trình truyền đạt lại; (iv) phương pháp đánh giá dựa trên con người chỉ có thể xử lý một số đơn xin nhất định, dẫn đến mất doanh thu.

Máy tính cung cấp các công cụ cần thiết để thực hiện các quy trình tự động hỗ trợ ngân hàng trong việc chấm điểm tín dụng phát triển theo phương pháp thống kê.[21] So với việc phán đoán trước đây, điều này giúp cho các ngân hàng bán lẻ đưa ra báo cáo về việc giảm đáng kể chi phí đánh giá tín dụng và tổn thất khoản vay do khách hàng vỡ nợ.[22] Vào những năm 1980 với những cải tiến về sức mạnh tính toán (ví dụ: chi phí, tốc độ và dung lượng lưu trữ), các ngân hàng bán lẻ bắt đầu sử dụng các phương pháp thống kê để theo dõi, đo lường, phát hiện, dự đoán và hiểu nhiều khía cạnh của hành vi khách hàng.[21] Dần dần điều này dẫn đến sự phát triển của các kỹ thuật ước lượng, trong số những thứ khác [21], [23]:

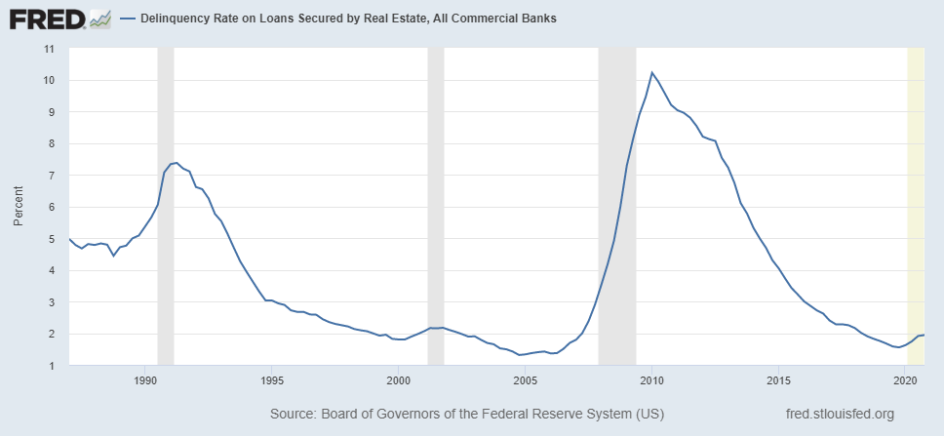
1. Rủi ro vỡ nợ - đo lường rủi ro khách hàng vỡ nợ đối với một sản phẩm cụ thể (tính điểm vỡ nợ sản phẩm) hoặc đối với bất kỳ sản phẩm nào (tính điểm vỡ nợ của khách hàng).[10]
2. Phát hiện gian lận - các kỹ thuật có thể phát hiện gian lận càng sớm càng tốt.[24]
3. Phản hồi đối với các chiến dịch quảng cáo - liệu khách hàng có trả lời thư trực tiếp về một sản phẩm mới không? [25]
4. Giữ chân khách hàng - liệu khách hàng có tiếp tục sử dụng sản phẩm sau khi hết thời gian dùng thử ban đầu không?[26]
5. Tiêu hao - liệu khách hàng có đổi sang bên cho vay khác không?[27]
6. Sử dụng sản phẩm - khách hàng sẽ sử dụng một sản phẩm nhất định, và nếu có, với cường độ nào?
7. Tính điểm lợi nhuận - kỹ thuật đo lường khả năng sinh lời của khách hàng trên một sản phẩm (tính điểm lợi nhuận sản phẩm) và trên tất cả các sản phẩm (tính điểm lợi nhuận khách hàng).

Thông thường, hệ thống chấm điểm tín dụng được triển khai bằng thẻ điểm tín dụng. Thẻ điểm chỉ định điểm cho các đặc điểm chính của khách hàng và các khía cạnh của giao dịch để tính ra một giá trị là số để thể hiện rủi ro mà một khách hàng so với các khách hàng khác sẽ vỡ nợ.

Hệ thống chấm điểm tín dụng không hoàn hảo và chỉ có thể ước tính rủi ro tín dụng dựa trên hiệu suất trong quá khứ, không phải tương lai. Hàng năm, do hệ thống chấm điểm tín dụng không xác định được những cá nhân không trả được nợ, một phần đáng kể của món nợ không được thanh toán.[28] Thông thường, nguyên nhân của điều này có thể được cho là do các trường hợp không lường trước được như:

1. Gian lận.
2. Ly hôn.
3. Thiếu nhạy bén về tài chính.
4. Nợ do mất thu nhập.

Tỷ lệ quá hạn (tức là khách hàng không hoàn trả khoản vay của họ) đối với các khoản vay, ví dụ với hộ gia đình ở Hoa Kỳ được hiển thị trong Hình 2-1. Do đó có lợi ích đáng kể trong việc cải thiện hệ thống chấm điểm tín dụng để phân biệt giữa khách hàng sinh lời và khách hàng không sinh lời trên cơ sở hành vi trả nợ của họ trong tương lai.[28] Giữa những người trong ngành và các nhà nghiên cứu, người ta chỉ rằng ngay cả một sự cải thiện nhỏ trong việc đánh giá rủi ro tín dụng của khách hàng cũng có thể dẫn đến tiết kiệm đáng kể về mặt tài chính.[10]



Hình 2- 1: Tỷ lệ quá hạn đối với tất cả các khoản vay bất động sản, tất cả các ngân hàng, được điều chỉnh theo quý.

Nguồn: Federal Reserve Board.

2.2.2. Hiệp ước vốn Basel II

Cùng với việc tiết kiệm tài chính bằng cách tính điểm tín dụng, cũng có những vấn đề pháp lý cần tuân thủ. Ở một số quốc gia, ngân hàng trung ương chịu trách nhiệm giám sát ngân hàng, trong khi các quốc gia khác có các cơ quan quản lý riêng biệt —và đôi khi có nhiều cơ quan quản lý để giám sát ngân hàng.[29] Ngân hàng Thanh toán Quốc tế là một tổ chức quốc tế có nhiệm vụ thúc đẩy hợp tác tài chính và tiền tệ quốc tế giữa các ngân hàng trung ương. Ủy ban Basel về Giám sát Ngân hàng (BCBS) là một tiểu ban của BIS chịu trách nhiệm phát triển các hướng dẫn và khuyến nghị về các quy định ngân hàng áp dụng cho tất cả các quốc gia thành viên. Nhóm đề xuất đầu tiên, Basel I, đã phát triển một bộ tiêu chuẩn thống nhất về mức vốn điều tiết và chủ yếu tập trung vào rủi ro tín dụng.[30] Basel I được xuất bản lần đầu tiên vào

năm 1988 và được triển khai ở 12 quốc gia vào năm 1992. Cuối cùng, nó đã được chấp nhận rộng rãi vì các ngân hàng tuân thủ nhận được xếp hạng tín dụng được cải thiện và chi phí tài trợ thấp hơn.[20] Cách tính vốn theo quy định của Basel I sử dụng một bộ quy tắc đơn giản để chỉ định trọng số rủi ro cho một loại tài sản hoặc khoản vay nhất định. Basel I sử dụng bốn loại tài sản lớn (chính phủ, ngân hàng, công ty và cá nhân) có trọng số rủi ro khác nhau gắn liền với chúng. Bốn trọng số rủi ro cơ bản đã được xác định, cùng với một hạng mục bổ sung có trọng số theo quyết định của cơ quan quản lý quốc gia, bao gồm [31]:

- 0% (ví dụ: nợ công).
- 20% (ví dụ: nợ từ các ngân hàng khác hoặc các tổ chức khu vực công).
- 50% (ví dụ: cho vay bất động sản nhà ở).
- 100% (ví dụ: các khoản vay cho các công ty khu vực tư nhân).
- 0%, 10%, 20% hoặc 50% theo quyết định của cơ quan quản lý.

Sau khi áp dụng trọng số rủi ro cho từng loại tài sản, tổng số tiền cho vay của các loại tài sản được tính toán để cung cấp tổng tài sản có trọng số rủi ro (RWA). Tỷ lệ vốn bắt buộc được đặt tối thiểu là 8% RWA.

Một chỉ trích phổ biến đối với Basel I là nó thiếu tính nhạy cảm với rủi ro, khiến các ngân hàng có quá nhiều sự linh hoạt trong việc kiểm soát RWA của họ thông qua chênh lệch giá theo quy định. John Holman [32] định nghĩa chênh lệch giá theo quy định là một hoạt động mà trong đó "một ngân hàng khai thác sự khác biệt giữa mức độ rủi ro thực tế của họ và mức độ rủi ro theo quy định của ngân hàng". Điều này được thực hiện bằng cách sử dụng các đổi mới tài chính phức tạp (ví dụ: chứng khoán hóa và các phái sinh tín dụng) cho phép các ngân hàng giảm vốn bắt buộc tối thiểu của họ mà không thực sự giảm rủi ro liên quan. Để đáp lại điều này và những chỉ trích khác đối với Basel I (xem Balin, 2008), tiêu chuẩn Basel II sau đó bắt đầu phát triển từ năm 1999 cho đến khi công bố khung của nó vào giữa năm 2004.

2.2.3. Basel II dựa trên ba trụ cột củng cố lẫn nhau

- Liên quan tới việc duy trì vốn bắt buộc. Theo đó, tỷ lệ vốn bắt buộc tối thiểu (CAR) vẫn là 8% của tổng tài sản có rủi ro như Basel I. Tuy nhiên, rủi ro được tính toán theo ba yếu tố chính mà ngân hàng phải đối mặt: rủi ro tín dụng, rủi ro vận hành (hay rủi ro hoạt động) và rủi ro thị trường. So với Basel I, cách tính chi phí vốn đối với rủi ro tín dụng có sự sửa đổi lớn, đối với rủi ro thị trường có sự thay đổi nhỏ, nhưng hoàn toàn là phiên bản mới đối với rủi ro vận hành.

Trọng số rủi ro của Basel II bao gồm nhiều mức (từ 0%-150% hoặc hơn) và rất nhạy cảm với xếp hạng.

- Liên quan tới việc hoạch định chính sách ngân hàng, Basel II cung cấp cho các nhà hoạch định chính sách những "công cụ" tốt hơn so với Basel I. Trụ cột này cũng cung cấp một khung giải pháp cho các rủi ro mà ngân hàng đối mặt, như rủi ro hệ thống, rủi ro chiến lược, rủi ro danh tiếng, rủi ro thanh khoản và rủi ro pháp lý, mà hiệp ước tổng hợp lại dưới cái tên rủi ro còn lại (residual risk).
- Các ngân hàng cần phải công khai thông tin một cách thích đáng theo nguyên tắc thị trường. Basel II đưa ra một danh sách các yêu cầu buộc các ngân hàng phải công khai thông tin, từ những thông tin về cơ cấu vốn, mức độ đầy đủ vốn đến những thông tin liên quan đến mức độ nhạy cảm của ngân hàng với rủi ro tín dụng, rủi ro thị trường, rủi ro vận hành và quy trình đánh giá của ngân hàng đối với từng loại rủi ro này.

Trong trụ cột 1, việc tính toán vốn tối thiểu bắt buộc đối với rủi ro tín dụng có thể được thực hiện bằng cách sử dụng các phương pháp luận từ một chuỗi liên tục ngày càng phức tạp và nhạy cảm với rủi ro:

- Phương pháp tiếp cận tiêu chuẩn hóa.
- Phương pháp tiếp cận dựa trên đánh giá nội bộ (IRB).
  - Phương pháp tiếp cận cơ sở.
  - Phương pháp tiên tiến.

Theo cách tiếp cận tiêu chuẩn hóa, các ngân hàng sử dụng xếp hạng do các tổ chức xếp hạng tín dụng bên ngoài cung cấp để lượng hóa các yêu cầu vốn đối với rủi ro tín dụng. Tương tự như khuôn khổ Basel I, phương pháp tiêu chuẩn hóa sử dụng phương pháp tiếp cận theo trọng số rủi ro. Để tăng độ nhạy với rủi ro, việc phân loại các loại tài sản được xác định chi tiết hơn. Thông qua việc khuyến khích nắm giữ dự trữ vốn thấp hơn, các phương pháp tiếp cận IRB khuyến khích các ngân hàng xây dựng xếp hạng rủi ro nội bộ của riêng họ để có thể đo lường rủi ro tín dụng thực tế của ngân hàng. Trong cách tiếp cận IRB, cả cách tiếp cận cơ sở và tiên tiến đều dựa trên bốn thành phần chính:

- Xác suất vỡ nợ - Probability of default (PD): là khả năng một sự kiện vỡ nợ sẽ xảy ra. PD được sử dụng như một thước đo về khả năng và mức độ sẵn sàng trả nợ của người đi vay.
- Tỉ trọng tổn thất ước tính - Loss given default (LGD): được định nghĩa là tổn thất kinh tế dự kiến phát sinh trong trường hợp người đi vay không trả được nợ. LGD

thường được biểu thị bằng tỷ lệ phần trăm của tổng số tổn thất tiềm năng tại thời điểm khách hàng không có khả năng trả nợ. Trong trường hợp không có tổn thất, LGD bằng không. Nếu ngân hàng mất toàn bộ số giá trị chịu rủi ro, LGD sẽ bằng 100%. LGD âm sẽ cho thấy một khoản lợi nhuận (ví dụ: do các khoản phí phạt đã trả và tiền lãi do truy thu). LGD làm tăng tỷ lệ thu hồi kỳ hạn, có thể được biểu thị bằng 1 – LGD.

- Tổng dư nợ tại thời điểm khách hàng không trả được nợ - Exposure at default (EAD): là tổng giá trị ngân hàng có thể bị tổn thất khi khoản vay mất khả năng trả nợ.
- Thời gian đáo hạn hiệu quả (M) là khoảng thời gian trước khi khoản vay được trả hết.

Theo nguyên tắc chung, đối với cách tiếp cận cơ sở, các ngân hàng cung cấp ước tính PD của riêng mình cho từng loại tài sản, nhưng sử dụng ước tính do các cơ quan quản lý cung cấp cho các thành phần rủi ro khác. Đối với phương pháp tiên tiến, các ngân hàng phải tính toán thời gian đáo hạn hiệu quả và cung cấp các ước tính của riêng họ về PD, LGD và EAD. Tuy nhiên, cần quy định rằng đối với các khoản vay bán lẻ không có sự phân biệt giữa cách tiếp cận cơ sở và tiên tiến, và các ngân hàng phải cung cấp các ước tính của riêng họ về PD, LGD, EAD.

Cần phân biệt giữa PD của một khoản vay cá nhân và PD của một danh mục cho vay, PD của một khoản vay cá nhân có thể được ước tính bằng cách sử dụng một mô hình phân loại, chẳng hạn như Logistic Regression, Support Vector Machines, k-Nearest Neighbour,… Ví dụ Logistic Regression sử dụng điểm log(odd) để dự báo về PD của từng người đi vay.

Sau đó, PD quan sát được sử dụng để gán khách hàng vào một lớp xếp hạng cụ thể. Bằng cách nhóm các khoản vay riêng lẻ, có PD tương tự nhau, thành các loại xếp hạng, có thể xác định được ước tính chính xác và nhất quán về PD của danh mục khoản vay.

Một danh mục cho vay bao gồm các khoản vay riêng lẻ được nhóm lại với nhau thành các nhóm đồng nhất. Thông thường, việc phân đoạn danh mục cho vay bán lẻ được thực hiện.[33] bởi: sản phẩm, kênh chuyển đổi, điểm tín dụng, vị trí địa lý hoặc khoản vay theo giá trị Các nhà cho vay có thể phân khúc danh mục đầu tư xa hơn theo các dải PD và đôi khi, các dải LGD (tức là loại xếp hạng).[23] Bên cho vay ước tính PD của từng loại xếp hạng, với số lần vỡ nợ dự kiến chia cho số lượng khách hàng.

Thông qua việc sử dụng các danh mục cho vay, bên cho vay có thể sử dụng quá trình chứng khoán hóa, theo đó các tài sản kém thanh khoản như các khoản thế chấp được chuyển thành chứng khoán thị trường. Chứng khoán được bán cho một công ty mua bán đặc biệt của bên thứ ba, bên này sau đó sẽ phát hành trái phiếu trong đó các khoản hoàn trả khoản vay được sử dụng để trang trải cho việc hoàn trả các phiếu mua hàng và tiền gốc của trái phiếu. Thông qua chứng khoán hóa, bên cho vay có thể giảm quy mô bảng cân đối kế toán của họ, dẫn đến yêu cầu vốn thấp hơn.

Sau khi những khiếm khuyết trong Basel II được bộc lộ bởi cuộc khủng hoảng tài chính năm 2008, một bản sửa đổi (Basel III) đã được bắt đầu và việc thực hiện các hướng dẫn từ đầu năm 2013. Basel III mở rộng các công việc hiện có trong Basel II bằng cách tăng cường các yêu cầu về vốn và đưa ra các yêu cầu về tính linh động và tỷ số vay vốn. Do đó, các tổ chức tài chính phải duy trì vùng đệm vốn cao hơn để họ ít bị khủng hoảng hơn và cần các gói cứu trợ của chính phủ. Tóm lại, theo Hiệp ước vốn Basel II, sử dụng phương pháp tiếp cận dựa trên xếp hạng nội bộ, các ngân hàng có thể tính toán các yêu cầu về vốn của họ bằng cách sử dụng dữ liệu nội bộ của họ để xây dựng các mô hình rủi ro tín dụng. Do kết quả của phương pháp này, người ta chú trọng nhiều hơn vào việc ước tính chính xác xác suất vỡ nợ (PD) của khách hàng hơn là xếp hạng khách hàng một cách chính xác dựa trên rủi ro vỡ nợ của họ.[13] PD là "khái niệm đo lường trung tâm mà phương pháp tiếp cận IRB được xây dựng".[34] PD cũng phải được dự đoán không chỉ ở cấp độ cá nhân mà còn đối với các phân đoạn của danh mục cho vay. Một danh mục cho vay bao gồm các khoản vay được phân chia thành các nhóm xếp hạng và PD được ước tính cho các khách hàng trong mỗi nhóm. Mô hình hóa PD ở mức cho vay về cơ bản là một vấn đề phân biệt (tốt hay xấu), do đó người ta có thể sử dụng nhiều kỹ thuật phân loại đã được đề xuất trong tài liệu. Nhiều mô hình phân loại này có nguồn gốc từ phương pháp thống kê, phương pháp phi tham số và phương pháp tiếp cận trí tuệ nhân tạo. Bằng cách ước tính PD ở cấp tài khoản và sau đó ở cấp danh mục đầu tư, bên cho vay có thể ước tính tổn thất (hoặc rủi ro tín dụng) liên quan đến một danh mục cho vay cụ thể.

Mục đích của phần này là giới thiệu rủi ro tín dụng bán lẻ bằng cách mô tả các yếu tố chính đằng sau sự thành lập và tăng trưởng vượt bậc của nó trong nửa sau của thế kỷ XX. Phần tiếp theo sẽ nói về dạng dữ liệu và những vấn đề về dữ liệu xấu gây ảnh hưởng đến kết quả dự đoán, các cách tiền xử lý về làm sạch dữ liệu để áp dụng mô hình học máy cho một kết quả dự đoán chính xác hơn.

2.3. Dữ Liệu

2.3.1. Loại tập dữ liệu

Đặc điểm chung của tập dữ liệu:

Chiều không gian (Dimensionality): phân tích dữ liệu với một số chiều nhỏ có chất lượng tốt hơn là phân tích trung bình cộng hoặc dữ liệu với số chiều lớn. Vì vậy trước khi sử lí dữ liệu phải giảm bớt số chiều (dimensionality reduction) của dữ liệu.[35]

Phân bố (Distribution): Phân bố của một tập dữ liệu là tần suất xuất hiện của các giá trị khác nhau hoặc những tập giá trị cho các thuộc tính bao gồm đối tượng dữ liệu. Và cũng được coi như là một mô tả về sự tập trung của các đối tượng trong các vùng khác nhau của không gian dữ liệu. Sự sai lệnh (skewness) trong phân bố gây khó khăn cho việc phân loại. Và một trường hợp đặc biệt của dữ liệu sai lệch là sự thưa thớt, thường các thuộc tính trong tập dữ liệu thưa thớt là thuộc tính không đối xứng.[35]

Độ phân giải (Resolution): Thường có thể lấy dữ liệu ở các mức độ phân giải khác nhau và thường các thuộc tính của dữ liệu khác nhau ở các độ phân giải khác nhau. Các mẫu trong dữ liệu phụ thuộc vào mức độ phân giải. Nếu độ phân giải quá tốt, một mẫu có thể không hiển thị hoặc có thể bị nhiễu; nếu độ phân giải quá thô, một số mẫu có thể biến mất.[35]

Ghi lại dữ liệu(Record Data): Tập dữ liệu là một tập hợp các trường hợp được ghi lại (đối tượng dữ liệu), mỗi đối tượng dữ liệu ấy bao gồm một tập cố định của các trường dữ liệu hay chính là các thuộc tính của đối tượng đó. Đối với dạng dữ liệu bản ghi cơ bản nhất, không có mối quan hệ rõ ràng giữa các bản ghi hoặc trường dữ liệu. Dữ liệu bản ghi thường được lưu dưới dạng flat files[1] hoặc in relational database[2].[35]

2.3.2. Chất lượng dữ liệu

Các thuật toán khai thác dữ liệu thường được áp dụng cho dữ liệu được thu thập cho một mục đích khác hoặc cho các ứng dụng trong tương lai, nhưng không xác định. Vì lý do đó, khai thác dữ liệu thường không thể tận dụng được những lợi ích đáng kể của việc "giải quyết các vấn đề chất lượng tại nguồn". Ngược lại, phần lớn số liệu thống

---

[1] Cơ sở dữ liệu tệp phẳng là cơ sở dữ liệu được lưu trữ trong một tệp gọi là tệp phẳng. Các bản ghi tuân theo một định dạng thống nhất và không có cấu trúc để lập chỉ mục hoặc nhận biết mối quan hệ giữa các bản ghi. Các tập tin là đơn giản.[45]

[2] Cơ sở dữ liệu quan hệ là một loại cơ sở dữ liệu lưu trữ và cung cấp quyền truy cập vào các điểm dữ liệu có liên quan đến nhau. Cơ sở dữ liệu quan hệ dựa trên mô hình quan hệ, một cách trực quan, đơn giản để biểu diễn dữ liệu trong bảng. Trong cơ sở dữ liệu quan hệ, mỗi hàng trong bảng là một bản ghi với một ID duy nhất được gọi là khóa. Các cột của bảng chứa các thuộc tính của dữ liệu và mỗi bản ghi thường có một giá trị cho mỗi thuộc tính, giúp dễ dàng thiết lập mối quan hệ giữa các điểm dữ liệu.[46]

kê liên quan đến việc thiết kế các thử nghiệm hoặc khảo sát đạt được mức chất lượng dữ liệu được chỉ định trước.[35]

Bởi vì việc ngăn chặn các vấn đề về chất lượng dữ liệu thường không phải là một lựa chọn, khai thác dữ liệu tập trung vào việc:

- Phát hiện và sửa chữa các vấn đề về chất lượng dữ liệu.
- Sử dụng các thuật toán có thể chịu được chất lượng dữ liệu kém.
- Bước đầu tiên, phát hiện và sửa chữa thường được gọi là làm sạch dữ liệu.

Các vấn đề về đo lường và thu thập dữ liệu

Chúng tôi bắt đầu với định nghĩa về sai số đo lường và thu thập dữ liệu, sau đó xem xét nhiều vấn đề liên quan đến sai số đo lường: nhiễu, hiện vật, độ chệch, độ chính xác. Xem xét về các vấn đề chất lượng dữ liệu liên quan đến cả vấn đề đo lường và thu thập dữ liệu: giá trị ngoại lệ, giá trị bị thiếu và không nhất quán cũng như dữ liệu trùng lặp.[35]

Lỗi đo lường và thu thập dữ liệu:

Thuật ngữ đo lường sai số (measurement error) đề cập đến bất kỳ vấn đề nào phát sinh từ quá trình đo lường. Một vấn đề phổ biến là giá trị được ghi lại khác với giá trị thực ở một mức độ nào đó. Đối với các thuộc tính liên tục, sự khác biệt về số của giá trị đo được và giá trị thực được gọi là sai số. Thuật ngữ lỗi thu thập dữ liệu đề cập đến các lỗi như bỏ qua các đối tượng dữ liệu hoặc giá trị thuộc tính hoặc bao gồm một đối tượng dữ liệu một cách không thích hợp.[35]

Những thành phần ngoại lai(Outliers):

Các đối tượng ngoại lai là

(1) các đối tượng dữ liệu, theo một nghĩa nào đó, có các đặc điểm khác với hầu hết các đối tượng dữ liệu khác trong tập dữ liệu.[35]

(2) các giá trị của một thuộc tính khác thường so với các giá trị điển hình của thuộc tính.

Phân biệt giữa các khái niệm về nhiễu và ngoại lệ. Không giống như nhiễu, các giá trị ngoại lai có thể là các đối tượng hoặc giá trị dữ liệu hợp pháp mà chúng ta muốn phát hiện. Ví dụ trong phát hiện gian lận và xâm nhập mạng, mục tiêu là tìm ra các đối tượng hoặc sự kiện bất thường từ một số lượng lớn các sự kiện bình thường.

Dữ liệu bị mất

Không có gì lạ khi một đối tượng bị thiếu một hoặc nhiều giá trị thuộc tính. Trong một số trường hợp, thông tin không được thu thập đầy đủ. Dù vậy, các giá trị bị thiếu cần được tính đến trong quá trình phân tích dữ liệu.[35]

Ước tính các giá trị bị thiếu:

Đôi khi dữ liệu bị thiếu có thể được ước tính một cách đáng tin cậy. Ví dụ khác, hãy xem xét một tập dữ liệu có nhiều điểm dữ liệu giống nhau. Trong tình huống này, các giá trị thuộc tính của các điểm gần nhất với điểm có giá trị bị thiếu thường được sử dụng để ước tính giá trị bị thiếu. Nếu thuộc tính là liên tục, thì giá trị thuộc tính trung bình của các hàng xóm gần nhất được sử dụng; nếu thuộc tính là phân loại, thì giá trị thuộc tính phổ biến nhất có thể được. Để có một minh họa cụ thể, hãy xem xét các phép đo lượng mưa được các trạm tại mặt đất ghi lại. Đối với các khu vực không có trạm, lượng mưa có thể được ước tính bằng cách sử dụng các giá trị quan sát được tại các trạm gần đó.[35]

2.3.3. Tiền xử lý dữ liệu

Lấy mẫu

Lấy mẫu là một cách tiếp cận thường được sử dụng để chọn một tập hợp con của các đối tượng dữ liệu được phân tích. [35]

Lí do lấy mẫu trong thống kê và khai thác dữ liệu thường khác nhau. Các nhà thống kê sử dụng phương pháp lấy mẫu vì việc thu thập toàn bộ tập dữ liệu quan tâm là quá đắt hoặc tốn thời gian, trong khi người khai thác dữ liệu thường lấy mẫu vì quá tốn kém về mặt tính toán về bộ nhớ hoặc thời gian cần thiết để xử lý tất cả dữ liệu. Trong một số trường hợp, sử dụng thuật toán lấy mẫu có thể giảm kích thước dữ liệu đến mức có thể sử dụng thuật toán tốt hơn, nhưng tốn kém hơn về mặt tính toán.[35]

Nguyên tắc chính để lấy mẫu hiệu quả là: Sử dụng một mẫu sẽ hoạt động gần như tốt như sử dụng toàn bộ tập dữ liệu nếu mẫu là đại diện. Ngược lại, một mẫu là đại diện nếu nó có cùng thuộc tính (được quan tâm) như tập dữ liệu ban đầu. Nếu giá trị trung bình (trung bình cộng) của các đối tượng dữ liệu là thuộc tính quan tâm, thì một mẫu là đại diện nếu nó có giá trị trung bình gần với giá trị của dữ liệu gốc. Bởi vì lấy mẫu là một quá trình thống kê, tính đại diện của bất kỳ mẫu cụ thể nào sẽ khác nhau và cách tốt nhất mà chúng ta có thể làm là chọn một sơ đồ lấy mẫu đảm bảo xác suất cao để lấy được mẫu đại diện.[35]

Phương pháp lấy mẫu:

Khi tổng thể tập dữ liệu bao gồm các loại đối tượng khác nhau, với số lượng đối tượng khác nhau, thì việc lấy mẫu ngẫu nhiên đơn giản có thể không thể hiện đầy đủ những loại đối tượng ít thường xuyên xuất hiện. Điều này có thể gây ra vấn đề khi phân tích yêu cầu đại diện thích hợp của tất cả các loại đối tượng.[35]

Giai đoạn học tập và dự đoán tiếp theo của các thuật toán học máy có thể bị ảnh hưởng bởi vấn đề tập dữ liệu không cân bằng. Vấn đề cân bằng tương ứng với sự khác biệt của số lượng mẫu ở các lớp khác nhau.[36]

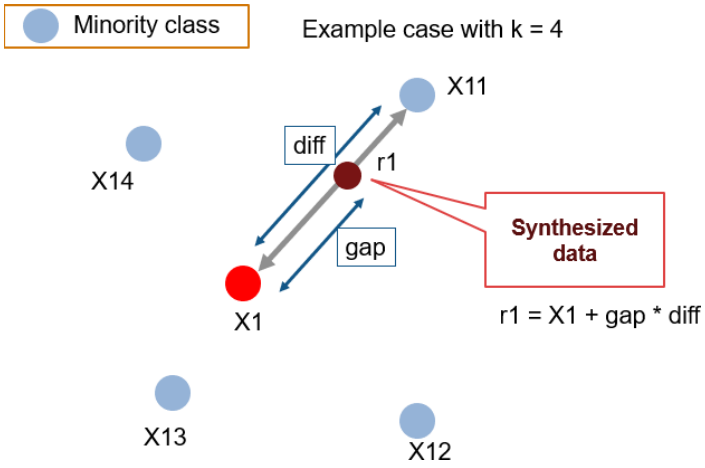
Ví dụ, khi xây dựng mô hình phân loại cho các tập dữ liệu phân loại với các lớp không cân đối. Ngoài việc chọn mẫu ngẫu nhiên có thay thế, có hai phương pháp phổ biến để lấy mẫu quá mức các nhóm thiểu số: chúng tôi sẽ sử dụng thực hiện lấy mẫu quá mức (oversampling) bằng SMOTE nếu tập dữ liệu có lớp hiếm hay còn gọi là nhóm thiểu số. SMOTE tạo ra các mẫu mới bằng cách nội suy tức là thêm vào, chèn vào. Việc triển khai cơ bản của SMOTE sẽ không tạo ra bất kỳ sự phân biệt nào giữa các mẫu dễ và khó được phân loại bằng cách sử dụng quy tắc láng giềng gần nhất (nearest neighbors).[36]

Công thức toán học của phương pháp SMOTE

Quy trình làm việc:

Lúc đầu, không có tổng số quan sát lấy mẫu quá mức, N được thiết lập. Nói chung, nó được chọn sao cho phân phối lớp nhị phân là 1: 1. Nhưng điều đó có thể được điều chỉnh dựa trên nhu cầu. Sau đó, quá trình lặp bắt đầu bằng cách chọn ngẫu nhiên một cá thể lớp positive tức là nhãn 1 trong tập dữ liệu mà chúng tôi làm việc Tiếp theo, KNN’s (theo mặc định là 5) cho trường hợp đó sẽ được lấy. Cuối cùng, N trong số K cá thể này được chọn để nội suy các cá thể tổng hợp mới. Để làm điều đó, sử dụng bất kỳ số liệu khoảng cách nào, sự khác biệt về khoảng cách giữa vectơ đối tượng và các vùng lân cận của nó sẽ được tính toán. Bây giờ, sự khác biệt này được nhân với bất kỳ giá trị ngẫu nhiên nào trong (0,1] và được thêm vào vectơ đặc trưng trước đó. Điều này được biểu diễn bằng hình ảnh bên dưới:

Synthesized data là dữ liệu tổng hợp. Minority class là lớp thiểu số tức là nhãn 1 trong bộ dữ liệu.

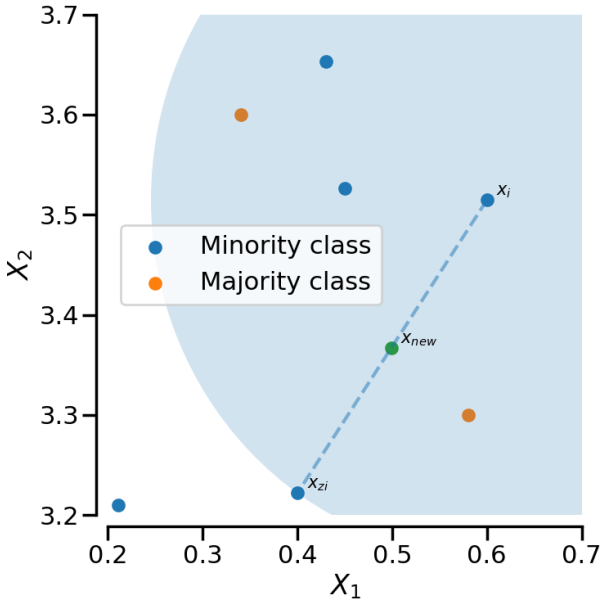


Hình 2- 2: Ví dụ minh họa về cách tạo một mẫu mới hay cá thể mới bằng phương pháp SMOTE

Tạo mẫu: xem xét một mẫu x_i, một mẫu mới x_{new} sẽ được tạo ra xem xét k lân cận láng giềng gần nhất của nó (tương ứng với k_neighbors). Sau đó, một trong những người hàng xóm gần nhất này x_{zi} được chọn và một mẫu được tạo như sau:

$$x_{new} = x_i + \lambda \times (x_{zi} - x_i) \quad (1)$$

Trong đó λ là một số ngẫu nhiên trong phạm vi [0, 1]. Phép nội suy này sẽ tạo ra một mẫu trên dòng giữa x_i và x_{zi} như được minh họa trong hình dưới đây



Hình 2- 3: Hình minh hoa giữa x_i và x_{zi}

Lựa chọn tập hợp con feature

Một cách khác để giảm số chiều là chỉ sử dụng một tập hợp con của các feature. Mặc dù có vẻ như cách tiếp cận như vậy sẽ làm mất thông tin, nhưng đây không phải là trường hợp nếu có các feature thừa và không liên quan. Thừa các feature bản sao nhiều hoặc tất cả thông tin có trong một hoặc nhiều thuộc tính khác. Các thuộc tính không liên quan hầu như không chứa thông tin hữu ích cho nhiệm vụ khai thác dữ liệu.[35] Ví dụ: số ID của khách hàng không liên quan đến nhiệm vụ dự đoán điểm tính dụng.

Có ba cách tiếp cận tiêu chuẩn để lựa chọn thuộc tính: nhúng, bộ lọc và trình bao bọc.[35]

Phương pháp nhúng: bản thân thuật toán quyết định sử dụng thuộc tính nào và bỏ qua thuộc tính nào. Như là các thuật toán để xây dựng bộ phân loại cây quyết định

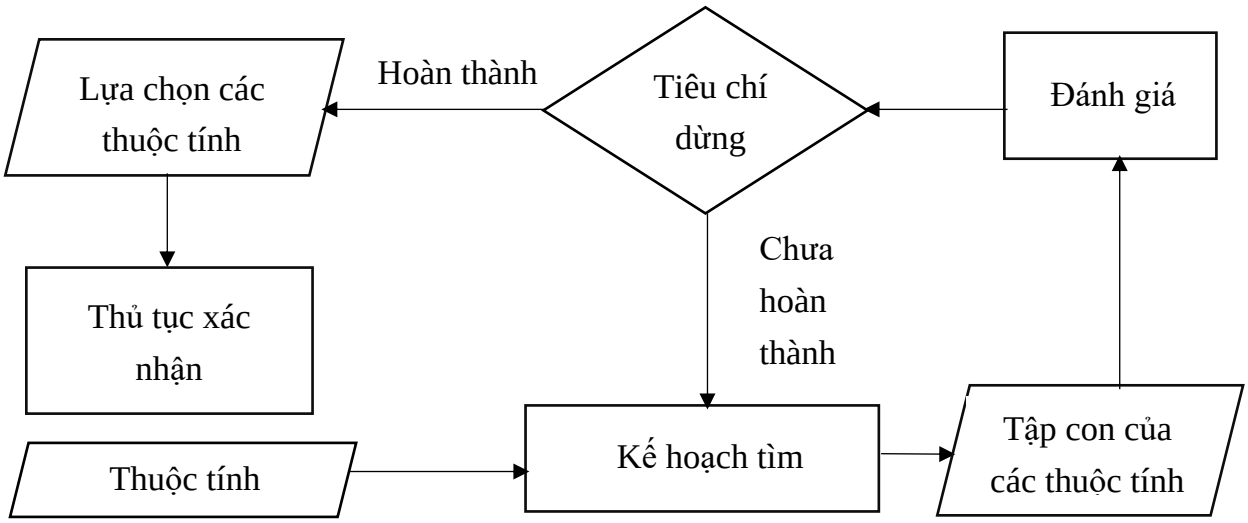
Bộ lọc: Các feature được chọn trước khi chạy thuật toán khai thác dữ liệu, sử dụng một số phương pháp tiếp cận độc lập với tác vụ khai thác dữ liệu.[35]

Phương pháp gói-bao bọc: các phương pháp này sử dụng thuật toán khai thác dữ liệu mục tiêu như một hộp đen để tìm tập hợp con tốt nhất của các thuộc tính. Nhưng thường không liệt kê tất cả các tập hợp con có thể có.[35]

Kiến trúc để lựa chọn tập hợp con feature:

Quá trình lựa chọn feature được xem như bao gồm bốn phần: thước đo để đánh giá một tập hợp con, chiến lược tìm kiếm kiểm soát việc tạo ra một tập hợp con các feature mới, tiêu chí dừng và thủ tục xác nhận.[35]

Phương pháp bộ lọc và phương pháp trình gói chỉ khác nhau về cách chúng đánh giá một tập hợp con các feature. Đối với phương pháp gói, đánh giá tập hợp con sử dụng thuật toán khai thác dữ liệu mục tiêu, trong khi đối với cách tiếp cận bộ lọc, kỹ thuật đánh giá này khác biệt với thuật toán khai thác dữ liệu mục tiêu.[35]



Hình 2- 4: Sơ đồ của quá trình lựa chọn tập hợp con các feature.

Chiến lược tìm kiếm phải không tốn kém về mặt tính toán và nên tìm các bộ feature tối ưu hoặc gần tối ưu. Thông thường không thể thỏa mãn cả hai yêu cầu và do đó, cần phải đánh đổi. Một phần không thể thiếu của việc tìm kiếm là một bước đánh giá để đánh giá xem tập hợp con hiện tại của các feature so với những thuộc tính khác đã được xem xét như thế nào. Điều này yêu cầu một biện pháp đánh giá cố gắng xác định mức độ tốt của một tập hợp con các thuộc tính đối với một nhiệm vụ khai thác dữ liệu cụ thể, chẳng hạn như phân loại hoặc phân cụm.[35]

Đối với cách tiếp cận bộ lọc, các biện pháp như vậy cố gắng dự đoán thuật toán khai thác dữ liệu thực tế sẽ hoạt động tốt như thế nào trên một tập hợp các thuộc tính nhất định.[35]

Đối với cách tiếp cận phương pháp gói, trong đó việc đánh giá bao gồm việc thực sự chạy thuật toán khai thác dữ liệu đích, chức năng đánh giá tập hợp con chỉ đơn giản là tiêu chí thường được sử dụng để đo lường kết quả của việc khai thác dữ liệu.[35]

Bởi vì số lượng các tập hợp con có thể rất lớn và việc kiểm tra tất cả chúng là không thực tế, nên một số loại tiêu chí dừng là cần thiết. Chiến lược này thường dựa trên một hoặc nhiều điều kiện liên quan đến những điều sau: số lần lặp lại, giá trị của thước đo đánh giá tập hợp con là tối ưu hay vượt quá một ngưỡng nhất định, liệu tập hợp con có kích thước nhất định đã được thu được chưa và liệu có cải tiến nào không có thể đạt được bằng các tùy chọn có sẵn cho chiến lược tìm kiếm.[35]

Cuối cùng, khi một tập hợp con các feature đã được chọn, kết quả của thuật toán khai thác dữ liệu mục tiêu trên tập hợp con đã chọn sẽ được xác nhận. Một cách tiếp cận xác thực đơn giản là chạy thuật toán với tập hợp đầy đủ các feature và so sánh kết quả đầy đủ với kết quả thu được bằng cách sử dụng tập hợp con các feature. Hy vọng rằng

tập hợp con các feature sẽ tạo ra kết quả tốt hơn hoặc gần như tốt hơn so với những kết quả được tạo ra khi sử dụng tất cả các feature. Một cách tiếp cận xác nhận khác là sử dụng một số thuật toán lựa chọn feature khác nhau để thu được các tập hợp con của các feature và sau đó so sánh kết quả của việc chạy thuật toán khai thác dữ liệu trên từng tập hợp con.[35]

Tạo feature

Từ các thuộc tính ban đầu, một tập hợp các thuộc tính mới để nắm bắt thông tin quan trọng trong tập dữ liệu một cách hiệu quả hơn nhiều.[35]

Khai thác-trích xuất feature:

Việc tạo một tập hợp các feature mới từ dữ liệu thô ban (raw data) đầu được gọi là trích xuất feature. Thật không may, theo nghĩa mà nó được sử dụng phổ biến nhất, việc trích xuất feature rất đặc trưng cho từng lĩnh vực. Bất cứ khi nào việc khai thác dữ liệu được áp dụng cho một lĩnh vực tương đối mới, nhiệm vụ quan trọng là phát triển các thuộc tính mới và các phương pháp khai thác feature.[35]

Chuẩn hóa (Normalization or Standardization):

Normalization: đề cập đến các kỹ thuật khác nhau để điều chỉnh sự khác biệt giữa các thuộc tính về tần suất xuất hiện, trung bình, phương sai, phạm vi. Loại bỏ tín hiệu phổ biến, không mong muốn, ví dụ: tính thời vụ.[35]

Trong thống kê, Standardization đề cập đến việc trừ đi các giá trị trung bình và chia cho độ lệch chuẩn std:

$$x' = \frac{x - \bar{x}}{std} \quad (2)$$

Tạo một biến mới có giá trị trung bình là 0 và độ lệch chuẩn là 1. Giá trị trung bình và độ lệch chuẩn bị ảnh hưởng mạnh bởi các giá trị ngoại lai, do đó, phép biến đổi trên thường được sửa đổi. Đầu tiên, giá trị trung bình được thay thế bằng giá trị trung vị, tức là giá trị năm ở giữa. Thứ hai, độ lệch chuẩn được thay thế bằng độ lệch chuẩn tuyệt đối.[35]

2.4. Các thước đo về sự giống nhau và không giống nhau

2.4.1. Định Nghĩa

Định nghĩa về giống nhau(similarity) giữa hai đối tượng là thước đo bằng số về mức độ giống nhau của hai đối tượng dữ liệu..[35]

Định nghĩa về không giống nhau(dissimilarity) giữa hai đối tượng là một số đo mức độ mà hai đối tượng khác nhau.[35]

Thông thường, thuật ngữ khoảng cách(distance) được sử dụng như một từ đồng nghĩa với dissimilarity, mặc dù, như chúng ta sẽ thấy, khoảng cách thường đề cập đến một lớp đặc biệt của dissimilarity. Dissimilarity đôi khi rơi vào khoảng [0, 1], nhưng chúng cũng thường nằm trong khoảng từ 0 đến ∞.[35]

Thuật ngữ proximity: được sử dụng để chỉ sự giống nhau hoặc không giống nhau.

Sự biến đổi thường được áp dụng để chuyển đổi một điểm tương tự thành một điểm khác nhau, hoặc ngược lại, hoặc để biến đổi một số đo độ gần thuộc phạm vi cụ thể, chẳng hạn như [0,1].[35]

2.4.2. Sự khác biệt giữa các đối tượng dữ liệu

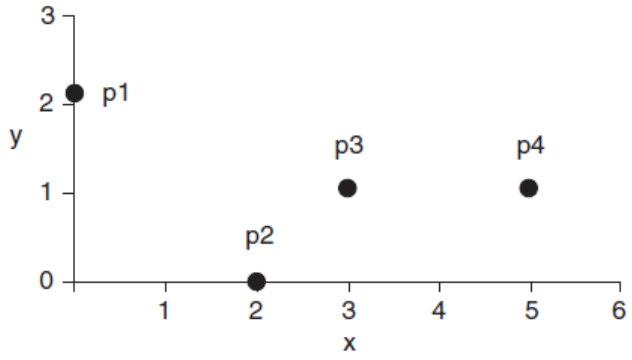
Khoảng cách(Distances):

Khoảng cách Euclide, d, giữa hai điểm, x và y, trong không gian một, hai, ba, hoặc chiều cao hơn, được cho bởi công thức sau:

$$d(x,y) = \sqrt{\sum_{k=1}^{n}(x_k - y_k)^2} \quad (3)$$

n là số chiều(số thuộc tính) với x_k và y_k tương ứng, các thuộc tính thứ k(các thành phần) hoặc các đối tượng dữ liệu x và y. Chuẩn hóa là cần thiết nếu các thang đo khác nhau.[35]

Hiển thị một tập hợp các điểm, tọa độ x và y của các điểm này và ma trận khoảng cách chứa các khoảng cách theo cặp của các điểm này.[35]



Hình 2- 5: Bốn điểm trên trục tọa độ hai chiều.

Bảng 2- 1: Tọa độ các điểm trên Euclide

Điểm	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

Bảng 2- 2: Ma trận khoảng cách

L2	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Trong Bảng 2-1 x, y là trục tọa độ không phải đối tương dữ liệu như các điểm p1 đến p1.

Số đo khoảng cách Euclide được đưa ra trong công thức 11 được tổng quát hóa bởi thước đo khoảng cách Minkowski:

$$d(x,y) = (\sum_{k=1}^{n} |x_k - y_k|^r)^{1/r} \quad (4)$$

r là tham số, n là số chiều(số thuộc tính) với x_k và y_k tương ứng, các thuộc tính thứ k(các thành phần) hoặc các đối tượng dữ liệu x và y.[35]

Sau đây là ba ví dụ phổ biến nhất về khoảng cách Minkowski:

- r = 1 Khoảng cách (Manhattan, L_1 norm). Một ví dụ phổ biến là khoảng cách Hamming, là số bit khác nhau giữa hai đối tượng chỉ có thuộc tính nhị phân, tức là giữa hai vectơ nhị phân.
- r=2. Khoảng cách Euclide (L_2 norm) .
- r = ∞. Supremum (L_{max} hoặc L∞ norm) khoảng cách. Đây là sự khác biệt lớn nhất giữa bất kỳ thành phần nào của các vectơ.[35]

Đừng nhầm lẫn r với n, tức là, tất cả các khoảng cách này là được xác định cho tất cả các số chiều.[35]

Bảng 2- 3: *Ma trận khoảng cách L1*

L1	p1	p2	p3	p4
p1	0	4	4	6
p2	4	0	2	4
p3	4	2	0	2
p4	6	4	2	0

Bảng 2- 4: Ma trận khoảng cách L∞

L∞	p1	p2	p3	p4

p1	0	2	3	5
p2	2	0	1	3
p3	3	1	0	2
p4	5	3	2	0

Khoảng cách Euclide, có một số đặc tính bật. Nếu d (x, y) là khoảng cách giữa hai điểm, x và y, thì có các tính chất sau đây:

1. d(x, y) ≥ 0 với mọi x và y
2. d(x, y) = 0 khi và chi khi x=y
3. Tính đối xứng d(x, y) = d(y, x) với mọi x và y
4. Bất đẳng thức tam giác d(x, z) ≤ d(x, y)+d(y, z) cho tất cả các điểm x, y và z

trong đó d (x, y) là khoảng cách (không giống nhau) giữa các điểm (đối tượng dữ liệu), x và y. Khoảng cách thỏa mãn các tính chất này là một thước đo.[35]

2.4.3. Về đo lường Proximity

Tương quan đo lường mối quan hệ tuyến tính giữa các đối tượng:

Tương quan(Correlation) thường được sử dụng để đo lường mối quan hệ tuyến tính giữa hai bộ giá trị được quan sát cùng nhau. Do đó, mối tương quan có thể đo lường mối quan hệ giữa hai biến. Tương quan được sử dụng thường xuyên hơn để đo mức độ giống nhau giữa các thuộc tính vì các giá trị trong hai đối tượng dữ liệu đến từ các thuộc tính khác nhau, có thể có các loại thuộc tính và tỷ lệ rất khác nhau. Có nhiều loại tương quan, và thực sự tương quan đôi khi được sử dụng theo nghĩa chung để chỉ mối quan hệ giữa hai bộ giá trị được quan sát cùng nhau.[35]

Pearson's correlation giữa hai bộ giá trị số, tức là hai vectơ, x và y, được xác định bởi phương trình sau[35]:

$$corr(x,y) = \frac{covariance(x,y)}{standard\_deviation(x) \times standard\_deviation(y)} = \frac{s_{xy}}{s_x \times s_y} \quad (5)$$

$$covariance(x,y) = s_{xy} = \frac{1}{n-1}\sum_{k=1}^{n}(x_k - \bar{x})(y_k - \bar{y}) \quad (6)$$

$$std(x) = s_x = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(x_k - \bar{x})^2} \quad (7)$$

$$std(x) = s_x = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(x_k - \bar{x})^2} \quad (8)$$

$$\bar{x} = \frac{1}{\mathrm{n}}\sum_{k=1}^{n} x_k \quad (9)$$

$$\bar{y} = \frac{1}{\mathrm{n}}\sum_{k=1}^{n} y_k \quad (10)$$

Ví dụ về mối tương quan hoàn hảo: Tương quan luôn nằm trong phạm vi -1 đến 1. Tương quan nghĩa là x và y có mối quan hệ tuyến tính dương (âm) hoàn hảo 1 (-1). Đó là, $x_k = a \times y_k + b$ trong đó a và b là hằng số. Hai vectơ x và y sau đây minh họa các trường hợp mà mối tương quan là -1 và +1, tương ứng: x = (−3, 6, 0, 3, −6) và y = (1, −2, 0, −1, 2)

mean(x) = mean(y) = 0, std(x) = 4.74, sdt(y) = 1.58,

$$corr(x, y) = \frac{(-3)\times(1)+(6)\times(-2)+(3)\times(-1)+(-6)\times(2)}{(4\times 4.74\times 1.58)} = -1.00144$$

Ví dụ mối quan hệ phi tuyến: Nếu mối tương quan bằng 0, thì không có mối quan hệ tuyến tính giữa hai bộ giá trị. Tuy nhiên, các mối quan hệ phi tuyến vẫn có thể tồn tại. Trong ví dụ sau $y_k = x_k^2$. x = (−3, −2, −1, 0, 1, 2, 3)y=(9, 4, 1, 0, 1, 4, 9)

mean(x) = 0, mean(y) = 4, std(x) = 2.16, std(y) = 3.74,

$$corr = \frac{(-3)(5)+(-2)(0)+(-1)(-3)+(0)(-4)+(1)(-3)+(2)(0)+3(5)}{(\,6 * 2.16 * 3.74\,)} = 0$$

Tương quan vs Cosine và Khoảng cách Euclide:

So sánh ba thước đo độ gần nhau theo hành vi của chúng trong điều kiện biến đổi biến đổi. Chia tỷ lệ: nhân với một giá trị, dịch: thêm một hằng số.[35]

Bảng 2- 5: Các thuộc tính của cosine, tương quan và các phép đo khoảng cách Euclide.[35]

Tính chất	Tương quan(Correlation)	Khoảng cách Euclide
Bất biến để mở rộng quy mô (phép nhân)	Có	Không
Bất biến dịch (bổ sung)	Có	Không

Ví dụ: x = (1, 2, 4, 3, 0, 0, 0), y = (1, 2, 3, 4, 0, 0, 0), y_s = y * 2 (phiên bản mở rộng của y), y_t = y + 5 (phép dịch)

Bảng 2- 6: Sự giống nhau giữa (x, y), (x, y_s), và (x, y_t).[35]

Đo lường	(x,y)	(x,y_s)	(x,y_t)
Tương quan(Correlation)	0.9429	0.9429	0.9429
Khoảng cách Euclide	1.4142	5.8310	14.2127

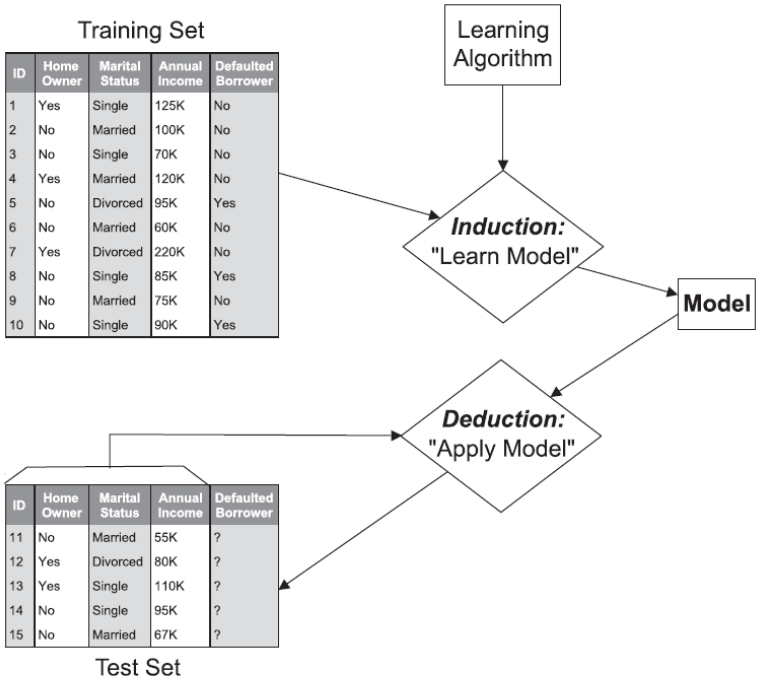
Lựa chọn thước đo độ gần phù hợp tùy thuộc vào miền.

2.5. Các Mô Hình

2.5.1. Khung phân loại chung

Quá trình phân loại bao gồm hai bước: áp dụng một thuật toán học để huấn luyện dữ liệu để học một mô hình và sau đó áp dụng mô hình để gán nhãn cho các cá thể không được gắn nhãn.[35]

Quá trình sử dụng thuật toán học tập để xây dựng mô hình phân loại từ dữ liệu huấn luyện được gọi là induction. Quá trình áp dụng mô hình phân loại trên các trường hợp thử nghiệm chưa thấy để dự đoán nhãn lớp của chúng được gọi là deduction.[35]



ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
11	No	Married	55K	?
12	Yes	Divorced	80K	?
13	Yes	Single	110K	?
14	No	Single	95K	?
15	No	Married	67K	?

Hình 2- 6: Khung chung để xây dựng mô hình phân loại.[35]

Dựa trên bộ dữ liệu chúng tôi sẽ sử dụng sẽ nói ở phần 3.2.1, chúng tôi sẽ sử dụng ba mô hình là cây quyết định phân loại và hồi quy và Logistic Regression, LightGBM. Với mô hình đầu tiên chúng tôi định sử dụng là mô hình cây quyết định phân loại và hồi quy có thuật toán học là CART (Classification and Regression Trees) mục đích là để xác định các lớp có mật độ giống nhau, đồng nhất về mức độ rủi ro, đồng thời để tối đa hóa sự khác biệt về mức độ rủi ro giữa các nhóm. Sử dụng thứ 2 là mô hình Logistic Regression vì output đầu ra là xác suất từ 0 đến 1 phù hợp với yêu cầu đầu ra của bộ dữ liệu của chúng tôi và nó nhanh về mặt tính toán và tạo ra một mô hình có thể được tính điểm cho dữ liệu mới chỉ với một vài phép toán số học. Còn mô hình LightGBM thuật toán rất mạnh khi nói đến tính toán và xử lí nhanh dùng để phân loại được nhiều lớp.

2.5.2. Bộ phân loại cây quyết định

Cây có ba loại nút:

- Một nút gốc, không có liên kết đến và không có hoặc nhiều liên kết đi.
- Các nút nội bộ, mỗi nút có chính xác một liên kết đến và hai hoặc nhiều liên kết đi.
- Các nút lá hoặc nút đầu cuối, mỗi nút có chính xác một liên kết đến và không có liên kết đi.[35]

Thuật toán cơ bản để xây dựng cây quyết định

Các thuật toán hiệu quả đã được phát triển để tạo ra một cây quyết định chính xác, mặc dù chưa tối ưu, trong một khoảng thời gian hợp lý. Các thuật toán này thường sử dụng một chiến lược tham lam để phát triển cây quyết định theo kiểu từ trên xuống bằng cách đưa ra một loạt các quyết định tối ưu cục bộ về thuộc tính nào sẽ sử dụng khi phân vùng dữ liệu huấn luyện.[35]

Thuật toán CART:

Cây phân loại và cây hồi quy (CART) là một kỹ thuật học trong cây quyết định phi tham số tạo ra cây phân loại hoặc cây hồi quy, tùy thuộc vào việc biến phụ thuộc tương ứng là phân loại hay số.[37]

Cây quyết định được hình thành bởi một tập hợp các quy tắc dựa trên các biến trong tập dữ liệu mô hình hóa:

- Các quy tắc dựa trên giá trị của các biến được chọn để có được sự phân tách tốt nhất nhằm phân biệt các quan sát dựa trên biến phụ thuộc.
- Khi một quy tắc được chọn và chia một nút thành hai, quy trình tương tự sẽ được áp dụng cho mỗi nút "con" (tức là nó là một thủ tục đệ quy)
- Việc tách dừng khi CART phát hiện không thể thực hiện thêm được nữa hoặc đáp ứng một số quy tắc dừng đặt trước. (Ngoài ra, dữ liệu được phân chia càng nhiều càng tốt và sau đó cây sẽ được cắt tỉa)

Mỗi nhánh của cây kết thúc bằng một nút lá. Mỗi quan sát rơi vào một và chính xác một nút lá, và mỗi nút lá được xác định duy nhất bởi một bộ quy tắc.[37]

Các vấn đề thiết kế của Decision Tree Induction

Các bản ghi ở tập dữ liệu huấn luyện nên được chia như thế nào?

- Phương pháp thể hiện điều kiện kiểm tra tùy thuộc vào loại thuộc tính
- Đo lường để đánh giá mức độ tốt của một điều kiện thử nghiệm

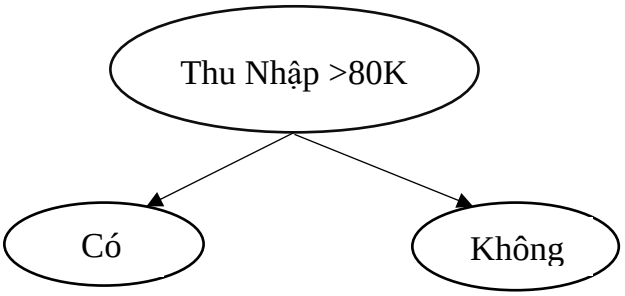
Làm thế nào nên dừng thủ tục tách?

- Dừng tách nếu tất cả các bản ghi thuộc cùng một lớp hoặc có các giá trị thuộc tính giống hệt nhau
- Có những lý do để ngừng mở rộng một nút sớm hơn nhiều ngay cả khi nút lá chứa các cá thể huấn luyện từ nhiều hơn một lớp. Quá trình này được gọi là kết thúc sớm và điều kiện được sử dụng để xác định thời điểm nên dừng mở rộng một nút được gọi là tiêu chí dừng.[35]

Các phương pháp để thể hiện các điều kiện kiểm tra thuộc tính

Thuộc tính nhị phân

Điều kiện kiểm tra cho một thuộc tính nhị phân tạo ra hai kết quả tiềm năng.[35]

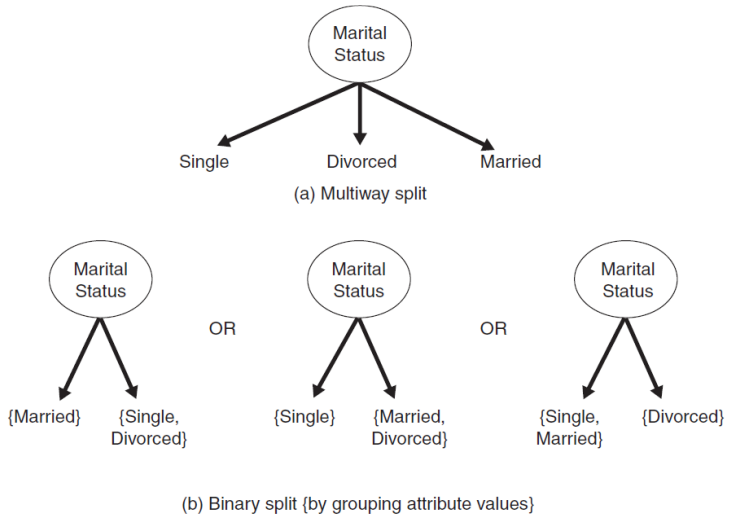


Hình 2- 7: Điều kiện kiểm tra thuộc tính cho một thuộc tính nhị phân.

Thuộc tính danh nghĩa

Một thuộc tính danh nghĩa có thể có nhiều giá trị, điều kiện kiểm tra thuộc tính của nó có thể được thể hiện theo hai cách, dưới dạng phân tách nhiều nhánh hoặc phân tách nhị phân.[35]

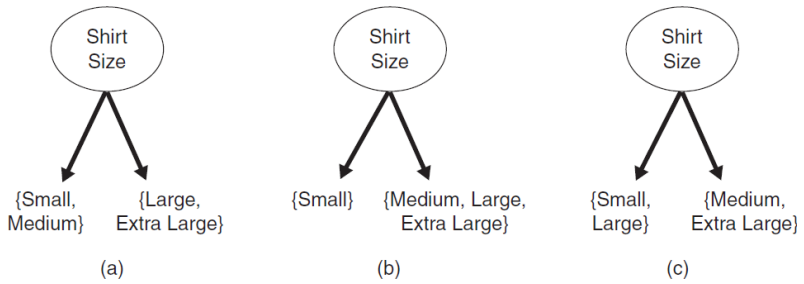
- Phân tách nhiều nhánh (a): sử dụng nhiều phân vùng như các giá trị riêng biệt.
- Phân tách nhị phân (b): Chia các giá trị thành hai tập con.



Hình 2- 8: Điều kiện kiểm tra thuộc tính cho các thuộc tính danh nghĩa.[35]

Thuộc tính theo thứ tự

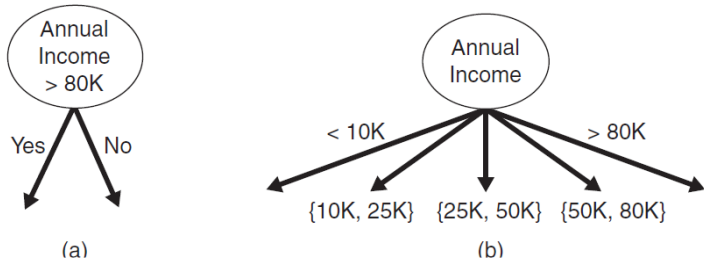
Các thuộc tính thứ tự cũng có thể tạo ra các phân tách nhị phân hoặc nhiều nhánh. Các giá trị thuộc tính thứ tự có thể được nhóm lại miễn là việc nhóm không vi phạm thuộc tính thứ tự của các giá trị thuộc tính.[35]



Hình 2- 9: Các cách khác nhau để nhóm các giá trị thuộc tính thứ tự.[35]

Thuộc tính liên tục

Đối với các thuộc tính liên tục, điều kiện kiểm tra thuộc tính có thể được biểu thị dưới dạng kiểm tra so sánh (ví dụ: A < v) tạo ra phép tách nhị phân hoặc dưới dạng truy vấn phạm vi có dạng $v_i \leq A < v_i + 1$, với i = 1,..., k ,tạo ra phép tách nhiều chiều.[35]



Hình 2- 10: Điều kiện kiểm tra các thuộc tính liên tục.[35]

Discretization để tạo thuộc tính phân loại theo thứ tự(b):

Các phạm vi có thể được tìm thấy bằng cách bỏ phiếu theo khoảng thời gian bằng nhau, tăng tần suất bằng nhau (phần trăm) hoặc phân nhóm.

- Tĩnh(Static) – phân nhóm 1 lần duy nhất ngay lúc trước khi xây dựng mô hình trước khi áp dụng các công thức để chọn ngọn(root rode)
- Động(Dynamic) - tại mỗi lần phân nhánh, chúng ta phân nhóm (lặp lại ở mỗi nút)

Quyết định nhị phân(a): (A < v) hoặc (A ≥ v) xem xét tất cả các phân tách có thể có và tìm ra cách cắt tốt nhất, có thể tính toán chuyên sâu hơn.[35]

Các biện pháp để chọn một điều kiện kiểm tra thuộc tính

Các biện pháp này cố gắng ưu tiên các điều kiện kiểm tra thuộc tính để phân chia các cá thể huấn luyện thành các tập con thuần hơn trong các nút con, hầu hết có cùng nhãn lớp. Việc có các nút tinh khiết hơn rất hữu ích vì một nút có tất cả các phiên bản huấn luyện của nó từ cùng một lớp không cần phải mở rộng thêm. Ngược lại, một nút

không tinh khiết chứa các cá thể huấn luyện từ nhiều lớp có khả năng yêu cầu một số cấp độ mở rộng của nút, do đó làm tăng đáng kể độ sâu của cây. Các cây lớn hơn ít được mong muốn hơn vì chúng dễ bị trang bị quá mức mô hình, một điều kiện có thể làm giảm hiệu suất phân loại trên các trường hợp không nhìn thấy.[35]

Đo nhiễu cho một nút duy nhất

Nhiễu của một nút đo lường mức độ khác nhau của các nhãn lớp đối với các cá thể dữ liệu thuộc về một nút chung. Các biện pháp có thể được sử dụng để đánh giá nhiễu của một nút t:

$$Gini\ Index = 1 - \sum_{i=0}^{c-1} p_i(t)^2 \tag{11}$$

- Tối đa là 1 - 1 / c khi các bản ghi được phân bổ đều cho tất cả các lớp, ngụ ý tình huống ít có lợi nhất cho việc phân loại. Tối thiểu là 0 khi tất cả các bản ghi thuộc một lớp, ngụ ý tình huống có lợi nhất cho việc phân loại. Gini Index được sử dụng trong các thuật toán cây quyết định như CART, SLIQ, SPRINT.

1 biến (sự kiện) là X. Với n giá trị có khả năng(đầu ra) là x1,x2,...,xn. Mỗi đầu ra có 1 xác suất là p1, p2,...,pn. Entropy của X là H(X), theo công thức sau[33]:

$$Entropy = -\sum_{i=0}^{c-1} p_i(t) log_2 p_i(t) \tag{12}$$

- Entropy nằm giữa 0 và log2n và được đo bằng các bit. Do đó, entropy là thước đo trung bình cần bao nhiêu bit để đại diện cho một quan sát về X.[33]
- Tối đa là $\log_2 c$ khi các bản ghi được phân bổ đồng đều giữa tất cả các lớp, ngụ ý tình huống ít có lợi nhất cho việc phân loại. Tối thiểu là 0 khi tất cả các bản ghi thuộc về một lớp, ngụ ý tình huống có lợi nhất cho việc phân loại. Tính toán dựa trên Entropy khá giống với tính toán GINI Index.

Entroy cho mẫu dữ liệu: Giả sử số các quan sát (m) của một vài thuộc tính là X, nơi có n giá trị có thể khác nhau, và số lượng quan sát trong danh mục i là m_i. Và entropy cho mẫu dữ liệu được tính theo công thức[35]:

$$H(X) = -\sum_{i=1}^{n} \frac{m_i}{m} \log_2 \frac{m_i}{m} \tag{13}$$

Lỗi phân loại sai:

$$Classification\ error = 1 - \max\left[p_i(t)\right] \quad (14)$$

- Tối đa 1 - 1 / c khi các bản ghi được phân bổ đều cho tất cả các lớp, ngụ ý tình huống ít thú vị nhất. Tối thiểu là 0 khi tất cả các bản ghi thuộc một lớp, ngụ ý tình huống thú vị nhất.

Trong đó $\boldsymbol{p_i(t)}$ là tần suất của lớp $\boldsymbol{i}$ tại nút t, và $\boldsymbol{c}$ là tổng số lớp. Cả ba phép đo đều cho giá trị nhiễu bằng không nếu một nút chứa các cá thể từ một lớp đơn lẻ và nhiễu tối đa nếu nút có tỷ lệ các cá thể từ nhiều lớp bằng nhau.[35]

Tính toán Gini index cho một tập hợp các nút

Khi một nút p được chia thành k phân vùng (con):

$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} \times GINI(i) \quad (15)$$

n_i = số bản ghi ở nút con i, n = số bản ghi ở nút cha p.[35]

Nhiễu tập hợp của các nút con[35]

Hãy xem xét một điều kiện kiểm tra thuộc tính chia một nút chứa N mẫu huấn luyện thành k con, $\{v_1, v_2,\ldots,v_k\}$, trong đó mọi nút con đại diện cho một phân vùng dữ liệu được tạo ra từ một trong k kết quả của điều kiện kiểm tra thuộc tính. Để $N(v_j)$ số lượng mẫu huấn luyện được liên kết với một nút con v_j , giá trị nhiễu của nó là $I(v_j)$. Kể từ khi một trường hợp huấn luyện trong nút cha đạt nút v_j cho một tỷ lệ của $N(v_j)/N$ lần, nhiễu chung của các nút con có thể được tính bằng cách lấy tổng trọng số của các nhiễu của các nút con, như sau:

$$I(children) = \sum_{j=1}^{k} \frac{N(v_j)}{N} \times I(v_j) \quad (16)$$

Tìm sự phân chia tốt nhất[35]

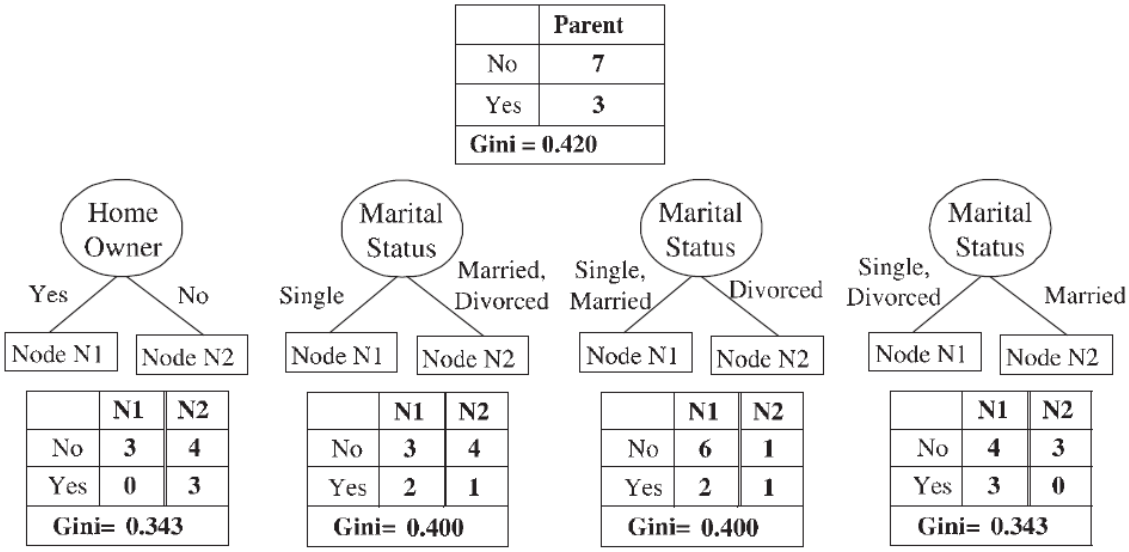
1) Tính toán số đo nhiễu (P) trước khi tách.
2) Tính toán số đo nhiễu (M) sau khi tách. Tính toán số đo nhiễu của mỗi nút con. M là nhiễu có trọng số của các nút con.
3) Chọn điều kiện kiểm tra thuộc tính tạo ra mức tăng (gain) cao nhất

$$Gain = P - M \quad (17)$$

Hay là

$$\Delta = I(parent) - I(children) \quad (18)$$

hoặc tương đương, đo nhiễu thấp nhất sau khi tách (M).



Hình 2- 11: Tách các tiêu chí cho bài toán phân loại người đi vay bằng cách sử dụng chỉ số Gini.

Cuối cùng, khi entropy được sử dụng làm thước đo nhiễu, sự khác biệt về entropy thường được gọi là độ lợi thông tin(information gain), Δinfo.

Tính toán thông tin thu được sau khi tách (Information gain):

$$Gain_{split} = Entropy(p) - \sum_{i=1}^{k} \frac{n_i}{n} \times Entropy(i) \quad (19)$$

Nút cha p được chia thành k phân vùng (con) n_i là số bản ghi trong nút con i. Chọn phần tách giảm được nhiều nhất (tối đa hóa GAIN). Được sử dụng trong thuật toán cây quyết định ID3 và C4.5. Độ lợi thông tin(information gain) là thông tin lẫn nhau(mutual information) giữa biến lớp và biến tách.

Tách nhị phân của các thuộc tính định tính[35]

Tách thành hai phân vùng (các nút con). Tác dụng của phân vùng có trọng lượng: Các phân vùng lớn hơn và tinh khiết hơn được tìm kiếm.

Dựa vào dữ liệu Hình 2-11:

Gini index của nút cha trước khi tách = $1 - (3/10)^2 – (7/10)^2 = 0.42$

Nếu Home Owner được chọn là thuộc tính phân tách.

Gini index(N1) = $1 - (3/3)^2 – (0/3)^2 = 0$. Gini index(N2) $=1 – (4/7)^2 – (3/7)^2 = 0.49$

Chỉ số Gini trung bình có trọng số của nút con = $(3/10)*0+(7/10)*0.49 = 0.343$.

Gain = 0.42 – 0.343 = 0.077

Tách nhị phân của các thuộc tính định lượng[35]

Hãy xem xét vấn đề xác định Thu nhập hàng năm $\leq$ t phân tách nhị phân tốt nhất cho bài toán phân loại phê duyệt khoản vay trước đó. Như đã thảo luận trước đây, mặc dù có thể lấy bất kỳ giá trị nào giữa giá trị tối thiểu và tối đa của thu nhập hàng năm trong tập huấn luyện, nhưng chỉ cần xem xét các giá trị thu nhập hàng năm được quan sát trong tập huấn luyện là các ứng cử viên cho vị trí phân chia là đủ. Đối với mỗi ứng cử viên t, tập huấn luyện được quét một lần để đếm số người vay có thu nhập hàng năm nhỏ hơn hoặc lớn hơn t cùng với tỷ lệ lớp của họ. Sau đó, chúng tôi có thể tính toán chỉ số Gini tại mỗi vị trí phân chia và chọn t tạo ra giá trị thấp nhất. Việc tính toán chỉ số Gini tại mỗi vị trí phân chia yêu cầu các phép toán O(N), trong đó N là số lượng các trường hợp huấn luyện. Kể từ khi có nhiều nhất N có thể ứng cử viên cho vị trí phân chia, mức độ phức tạp tổng thể của phương pháp brute-force này là $O(N^2)$. Có thể giảm mức độ phức tạp của vấn đề này xuống O(Nlog N) bằng cách sử dụng một phương pháp được mô tả như sau:

Sử dụng Quyết định nhị phân dựa trên một giá trị, một số lựa chọn cho giá trị phân tách: Số lượng các giá trị có thể tách = Số lượng các giá trị riêng biệt.

Mỗi giá trị tách có một ma trận đếm được liên kết với nó: đếm lớp trong mỗi phân vùng, A ≤ v và A> v.

Phương pháp đơn giản để chọn tốt nhất v: Đối với mỗi v, hãy quét cơ sở dữ liệu để thu thập ma trận đếm và tính toán chỉ số Gini của nó. Tính toán không hiệu quả. Lập lại các bước trên. Để tính toán hiệu quả cho mỗi thuộc tính:

- Sắp xếp thuộc tính trên các giá trị
- Quét tuyến tính các giá trị này, mỗi lần cập nhật ma trận đếm và chỉ số gini tính toán
- Chọn vị trí tách có chỉ số gini ít nhất

Đặc điểm của bộ phân loại cây quyết định

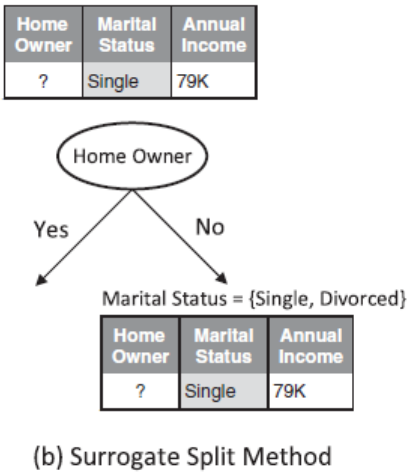
Khả năng áp dụng: Cây quyết định là một cách tiếp cận phi tham số để xây dựng các mô hình phân loại. Cách tiếp cận này không yêu cầu bất kỳ giả định trước nào về phân phối xác suất chi phối lớp và thuộc tính của dữ liệu, do đó, có thể áp dụng cho nhiều loại tập dữ liệu. Nó cũng có thể áp dụng cho cả dữ liệu phân loại và dữ liệu liên tục mà không yêu cầu các thuộc tính phải được chuyển đổi thành một đại diện chung

thông qua mã hóa nhị phân, chuẩn hóa hoặc chuẩn hóa, nó cũng có thể đối phó với vấn đề nhiều lớp mà không cần phải phân tách chúng thành nhiều nhiệm vụ phân loại nhị phân. Một thuộc tính hấp dẫn khác của các bộ phân loại cây quyết định là các cây quy nạp, đặc biệt là các cây ngắn hơn, tương đối dễ giải thích. Độ chính xác của cây cũng tương đối so với các kỹ thuật phân loại khác cho nhiều tập dữ liệu đơn giản.[35]

Khả năng truyền đạt: Cây quyết định cung cấp một biểu diễn phổ quát cho các hàm có giá trị rời rạc. Nói cách khác, nó có thể mã hóa bất kỳ chức năng nào của các thuộc tính có giá trị rời rạc. Điều này là do mọi hàm có giá trị rời rạc có thể được biểu diễn dưới dạng bảng gán, trong đó mọi kết hợp duy nhất của các thuộc tính rời rạc được gán một nhãn lớp. Vì mọi tổ hợp thuộc tính có thể được biểu diễn dưới dạng một lá trong cây quyết định, chúng ta luôn có thể tìm thấy một cây quyết định có các phép gán nhãn tại các nút lá khớp với bảng gán của hàm gốc. Cây quyết định cũng có thể giúp cung cấp các biểu diễn ngắn gọn của các hàm khi một số tổ hợp thuộc tính duy nhất có thể được biểu diễn bằng cùng một nút lá.

Hiệu quả tính toán: Vì số lượng cây quyết định có thể có có thể rất lớn, nhiều thuật toán cây quyết định sử dụng cách tiếp cận dựa trên kinh nghiệm để hướng dẫn tìm kiếm của chúng trong không gian giả thuyết rộng lớn. Đối với nhiều tập dữ liệu, các kỹ thuật như vậy nhanh chóng xây dựng một cây quyết định hợp lý tốt ngay cả khi kích thước tập huấn luyện rất lớn. Hơn nữa, một khi cây quyết định đã được xây dựng, việc phân loại bản ghi kiểm tra là cực kỳ nhanh chóng, với độ phức tạp trong trường hợp xấu nhất là O(w), trong đó w là độ sâu tối đa của cây.[35]

Xử lý các giá trị bị thiếu: Bộ phân loại cây quyết định có thể xử lý các giá trị thuộc tính bị thiếu theo một số cách, cả trong tập huấn luyện và tập kiểm tra. Khi thiếu các giá trị trong tập kiểm tra, bộ phân loại phải quyết định nhánh nào sẽ tuân theo nếu thiếu giá trị của thuộc tính nút tách đối với một phiên bản kiểm tra nhất định. Thuật toán CART sử dụng phương pháp phân tách thay thế (surrogate split method), trong đó cá thể có giá trị thuộc tính phân tách bị thiếu được chỉ định cho một trong các nút con dựa trên giá trị của một thuộc tính thay thế không bị thiếu khác mà phần phân tách giống nhất với các phân vùng được tạo bởi phần bị thiếu thuộc tính. Hình 2-14 cho thấy một ví dụ về cách để xử lý các giá trị bị thiếu trong bộ phân loại cây quyết định. Các chiến lược khác để xử lý các giá trị bị thiếu dựa trên xử lý trước dữ liệu, trong đó cá thể có giá trị bị thiếu được áp dụng với giá trị mode (đối với thuộc tính phân loại) hoặc giá trị trung bình (đối với thuộc tính liên tục) hoặc bị loại bỏ trước khi bộ phân loại được huấn luyện.[35]



Hình 2- 12: Các phương pháp để xử lý các giá trị thuộc tính bị thiếu trong phân loại cây quyết định.[35]

Xử lý tương tác giữa các thuộc tính: Các thuộc tính được coi là tương tác nếu chúng có thể phân biệt giữa các lớp khi được sử dụng cùng nhau, nhưng riêng lẻ chúng cung cấp ít hoặc không có thông tin. Do tính chất tham lam của các tiêu chí phân tách trong cây quyết định, các thuộc tính như vậy có thể được chuyển qua để có lợi cho các thuộc tính khác không hữu ích bằng. Điều này có thể dẫn đến nhiều cây quyết định phức tạp hơn mức cần thiết. Do đó, cây quyết định có thể hoạt động kém khi có sự tương tác giữa các thuộc tính.

Xử lý các thuộc tính không liên quan: Một thuộc tính không liên quan nếu nó không hữu ích cho nhiệm vụ phân loại. Vì các thuộc tính không liên quan được liên kết kém với các nhãn lớp mục tiêu, chúng sẽ cung cấp ít hoặc không tăng độ tinh khiết và do đó sẽ bị các thuộc tính khác phù hợp hơn chuyển qua. Do đó, sự hiện diện của một số lượng nhỏ các thuộc tính không liên quan sẽ không ảnh hưởng đến quá trình xây dựng cây quyết định. Tuy nhiên, không phải tất cả các thuộc tính cung cấp ít hoặc không tăng đều không liên quan và có một số lượng lớn các thuộc tính không liên quan, khi đó một số thuộc tính này có thể vô tình được chọn trong quá trình phát triển cây, vì chúng có thể cung cấp mức tăng tốt hơn so với thuộc tính có liên quan chỉ do ngẫu nhiên. Các kỹ thuật lựa chọn feature có thể giúp cải thiện độ chính xác của cây quyết định bằng cách loại bỏ các thuộc tính không liên quan trong quá trình tiền xử lý.[35]

Xử lý các thuộc tính thừa thãi: Một thuộc tính là dư thừa nếu nó có tương quan chặt chẽ với một thuộc tính khác trong dữ liệu. Vì các thuộc tính dư thừa cho thấy mức độ tinh khiết đạt được tương tự nếu chúng được chọn để tách, chỉ một trong số chúng sẽ được chọn làm điều kiện kiểm tra thuộc tính trong thuật toán cây quyết định. Do đó, cây quyết định có thể xử lý sự hiện diện của các thuộc tính dư thừa.[35]

Sử dụng Rectilinear Splits: Các điều kiện thử nghiệm được mô tả cho đến nay trong liên quan đến việc chỉ sử dụng một thuộc tính duy nhất tại một thời điểm. Do đó, thủ tục phát triển cây có thể được xem như là quá trình phân vùng không gian thuộc tính thành các vùng rời rạc cho đến khi mỗi vùng chứa các bản ghi của cùng một lớp. Biên giới giữa hai vùng lân cận thuộc các lớp khác nhau được gọi là ranh giới quyết định.

Lựa chọn biện pháp đo độ nhiễu: Cần lưu ý rằng việc lựa chọn thước đo độ nhiễu thường ít ảnh hưởng đến hiệu suất của bộ phân loại cây quyết định vì nhiều thước đo tạp chất khá nhất quán với nhau. Thay vào đó, chiến lược được sử dụng để cắt tỉa cây có tác động lớn hơn đến cây cuối cùng so với việc lựa chọn biện pháp đo độ nhiễu.[35]

Ưu điểm và nhược điểm

Ưu điểm:

- Tương đối rẻ để xây dựng
- Cực kỳ nhanh chóng trong việc phân loại các bản ghi không xác định
- Dễ hiểu đối với những cây có kích thước nhỏ
- Mạnh mẽ với tiếng ồn (đặc biệt là khi sử dụng các phương pháp tránh trang bị quá mức)
- Có thể dễ dàng xử lý các thuộc tính thừa
- Có thể dễ dàng xử lý các thuộc tính không liên quan (trừ khi các thuộc tính đang tương tác)

Nhược điểm:

- Do tính chất tham lam của tiêu chí phân tách, các thuộc tính tương tác (có thể phân biệt giữa các lớp với nhau nhưng không riêng lẻ) có thể được chuyển sang cho các thuộc tính khác ít phân biệt hơn.
- Mỗi ranh giới quyết định chỉ liên quan đến một thuộc tính duy nhất

2.5.3. Logistic Regression

Hồi quy logistic tương tự như hồi quy đa tuyến tính (multiple linear regression) chấp nhận kết quả là nhị phân. Biến đổi đa dạng được sử dụng để chuyển đổi các vấn đề mà có thể phù hợp trong một mô hình tuyến tính. Các mô hình phân loại chỉ định trực tiếp các nhãn lớp mà không tính toán xác suất có điều kiện của lớp được gọi là mô hình phân biệt. Giống như phân tích phân biệt, hồi quy logistic là một cách tiếp cận mô hình có cấu trúc chứ không phải là một cách tiếp cận tập trung vào dữ liệu. Do

tốc độ tính toán nhanh và đầu ra của một mô hình cho phép ghi nhanh dữ liệu mới, nên nó là một phương pháp phổ biến. Các thành phần quan trọng cho hồi quy logistic là logistic response function và logit, trong đó chúng ta ánh xạ một xác suất (ở thang điểm 0–1) sang một tỷ lệ mở rộng hơn phù hợp với mô hình tuyến tính. Bước đầu tiên là nghĩ về biến kết quả không phải là một nhãn nhị phân mà là xác suất p mà nhãn đó là "1". [38]

Lập mô hình p bằng cách áp dụng phản hồi logistic hoặc hàm logit nghịch đảo cho các yếu tố dự đoán:

$$p = \frac{1}{1 + e^{-(\beta_0+\beta_1 x_1+\beta_2 x_2+\ldots\ldots+\beta_p x_p)}} \tag{20}$$

Phép biến đổi này đảm bảo rằng p nằm trong khoảng từ 0 đến 1. Để lấy biểu thức hàm mũ ra khỏi mẫu số, chúng ta xem xét tỷ lệ odds thay vì xác suất. Về xác suất, odds là là xác suất của một sự kiện chia cho xác suất sự kiện đó sẽ không xảy ra.[38] Hay nói cách khác là tỷ lệ giữa, "vỡ nợ"(1) và "không vỡ nợ"(0). Ví dụ: nếu xác suất vỡ nợ là 0,5, thì xác suất không vỡ nợ là (1-0,5) = 0,5, thì Odds sẽ bằng 1, tương tự với công thức dưới đây:

$$Odds(Y = 1) = \frac{p}{1 - p} \tag{21}$$

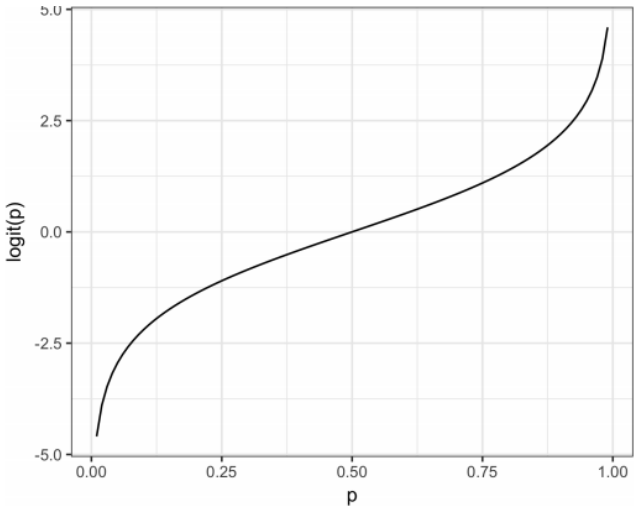
Chúng ta có thể có được xác suất từ biểu thức odds bằng cách sử dụng hàm odds nghịch đảo:

$$p = \frac{Odds}{1 + Odds} \tag{22}$$

Kết hợp công thức (21) và (20) rồi lấy logarit của cả hai bên chúng ta nhận được một biểu thức liên quan đến một hàm tuyến tính của các yếu tố dự đoán:

$$\log\left(Odds(Y = 1)\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots\ldots + \beta_p x_p \tag{23}$$

Hàm log-odds, còn được gọi là hàm logit. Ánh xạ xác suất p từ (0, 1) đến bất kỳ giá trị nào (- ∞, + ∞). Chúng tôi đã sử dụng mô hình tuyến tính để dự đoán một xác suất, tiếp theo chúng tôi có thể ánh xạ tới nhãn lớp bằng cách áp dụng quy tắc ngưỡng(threshold hoặc là cut-off), bất kỳ bản ghi nào có xác suất lớn hơn ngưỡng giới hạn đều được phân loại là 1.[38]



Hình 2- 13: Đồ thị của hàm logit ánh xạ xác suất đến một tỷ lệ phù hợp với mô hình tuyến tính

Phản hồi trong công thức hồi quy logistic là log odds của một kết quả nhị phân của 1. Chúng tôi chỉ quan sát kết quả nhị phân, không quan sát log odds, vì vậy cần có các phương pháp thống kê đặc biệt để phù hợp với phương trình. Hồi quy logistic là một ví dụ đặc biệt của mô hình tuyến tính tổng quát (generalized linear model-GLM) được phát triển để mở rộng hồi quy tuyến tính cho các cài đặt khác. Trong Python, sử dụng thư viện scikit-learn, LogisticRegression từ sklearn.linear_model. Các đối số penalty và C được sử dụng để ngăn chặn việc overfitting bởi L1 hoặc L2.[38]

Generalized Linear Models[38]

Mô hình tuyến tính tổng quát(GLMs) được đặc trưng bởi hai thành phần chính:

- Một phân bố xác suất hoặc gia đình (nhị thức trong trường hợp hồi quy logistic).
- Một hàm liên kết — tức là một hàm chuyển đổi ánh xạ phản ứng với các yếu tố dự đoán (logit trong trường hợp hồi quy logistic).

Giá trị dự đoán từ hồi quy logistic

Giá trị dự đoán từ hồi quy logistic là log odds: $\hat{Y} = \log(Odds(Y = 1))$. Xác suất dự đoán được đưa ra bởi hàm phản hồi logistic:

$$\hat{p} = \frac{1}{1 + e^{-\hat{Y}}} \tag{24}$$

Các xác suất có sẵn trực tiếp bằng cách sử dụng các phương pháp predict_proba trong scikit-learn. Các giá trị này nằm trên thang điểm từ 0 đến 1 và chưa tuyên bố

liệu giá trị dự đoán có phải là giá trị mặc định hay kết quả đúng hay không. Chúng tôi có thể khai báo bất kỳ giá trị nào lớn hơn 0,05 theo mặc định. Trong thực tế, mức giới hạn thấp hơn thường thích hợp nếu mục tiêu là xác định các thành viên của một lớp hiếm.[38]

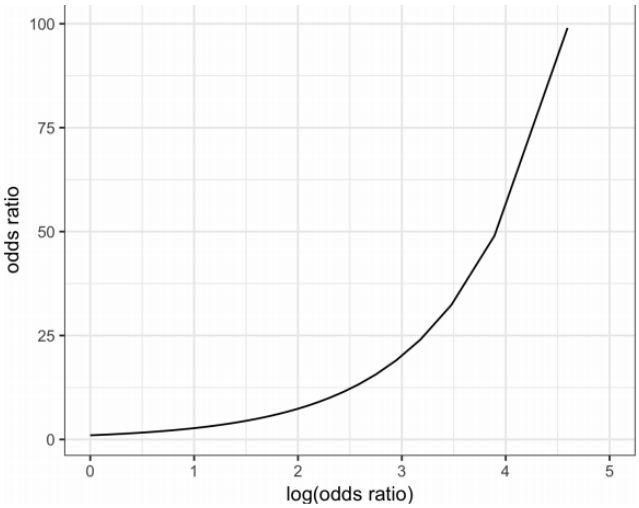
Giải thích các Tỷ lệ Odds (Odds Ratios)

Một ưu điểm của hồi quy logistic là nó tạo ra một mô hình có thể được ghi vào dữ liệu mới một cách nhanh chóng mà không cần tính toán lại. Một điều khác là mô hình tương đối dễ giải thích so với các phương pháp phân loại khác. Ý tưởng khái niệm chính là hiểu một tỷ lệ odds, Odds ratio dễ hiểu nhất đối với biến nhân tố nhị phân X:

$$odds\ ratio = \frac{Odds(Y = 1|X = 1)}{Odds(Y = 1|X = 0)} \quad (25)$$

Điều này được hiểu là odds Y=1 khi X=1 so với odds Y=1 khi X=0. Nếu tỷ lệ odds là 2, thì odds có Y =1 khi X = 1 cao hơn hai lần khi so với khi X = 0. Chúng ta làm việc với odds bởi vì hệ số βj trong hồi quy logistic là số mũ (log) của odds ratio cho Xj.

Hình 2-16 cho thấy mối quan hệ giữa odds ratio và log-odds ratio cho odds ratios > 1. Bởi vì các hệ số trên thang log, tăng 1 trong hệ số dẫn đến tăng exp 1 ≈ 2.72 trong odds ratio.



Hình 2- 14: Mối quan hệ giữa odds ratio và log-odds ratio

Odds ratios cho các biến số X có thể được hiểu tương tự: chúng đo lường sự thay đổi của odds ratio đối với một sự thay đổi đơn vị trong X.[38]

Fitting the model

Trong hồi quy logistic, không có giải pháp dạng đóng và mô hình phải fit bằng cách sử dụng ước tính khả năng xảy ra tối đa (maximum likelihood estimation-MLE).

MLE là một quá trình cố gắng tìm ra mô hình có nhiều khả năng đã tạo ra dữ liệu mà chúng ta thấy. Trong phương trình hồi quy logistic, phản hồi không phải là 0 hoặc 1 mà là ước tính của log odds cho rằng phản hồi là 1. MLE tìm ra giải pháp sao cho ước tính log odds mô tả tốt nhất kết quả quan sát được. Cơ chế của thuật toán liên quan đến tối ưu hóa quasi-Newton lặp lại giữa một bước tính điểm (Fisher's scoring) , dựa trên các tham số hiện tại và cập nhật các tham số để cải thiện sự phù hợp.[38]

Maximum Likelihood Estimation

Đây là một chút chi tiết hơn, nếu bạn thích các ký hiệu thống kê: hãy bắt đầu với một tập hợp dữ liệu (X_1 , X_2,…, X_n) và một mô hình xác suất P_θ (X_1 , X_2 , ⋯, X_n) điều đó phụ thuộc vào một tập hợp các tham số θ. Mục tiêu của MLE là tìm tập hợp các tham số $\hat{\theta}$ điều đó tối đa hóa giá trị của P_θ (X_1 , X_2 , ⋯, X_n) nghĩa là, nó tối đa hóa xác suất quan sát (X_1 , X_2,…, X_n) đưa ra mô hình P.[38]

Trong quá trình điều chỉnh, mô hình được đánh giá bằng cách sử dụng một số liệu gọi là độ lệch:

$$deviance = -2\log\left(P\theta\ (X1\ , X2\ , \cdots , Xn)\right) \qquad (26)$$

Độ lệch thấp hơn tương ứng với sự phù hợp tốt hơn. May mắn thay, hầu hết các chúng tôi không cần phải quan tâm đến các chi tiết của thuật toán điều chỉnh vì điều này được xử lý bởi phần mềm. Hầu hết các nhà khoa học dữ liệu sẽ không cần phải lo lắng về phương pháp phù hợp, ngoài việc hiểu rằng đó là một cách để tìm ra một mô hình tốt theo một số giả định nhất định.

Logistic Regression với các tham số như sau:

- Penalty: {'l1', 'l2', 'elasticnet', 'none'}, default='l2', được sử dụng để chỉ định chuẩn (norm) để sử dụng trong penalty ngăn cho dữ liệu bị overfiting.
- C: float, default=1.0, nghịch đảo của cường độ chính quy hóa; phải là một số thực dương, các giá trị nhỏ hơn xác định hiểu chỉnh mạnh hơn.

Điều chỉnh siêu tham số cho LR chúng tôi dùng với GridSearchCV các thông số như sau[39]***:***

Các tham số của công cụ ước tính được sử dụng để áp dụng các phương pháp này được tối ưu hóa bằng cách tìm kiếm lưới cross-validation trên lưới tham số.

- Estimator (đối tượng dự đoán): Điều này được giả định để triển khai giao diện ước tính scikit-learning. Công cụ ước tính cần cung cấp một hàm điểm hoặc phải cho điểm.
- param_grid (dict-từ điển hoặc danh sách các từ điển): Từ điển có tên tham số (str) làm khóa và danh sách cài đặt tham số để thử dưới dạng giá trị hoặc danh sách các từ điển như vậy, trong trường hợp đó, các lưới được mở rộng bởi từng từ điển trong danh sách sẽ được khám phá. Điều này cho phép tìm kiếm trên bất kỳ chuỗi cài đặt thông số nào.
- Scoring (str, callable, list, tuple or dict, default=None): Chiến lược đánh giá hiệu suất của mô hình được xác nhận chéo trên tập thử nghiệm.
- n_jobs (int, default=None): Số lượng công việc phải chạy song song. -1 có nghĩa là sử dụng tất cả các bộ xử lý.
- Cv (int, trình tạo cross-validation hoặc có thể lặp lại, default=None): Xác định chiến lược phân tách cross-validation.

Thuộc tính của GridSearchCV[39]***:***

- best_estimator_ (estimator): Công cụ ước tính được chọn bởi tìm kiếm, tức là công cụ ước tính cho điểm cao nhất (hoặc tổn thất nhỏ nhất nếu được chỉ định) trên dữ liệu bị bỏ sót. Không có sẵn nếu refit = False.
- best_score_ (float): Điểm trung bình được cross-validation của best_estimator
- best_params_ (dict): Thông số thiết lập đó đã cho kết quả tốt nhất về giữ lại dữ liệu.

2.5.4. LightGBM (Máy tăng độ dốc xử lí nhanh)

LightGBM

Nó là một khung tăng cường độ dốc sử dụng các thuật toán học tập dựa trên cây được coi là một thuật toán rất mạnh khi nói đến tính toán. Nó được coi là một thuật toán xử lý nhanh

- Tốc độ huấn luyện nhanh hơn và hiệu quả cao hơn.
- Sử dụng bộ nhớ thấp hơn.
- Độ chính xác tốt hơn.
- Hỗ trợ học tập song song, phân tán và GPU.
- Có khả năng xử lý dữ liệu quy mô lớn.

Trong LightGBM chúng tôi sử dụng kỹ thuật Gradient Boosting Decision Tree (GBDT) được sử dụng rộng rãi trong học máy và đầu ra của các triển khai GBDT hiện tại là một biến duy nhất. Khi có nhiều đầu ra, GBDT xây dựng nhiều cây tương ứng

với các biến đầu ra. Trong trường hợp này, các mối tương quan giữa các biến bị bỏ qua bởi một ý tưởng như vậy gây ra sự dư thừa của các cấu trúc cây đã học, một phương pháp chung để học GBDT cho nhiều đầu ra, được gọi là GBDT-MO. Mỗi lá của GBDT-MO xây dựng các dự đoán của tất cả các biến hoặc một tập hợp con của các biến được chọn tự động. Điều này đạt được bằng cách xem xét tổng hợp các lợi ích khách quan trên tất cả các biến đầu ra. Và mở rộng xấp xỉ biểu đồ thành nhiều trường hợp đầu ra và tăng tốc quá trình huấn luyện bằng trường hợp mở rộng. Các thí nghiệm khác nhau trên các bộ dữ liệu tổng hợp và trong thế giới thực xác minh rằng cơ chế học tập của GBDT-MO đóng một vai trò trong quá trình chính quy hóa hay còn được hiểu là hiệu chỉnh (regularization) gián tiếp. GBDT sử dụng cây quyết định làm trình học cơ sở và tính tổng các dự đoán của một loạt cây. Ở mỗi bước, một cây quyết định mới được huấn luyện để phù hợp với phần còn lại giữa kết quả thực tế và dự đoán hiện tại. Nhiều cải tiến đã được đề xuất sau khi LightGBM tổng hợp thông tin gradient trong biểu đồ và cải thiện đáng kể hiệu quả huấn luyện.[40]

Một hạn chế của việc triển khai GBDT hiện tại là đầu ra của mỗi cây quyết định là một biến duy nhất. Điều này là do mỗi lá của cây quyết định tạo ra một biến duy nhất. Ở mỗi bước, xây dựng nhiều cây quyết định, mỗi cây tương ứng với một biến riêng lẻ của đầu ra, sau đó ghép các dự đoán của tất cả các cây để thu được nhiều đầu ra.[40]

Hạn chế chính của chiến lược nói trên là mối tương quan giữa các biến bị bỏ qua trong quá trình đào tạo vì các biến đó được xử lý riêng biệt và chúng được học một cách độc lập. Tuy nhiên, các mối tương quan ít nhiều tồn tại giữa các biến đầu ra. Ví dụ, có những mối tương quan giữa các lớp để phân loại nhiều lớp. Nó được xác minh rằng các mối tương quan như vậy cải thiện khả năng tổng quát hóa. Bỏ qua các tương quan biến cũng dẫn đến dư thừa các cấu trúc cây đã học. Vì vậy, cần phải học GBDT để có nhiều đầu ra thông qua các chiến lược tốt hơn. Sử dụng xấp xỉ gradient và biểu đồ bậc hai để cải thiện GBDT-MO. Cơ chế học tập được thiết kế dựa trên chúng để phù hợp với tất cả các biến trong một cây duy nhất. Mỗi lá của cây quyết định tạo ra nhiều đầu ra cùng một lúc. Điều này đạt được bằng cách tối đa hóa tổng số lợi ích khách quan (objective gains)trên tất cả các biến đầu ra. Kỳ vọng rằng phương pháp được đề xuất sẽ tự động chọn các biến đó và xây dựng các dự đoán cho chúng tại một lá. Chúng tôi đạt được điều này bằng cách thêm ràng buộc L_0 vào hàm mục tiêu. Vì cơ chế học tập của GBDT-MO thực thi các cây đã học để nắm bắt các tương quan biến đổi, nên nó đóng một vai trò trong quá trình chính quy hóa gián tiếp.[40]

GBDT là một mô hình tập hợp các cây quyết định, được huấn luyện theo trình tự. Trong mỗi lần lặp, GBDT học các cây quyết định bằng cách điều chỉnh các độ dốc âm (còn được gọi là lỗi dư) chia điểm. Một trong những thuật toán phổ biến nhất để tìm điểm phân tách là thuật toán sắp xếp trước liệt kê tất cả các điểm phân tách có thể có trên các giá trị của đối tượng được sắp xếp trước.[40]

Chúng tôi lấy được hàm mục tiêu của GBDT. Mỗi lá của một cây quyết định xây dựng nhiều đầu ra. Ký hiệu D = { $(x_i y_i)_{i=1}^{n}$} dưới dạng tập dữ liệu với n mẫu, trong đó x ∈ R^m là đầu vào m chiều và y ∈ R^d là đầu ra thứ nguyên thay vì vô hướng. Ký hiệu f: $R^m \rightarrow R^d$ là hàm của cây quyết định ánh xạ x vào không gian đầu ra. Dựa trên cơ chế xây dựng của cây quyết định, f có thể được biểu diễn thêm như sau:

$$f(x) = \boldsymbol{W}_{q(x)}, q: R^m \rightarrow [1, L], \boldsymbol{W} \in R^{Lxd} \quad (27)$$

- L là số lá của cây quyết định
- q là hàm chọn lá cho trước x
- W_i là giá trị của lá thứ i.

Nghĩa là, một khi cây quyết định được xây dựng, trước tiên nó ánh xạ một đầu vào vào một lá, sau đó trả về vectơ thứ nguyên d của lá đó. Sau đó, dự đoán của t cây đầu tiên là:

$$\hat{\boldsymbol{y}}_{\boldsymbol{i}} = \sum_{k=1}^{t} f(x_i) \quad (28)$$

Chúng ta coi mục tiêu của cây thứ (t + 1) là $\hat{y}$. Chúng ta chỉ xem xét mục tiêu của một lá đơn như sau, w ∈ R^d là một vectơ có d phần tử thuộc một chiếc lá. Một lần nữa, chúng ta giả sử l là một hàm phân biệt bậc hai $l(\hat{y}_i + w, y_i)$ có thể được tính gần đúng bằng khai triển Taylor bậc hai của $l(\hat{y}_i, y_i)$. Đặt $R(w) = \frac{1}{2}\|w\|_2^2$ ta có công thức:

$$L = \sum_i \{ l(\hat{\boldsymbol{y}}_{\boldsymbol{i}}, \boldsymbol{y}_{\boldsymbol{i}}) + (\boldsymbol{g})_i^T \boldsymbol{w} + \frac{1}{2} w^T (\boldsymbol{H})_i w \} + \gamma R(w) \quad (29)$$

- R là số hạng chính quy của f, và λ kiểm soát sự cân bằng giữa hai số hạng
- $(g)_i = \frac{\partial l}{\partial \hat{y}_i}$ và $(H)_i = \frac{\partial^2 l}{\partial \hat{y}_i^2}$. Để tránh xung đột ký hiệu với chỉ số con của vectơ hoặc ma trận, chúng tôi sử dụng $(\cdot)_i$ để chỉ ra rằng một đối tượng thuộc mẫu thứ i. Ký hiệu này được bỏ qua khi không có sự mơ hồ.

Bằng cách thiết lập $\frac{\partial L}{\partial \mathbf{w}} = 0$ đối với (36), chúng tôi nhận được các giá trị lá tối ưu:

$$\boldsymbol{w}* = -\left(\sum_i (\boldsymbol{H})_i + \lambda \boldsymbol{I}\right)^{-1} (\sum_i (\boldsymbol{g})_i) \quad (30)$$

I là ma trân đơn vị. Bằng cách thay thế $\boldsymbol{w}*$ từ và bỏ qua hằng số $l(\hat{\boldsymbol{y}}_{\boldsymbol{i}}, \boldsymbol{y}_{\boldsymbol{i}})$ chúng tôi nhận được hàm mục tiêu tối ưu như sau:

$$L^* = -\frac{1}{2}\left(\sum_i (g)_i\right)^T \left(\sum_i (\boldsymbol{H})_i + \lambda \boldsymbol{I}\right)^{-1} (\sum_i (g)_i) \quad (31)$$

Chúng tôi đã suy ra các giá trị lá tối ưu và mục tiêu tối ưu cho nhiều đầu ra. Trong thực tế, khi hàm mất **L** là có thể phân tách các kích thước đầu ra khác nhau, hoặc tương đương, khi ma trận hessian H của nó là đường chéo, mỗi phần tử của **w***

$$\widetilde{\boldsymbol{w}}_j^* = -\frac{\sum_i (\boldsymbol{g}_j)_i}{\sum_i (\boldsymbol{h})_i + \gamma} \quad (32)$$

$\boldsymbol{h} \in R^d$ là các yếu tố đường chéo của **H**. Và mục tiêu tối ưu trong (36) có thể được biểu thị bằng tổng các mục tiêu trên tất cả các kích thước đầu ra.

Tổng các mục tiêu trên tất cả các kích thước đầu ra:

$$\tilde{L}^* = -\frac{1}{2}\sum_{j=1}^{d}\left\{\frac{\sum_i (\boldsymbol{g}_j)_i^{\,2}}{\sum_i (\boldsymbol{h})_i + \gamma}\right\} \quad (33)$$

GBDT-MO xem xét các hàm mục tiêu của tất cả các biến đầu ra cùng một lúc. Tuy nhiên, vấn đề là khi l không phân tách được hoặc tương đương với **H** là không đường chéo. Đầu tiên, rất khó lưu trữ **H** cho mọi mẫu khi kích thước đầu ra d lớn. Thứ hai, để có được mục tiêu tối ưu cho mỗi lần tách có thể, cần phải tính toán nghịch đảo của ma trận d × d, việc này tốn nhiều thời gian. Do đó, việc học GBDT-MO bằng cách sử dụng mục tiêu chính xác trong (36) và các giá trị lá chính xác trong (35) là không thực tế. May mắn thay, nó được thể hiện ở chỗ $\tilde{L}^*$ và $\widetilde{\boldsymbol{w}}^*$ là giá trị xấp xỉ tốt của các giá trị lá chính xác và hàm mục tiêu chính xác khi các phần tử đường chéo của H bị chi phối $\tilde{L}^*$ và $\widetilde{\boldsymbol{w}}^*$ được bắt nguồn từ giới hạn trên của $l(\hat{y}_i, y_i)$.[40]

LightGBM đưa các giá trị của feature (thuộc tính) liên tục vào các thùng riêng biệt. Điều này tăng tốc độ huấn luyện và giảm mức sử dụng bộ nhớ. Ưu điểm của các thuật toán dựa trên biểu đồ bao gồm:

Giảm chi phí tính toán lợi nhuận cho mỗi lần tách

- Các thuật toán dựa trên sắp xếp trước có độ phức tạp về thời gian O(#data)
- Tính toán biểu đồ có độ phức tạp về thời gian O(#data), nhưng điều này chỉ liên quan đến một hoạt động tổng hợp nhanh chóng. Khi biểu đồ được xây

dựng, một thuật toán dựa trên biểu đồ có độ phức tạp về thời gian O(#bins)và #bins nhỏ hơn nhiều #data.[41]

Sử dụng phép trừ biểu đồ để tăng tốc hơn nữa

- Để có được biểu đồ của một chiếc lá trong cây nhị phân, hãy sử dụng phép trừ biểu đồ của lá cha và người hàng xóm của nó
- Vì vậy, nó cần phải xây dựng biểu đồ cho chỉ một lá (với nhỏ #data hơn người hàng xóm của nó). Sau đó, nó có thể lấy biểu đồ của người hàng xóm của nó bằng cách trừ biểu đồ với chi phí nhỏ (O(#bins))[41]

Giảm mức sử dụng bộ nhớ

- Thay thế các giá trị liên tục bằng các thùng rời rạc. Nếu #binsnhỏ, có thể sử dụng kiểu dữ liệu nhỏ, ví dụ: uint8_t, để sao lưu trữ dữ liệu huấn luyện
- Không cần lưu trữ thông tin bổ sung để sắp xếp trước các giá trị thuộc tính[41]

Giảm chi phí liên lạc cho việc học tập phân tán

Nhược Điểm: Việc chạy LightGBM trên GPU thực sự có vấn đề. Gói cài đặt mặc định không hỗ trợ GPU. Bạn phải xây dựng phân phối GPU và bạn có thể gặp rắc rối khi cài đặt.[41]

Tối ưu hóa về độ chính xác

LightGBM phát triển cây lá (tốt nhất) . Nó sẽ chọn lá bị rụng tối đa để phát triển. Giữ #leaf cố định.[41]

Sự khôn ngoan của lá có thể gây ra hiện tượng #data quá khít khi cây còn nhỏ, vì vậy LightGBM bao gồm max_depth thông số để giới hạn độ sâu của cây. Tuy nhiên, cây vẫn mọc lá ngay cả khi max_depth được chỉ định.[41]

Hỗ trợ thuộc tính phân loại

LightGBM cung cấp độ chính xác tốt với các thuộc tính phân loại được mã hóa số nguyên. LightGBM áp dụng Fisher (1958) để tìm ra sự phân chia tối ưu cho các danh mục như được mô tả ở đây . Điều này thường hoạt động tốt hơn mã hóa one-hot.

Sử dụng categorical_feature để chỉ định các đối tượng địa lý phân loại. Tham khảo tham số categorical_feature trong Tham số.

Đối tượng phân loại phải được mã hóa dưới dạng số nguyên không âm (int) nhỏ hơn Int32.MaxValue(2147483647). Tốt nhất là sử dụng một dải số nguyên liền nhau bắt đầu từ số không.

Sử dụng min_data_per_group, cat_smooth để đối phó với việc Overfitting (khi #data nhỏ hoặc #category lớn).

Đối với đối tượng phân loại có số lượng cao (#category lớn), cách tốt nhất là coi đối tượng này là số, bằng cách đơn giản bỏ qua việc giải thích phân loại các số nguyên hoặc bằng cách nhúng các danh mục trong không gian số chiều thấp.[41]

Thông số mô hình[41]***:***

Chúng ta có thể thiết lập một số thông số cơ bản cho mô hình.

1. “num_leaves”: Đại diện cho số lá cây sẽ tạo ra trong khi huấn luyện.
2. “mục tiêu”: Để xác định xem phân loại có phải là phân loại nhị phân của phân loại nhiều lớp hay không.
3. subsample [default=1]: phạm vi (0,1], có thể chỉ định phần trăm số hàng được sử dụng cho mỗi lần lặp lại việc xây dựng cây. Điều đó có nghĩa là một số hàng sẽ được chọn ngẫu nhiên để phù hợp với từng cây. Điều này cải thiện khả năng tổng quát hóa mà còn tăng tốc độ huấn luyện.
4. max_depth [default = 6]: Phạm vi [0,∞]

   Tham số này kiểm soát độ sâu tối đa của mỗi cây được huấn luyện và sẽ có tác động đến:

   - Giá trị tốt nhất cho tham số num_leaves
   - Hiệu suất mô hình
   - Thời gian huấn luyện
5. colsample_bytree [default=1]: Là tỷ lệ mẫu con của các cột khi xây dựng mỗi cây. Việc lấy mẫu con xảy ra một lần cho mỗi cây được xây dựng.
6. n_estimators [default=100]: Đây là số lượng mẫu mà thuật toán này sẽ hoạt động, sau đó nó sẽ tổng hợp chúng lại để đưa ra câu trả lời cuối cùng cho bạn mà bạn muốn xây dựng trước khi lấy phiếu bầu tối đa hoặc trung bình của các dự đoán. Số lượng cây cao hơn mang lại cho bạn hiệu suất tốt hơn nhưng làm cho mã của bạn chậm hơn.
7. min_split_gain [default=0]: giảm tổn thất tối thiểu để tạo ra một phân vùng sâu hơn trên một nút lá của cây. Giá trị thấp hơn sẽ dẫn đến cây sâu hơn.

8. min_child_weight [default=1]: phạm vi [0, ∞], tổng trọng lượng cá thể tối thiểu (hessian) cần thiết trong một con (lá). Nếu bước phân vùng cây dẫn đến một nút lá có tổng trọng lượng cá thể nhỏ hơn min_child_weight, thì quá trình xây dựng sẽ từ bỏ việc phân vùng tiếp theo.
9. scale_pos_weight [default = 1]: Kiểm soát sự cân bằng của trọng số âm và dương, hữu ích cho các lớp không cân bằng. Giá trị điển hình cần xem xét: sum (trường hợp âm) / sum (trường hợp dương).
10. min_data_in_leaf [default=1]: Đây là một thông số rất quan trọng để ngăn ngừa sự phù hợp quá mức ở cây lá khôn. Giá trị tối ưu của nó phụ thuộc vào số lượng mẫu huấn luyện và num_leaves là một trong những tham số quan trọng nhất kiểm soát độ phức tạp của mô hình. Với nó, bạn đặt số lá tối đa mà mỗi người học yếu có. Số lượng lớn giúp tăng độ chính xác trong quá trình tập luyện và giảm nguy cơ bị sai lệch do overfiting num_leaves.
11. learning_rate [default = 6]: Tăng tốc độ học tập.
12. colsample_bytree (default = 1): Tỷ lệ mẫu con của các cột khi xây dựng mỗi cây.
13. random_state (default = none): Hạt giống số ngẫu nhiên
14. n_jobs [default= -1]: Số luồng song song.

Dự đoán mô hình[41]***:***

Chúng tôi có thể dự đoán kết quả đầu ra bằng mô hình LightGBM đã được huấn luyện bằng phương pháp predict( ) . Đầu ra sẽ nằm trong xác suất dự đoán, do đó chúng ta cần sử dụng hàm round( ) để phân loại nhị phân.

Điều chỉnh siêu tham số cho LGBM chúng tôi dùng RandomizedSearchCV với các thông số như sau[41]***:***

- estimator (đối tượng dự đoán): Một đối tượng thuộc được khởi tạo cho mỗi điểm lưới. Công cụ ước tính cần cung cấp một hàm điểm hoặc phải cho điểm được thông qua.
- param_distributions (mệnh lệnh hoặc danh sách các mệnh lệnh):Từ điển với tên tham số (str) làm khóa và phân phối hoặc danh sách các tham số để thử. Các bản phân phối phải cung cấp phương pháp rvs để lấy mẫu.
- scoring (str, callable, list, tuple or dict, default=None): Chiến lược đánh giá hiệu suất của mô hình được xác nhận chéo trên tập thử nghiệm.
- n_jobs (int, default = None): Số lượng công việc phải chạy song song.

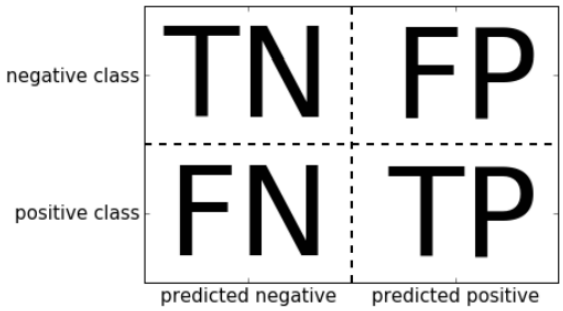
- refit (bool, str or callable, default = True): Trang bị lại một công cụ ước tính bằng cách sử dụng các tham số tìm được tốt nhất trên toàn bộ tập dữ liệu.
- cv (int, trình tạo xác thực chéo hoặc có thể lặp lại, default=None): Xác định chiến lược phân tách xác thực chéo.
- Verbose(int, Kiểm soát độ dài): càng cao, càng nhiều tin.
- pre_dispatch (int, or str, default=None): Kiểm soát số lượng công việc được gửi đi trong quá trình thực hiện song song. Giảm con số này có thể hữu ích để tránh sự bùng nổ tiêu thụ bộ nhớ khi nhiều công việc được gửi hơn mức CPU có thể xử lý.
- random_state (int, RandomState instance or None, default=None): Trạng thái tạo số ngẫu nhiên giả được sử dụng để lấy mẫu đồng nhất ngẫu nhiên từ danh sách các giá trị có thể có thay vì phân phối scipy.stats. Chuyển một int cho đầu ra có thể tái tạo qua nhiều lệnh gọi hàm.
- error_score('raise' or numeric, default=np.nan): Giá trị để gán cho điểm số nếu có lỗi xảy ra khi lắp công cụ ước tính.
- return_train_score (bool, default=False): Nếu False, cv_results_ thuộc tính sẽ không bao gồm điểm huấn luyện. Tính toán điểm số huấn luyện được sử dụng để hiểu rõ về cách các cài đặt thông số khác nhau ảnh hưởng đến sự cân bằng của việc trang bị quá mức / thiếu trang bị.

Và cá thuộc tính của RandomizedSearchCV[42]:

- cv_results_ (mệnh lệnh của numpy ndarrays): Một mệnh lệnh có các khóa làm tiêu đề cột và giá trị dưới dạng cột, có thể được nhập vào DataFrame.
- best_estimator_ (ước tính): Công cụ ước tính được chọn bởi tìm kiếm, tức là công cụ ước tính cho điểm cao nhất (hoặc tổn thất nhỏ nhất nếu được chỉ định) trên dữ liệu bị bỏ sót.
- best_score_ (float): Điểm trung bình được xác thực chéo của best_estimator.
- best_params_ (dict): Cài đặt thông số mang lại kết quả tốt nhất về dữ liệu tạm ngưng.

2.6. Các công thức đánh giá mô hình học máy[3]

Ma trận nhầm lẫn (Confusion matrices): Các số trên đường chéo chính của ma trận nhầm lẫn tương ứng với các phân loại chính xác, trong khi các mục nhập khác cho chúng ta biết có bao nhiêu mẫu của một lớp đã bị phân loại nhầm thành một lớp khác.



Hình 2- 15: Ma trận nhầm lẫn để phân loại nhị phân.[38]

Với 0 là negative class và 1 là positive class (0 với 1 là label trong tập dữ liệu đánh giá tín dụng cá nhân mà sẽ được sử dụng sau đây, với nhãn 0 sẽ là những khách hàng không có khả năng vỡ nợ và nhãn 1 sẽ là những khách hàng có khả năng vỡ nợ) TN: true negative là nhãn 0 thật và TP: true positive là nhãn 1 thật, FN: false negative là những nhãn 0 dự đoán nhầm, tức là thực chất nó phải là nhãn 1 và FP là false positive là những nhãn 1 dự đoán nhầm, tức là nó phải là nhãn 0.

Liên quan tới accuracy: Phần trăm (hoặc tỷ lệ) các trường hợp được phân loại chính xác, là một cách để tóm tắt kết quả trong ma trận nhầm lẫn - bằng độ chính xác của tính toán:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{34}$$

độ chính xác là số lần dự đoán đúng (TP và TN) được chia bằng số lượng của tất cả các mẫu (tất cả các mục của ma trận nhầm lẫn được cộng lại).

Có một số cách khác để tóm tắt ma trận nhầm lẫn, với những cách phổ biến nhất là precision và recall.

Precision đo lường bao nhiêu mẫu được dự đoán là nhãn 1 thực sự là 1:

$$Precision = \frac{TP}{TP + FP} \tag{35}$$

Precision được sử dụng làm thước đo hiệu suất khi mục tiêu là hạn chế số lần dự đoán sai nhãn 1.Nó còn được biết dưới cái tên là dự đoán giá trị positive hoặc dự đoán giá trị nhãn 1 (positive predictive value-PPV).

Mặt khác, recall, đo lường có bao nhiêu mẫu là nhãn 1 được thu thập bằng các dự đoán nhãn 1:

$$Recall = \frac{TP}{TP + FN} \tag{36}$$

Recall được sử dụng làm thước đo hiệu suất khi chúng ta cần xác định tất cả các mẫu là nhãn 1, nghĩa là khi đó điều quan trọng là tránh các dự đoán nhầm nhãn 1 thành nhãn 0. Một số tên gọi khác như là sensitivity(nhạy cảm), hit rate(tỷ lệ trúng), true positive rate (TPR).

Có một sự đánh đổi giữa việc tối ưu hóa recall và tối ưu hóa precision. Bạn có thể đạt điểm recall hoàn hảo theo cách ít giá trị sử dụng nếu bạn dự đoán tất cả các mẫu thuộc loại nhãn 1 sẽ không có dự đoán nhầm sang nhãn 0 và cũng không có nhãn 0. Tuy nhiên, dự đoán tất cả các mẫu là nhãn 1 sẽ dẫn đến nhiều kết quả dự đoán nhầm sang nhãn 1 và do đó precision sẽ rất thấp. Mặt khác, nếu bạn tìm thấy một mô hình chỉ dự đoán một điểm dữ liệu đơn lẻ mà nó chắc chắn nhất là nhãn 1 và phần còn lại là nhãn 0, thì precision sẽ là hoàn hảo (giả sử điểm dữ liệu này trên thực tế là nhãn 1), nhưng điểm recall sẽ rất tệ. Trong khí precision và recall là những thước đo rất quan trọng, chỉ nhìn vào một trong số chúng sẽ không cung cấp cho bạn bức tranh đầy đủ. Một cách để tóm tắt chúng là điểm số f hoặc số đo f, với giá trị trung bình hài hòa của precision và recall:

$$F = 2 \,.\, \frac{precision.recall}{precision + recall} \tag{37}$$

Biến thể cụ thể này còn được gọi là f1 score. Vì nó có tính đến precision và recall, nó có thể là một thước đo tốt hơn accuracy trên bộ dữ liệu phân loại nhị phân không cân bằng. Sử dụng f-score để đánh giá, chúng tôi đã tóm tắt lại hiệu suất dự đoán trong một con số. Tuy nhiên, f-score dường như nắm bắt được trực giác của chúng ta về điều gì làm cho một mô hình tốt tốt hơn nhiều so với độ chính xác đã làm. Tuy nhiên, một nhược điểm của f-score là khó diễn giải và giải thích hơn là độ chính xác. Nếu chúng ta muốn có một bản tóm tắt toàn diện hơn về precision, recall và f1-score chúng ta có thể sử dụng một hàm tiện lợi classification_report để tính toán cả ba cùng một lúc và in chúng ở định dạng đẹp.

Receiver operating characteristics (ROC) và AUC

Thường được sử dụng để phân tích hành vi của bộ phân loại ở các ngưỡng khác nhau: gọi là receiver operating characteristics (ROC), đường cong ROC xem xét tất cả các ngưỡng có thể có cho một bộ phân loại nhất định. Nó cho thấy tỷ lệ dự đoán nhầm

nhãn 1 (FPR) so với tỷ lệ nhãn 1 thật (TPR). Recall là tỷ lệ nhãn 1 thật, trong khi tỷ lệ dự đoán nhầm nhãn 1 còn được gọi là specificity là phần của nhãn 0 nhưng dự đoán nhầm thành nhãn 1 trong số tất cả các mẫu là nhãn 0:

$$FPR = \frac{FP}{FP + TN} \tag{38}$$

Đối với đường cong ROC, đường cong lý tưởng nằm gần trên cùng bên trái: nghĩa là một bộ phân loại tạo ra điểm recall cao trong khi vẫn giữ tỷ lệ dự đoán nhầm nhãn 1 thấp. Điểm gần trên cùng bên trái nhất có thể là điểm hoạt động tốt hơn điểm được chọn theo mặc định. Biểu đồ đường cong ROC recall trên trục y so với FPR trên trục x.

Tuy nhiên, đường cong ROC có thể được sử dụng để tạo ra diện tích bên dưới chỉ số đường cong (AUC). AUC chỉ đơn giản là tổng diện tích dưới đường cong ROC. Giá trị AUC càng lớn thì bộ phân loại càng hiệu quả. AUC bằng 1 cho biết một bộ phân loại hoàn hảo: nó nhận được tất cả các số 1 được phân loại chính xác và nó không phân loại sai bất kỳ số 0 nào là số 1.

“macro” tính trung bình tính toán không trọng số cho các f-score mỗi lớp. Điều này mang lại trọng lượng như nhau cho tất cả các lớp, bất kể kích thước của chúng là bao nhiêu.

“weighted” tính trung bình giá trị trung bình của các f-score mỗi lớp. Đây là những gì được báo cáo trong báo cáo phân loại.

2.7. Metrics

Hàm mean_squared_error tính toán lỗi bình phương trung bình, một chỉ số rủi ro tương ứng với giá trị kì vọng của lỗi hoặc mất mát bình phương (bậc hai). Nếu $\hat{y}_i$ là giá trị dự đoán của mẫu thứ i và y_i là giá trị thực tương ứng, thì sai số bình phương trung bình (MSE) được ước tính trên $n_{mẫu}$ được xác định là[43]:

$$MSE(y, \hat{y}) = \frac{1}{n_{mẫu}} \sum_{i=0}^{n_{mẫu}-1} (y_i - \hat{y}_i)^2 \tag{39}$$

RMSE là lấy căn bậc 2 của MSE.

Hàm mean_squared_log_error tính toán số liệu rủi ro tương ứng với giá trị kỳ vọng của lỗi hoặc mất mát logarit bình phương (bậc hai). Nếu $\hat{y}_i$ là giá trị dự đoán của mẫu thứ i và y_i là giá trị thực tương ứng, thì sai số logarit bình phương trung bình (MSLE) được ước tính trên $n_{mẫu}$ được xác định là[43]:

$$MSLE(y,\hat{y}) = \frac{1}{n_{mẫu}} \sum_{i=0}^{n_{mẫu}-1} (log_e(1+y_i) - log_e(1+\hat{y}_i))^2 \quad (40)$$

Trong $log_e(x)$ đó có nghĩa là lôgarit tự nhiên của x. Chỉ số này tốt nhất nên sử dụng khi các mục tiêu có tốc độ tăng trưởng theo cấp số nhân, chẳng hạn như số lượng dân số, doanh số bán hàng trung bình trong một khoảng thời gian dài, v.v. Lưu ý rằng chỉ số này phạt một ước tính dưới dự đoán lớn hơn ước tính quá dự đoán. RMSLE là lấy căn bậc 2 của MSLE.

Hàm mean_absolute_error tính toán lỗi tuyệt đối trung bình, một chỉ số rủi ro tương ứng với giá trị kỳ vọng của tổn thất lỗi tuyệt đối hoặc tổn thất L1-norm(L1 chuẩn). Nếu $\hat{y}_i$ là giá trị dự đoán của mẫu thứ i và y_i là giá trị thực tương ứng, thì sai số tuyệt đối trung bình (MAE) được ước tính trên $n_{mẫu}$ được định nghĩa là:

$$MAE(y,\hat{y}) = \frac{1}{n_{mẫu}} \sum_{i=0}^{n_{mẫu}-1} |y_i - \hat{y}_i| \quad (41)$$

Một số đo F_β đạt giá trị tốt nhất của nó là 1 và điểm kém nhất của nó là 0. Với $\beta = 1$ F_1 và F_β tương đương, và recall và precision đều quan trọng như nhau. Điểm F-beta là giá trị trung bình hài hào có trọng số giữa precision và recall, đạt giá trị tối ưu ở 1 và giá trị xấu nhất ở 0. Tham số beta xác định trọng số recall trong điểm tổng hợp. beta <1 cho precision cao hơn, trong khi beta> 1 cho recall cao hơn (beta -> 0 chỉ xem xét precision, beta -> +∞ chỉ xem xét duy nhất recall).[44]

Chương 3: Thực nghiệm và kết quả đánh giá nghiên cứu

3.1. Các thư viện cơ bản

Nên sử dụng một trong các bản phân phối Python được đóng gói sẵn sau đây, bản phân phối này sẽ cung cấp các gói cần thiết:

Anaconda: Một bản phân phối Python được tạo ra để xử lý dữ liệu quy mô lớn, phân tích dự đoán và tính toán khoa học. Anaconda đi kèm với NumPy, SciPy, matplotlib, pandas, IPython, Jupyter Notebook và scikit-learning.

Jupyter Notebook: là một môi trường tương tác để chạy mã trong trình duyệt, dễ dàng kết hợp mã, văn bản và hình ảnh này trên thực tế được viết dưới dạng sổ tay Jupyter.

NumPy : NumPy là một trong những gói cơ bản cho tính toán khoa học bằng Python. Nó chứa các chức năng cho mảng nhiều chiều, các hàm toán học cấp cao như các phép toán đại số tuyến tính và phép biến đổi Fourier, và các bộ tạo số giả ngẫu nhiên.

Scipy: là một tập hợp các hàm cho tính toán khoa học bằng Python. Nó cung cấp, trong số các chức năng khác, các quy trình đại số tuyến tính nâng cao, tối ưu hóa hàm toán học, xử lý tín hiệu, các hàm toán học đặc biệt và phân phối thống kê. Phần quan trọng nhất của SciPy đối với chúng tôi là scipy.sparse: phần này cung cấp các ma trận thưa thớt, là một biểu diễn khác được sử dụng cho dữ liệu trong scikitlearn. Ma trận thưa thớt được sử dụng bất cứ khi nào chúng ta muốn lưu trữ một mảng 2D chứa hầu hết các số không.

Matplotlib: matplotlib là thư viện vẽ sơ đồ khoa học chính bằng Python. Nó cung cấp các chức năng để tạo hình ảnh chất lượng xuất bản như biểu đồ đường, biểu đồ, biểu đồ phân tán, v.v. Khi làm việc bên trong Jupyter Notebook bạn có thể hiển thị các số liệu trực tiếp trong trình duyệt bằng cách sử dụng lệnh %matplotlib notebook và %matplotlib inline.

Pandas: pandas là một thư viện Python để phân tích và xử lý dữ liệu. Nó được xây dựng xung quanh một cấu trúc dữ liệu được gọi là DataFrame. Nói một cách đơn giản, DataFrame của gấu trúc là một bảng, tương tự như một bảng tính Excel. pandas cung cấp một loạt các phương pháp để sửa đổi và thao tác trên bảng này; đặc biệt, nó cho phép các truy vấn giống SQL và nối các bảng. Ngược lại với NumPy, yêu cầu tất cả các mục nhập trong một mảng phải cùng loại, pandas cho phép mỗi cột có một kiểu riêng biệt (ví dụ: số nguyên, ngày tháng, số dấu phẩy động và chuỗi). Một công cụ có

giá trị khác do pandas cung cấp là khả năng nhập từ nhiều định dạng tệp và cơ sở dữ liệu, như tệp SQL, Excel và tệp giá trị được phân tách bằng dấu phẩy (CSV).

Seaborn là một thư viện để tạo đồ họa thống kê bằng Python. Nó được xây dựng dựa trên matplotlib và tích hợp chặt chẽ với cấu trúc dữ liệu của pandas.

random: tạo ra một số ngẫu nhiên

vai trò os: tương tác với hệ thống

vai trò sys: tương tác với python

3.2. Kết quả thực nghiệm

Ở đây chúng tôi bỏ cột unnamed: 0 vì đấy là index trong 2 tập và không quan trọng trong việc dự đoán mô hình.

Khai phá dữ liệu:

Thông tin về tập train:

RangeIndex: 150000 entries, 0 to 149999

Cột dữ liệu (tổng 11 cột):

Bảng 3- 1: Thông tin về tập train

#	*Cột*	*Số lượng Non-Null*	*Dạng dữ liệu*
0	Thời gian trễ hạn nghiêm trọng trong 2 năm	150000 non-null	int64
1	Sử dụng quay vòng của hạn mức tín dụng không có bảo đảm	150000 non-null	float64
2	Tuổi	150000 non-null	int64
3	Số thời gian 30-59 ngày quá hạn không tệ hơn	150000 non-null	int64
4	Tỷ lệ nợ	150000 non-null	float64
5	Thu nhập hàng tháng	120269 non-null	float64
6	Số lượng hạn mức tín dụng mở và các khoản cho vay	150000 non-null	int64
7	Số lần trễ 90 ngày	150000 non-null	int64
8	Đánh số các khoản vay hoặc dòng bất động sản	150000 non-null	int64
9	Số thời gian 60-89 ngày quá hạn không tệ hơn	150000 non-null	int64
10	Số người phụ thuộc	146076 non-null	float64

Dạng dữ liệu: float64(4), int64(7)

Bộ nhớ được sử dụng: 12.6 MB

Chúng tôi thấy dữ liệu ở trên file train ở dưới dạng int và float, và feature thu nhập hàng tháng và số người phụ thuộc có dữ liệu bị thiếu. Có 150000 mẫu dữ liệu đã được gán nhãn.

Bảng 3- 2: Sử dụng lệnh describe() để kiểm tra mô tả thống kế bao gồm những thống kê tóm tắt xu hướng trung tâm, sự phân tán và hình dạng phân phối của tập dữ liệu train với 5 feature

	Thời gian trễ hạn nghiêm trọng trong 2 năm	***Sử dụng quay vòng của hạn mức tín dụng không có bảo đảm***	***Tuổi***	***Số thời gian 30-59 ngày quá hạn không tệ hơn***	***Tỷ lệ nợ***
số lượng	150000	150000	150000	150000	150000
giá trị trung bình	0.066840	6.048438	52.295207	0.421033	353.005076
độ lệch chuẩn	0.249746	249.755371	14.771866	4.192781	2037.818523
min	0	0	0	0	0
25%	0	0.029867	41	0	0.175074
50%	0	0.154181	52	0	0.366508
75%	0	0.559046	63	0	0.868254
max	1	50708	109	98	329664

Bảng 3- 3: Sử dụng lệnh describe() để kiểm tra mô tả thống kế bao gồm những thống kê tóm tắt xu hướng trung tâm, sự phân tán và hình dạng phân phối của tập dữ liệu train với 6 feature tiếp theo

	Thu nhập hàng tháng	***Số lượng hạn mức tín dụng mở và các khoản cho vay***	***Số lần trễ 90 ngày***	***Đánh số các khoản vay hoặc dòng bất động sản***	***Số thời gian 60-89 ngày quá hạn không tệ hơn***	***Số người phụ thuộc***
số lượng	120269	150000	150000	150000	150000	146076
giá trị trung bình	6670.221	8.452760	0.265973	1.018240	0.240387	0.757222
độ lệch chuẩn	14384.67	5.145951	4.169304	1.129771	4.155179	1.115086
min	0	0	0	0	0	0
25%	3400	5	0	0	0	0
50%	5400	8	0	1	0	0
75%	8249	11	0	2	0	1
max	3008750	58	98	54	98	20

Sau khi sử dụng lệnh trên chúng tôi thấy được các feature: sử dụng quay vòng của hạn mức tín dụng không có bảo đảm, số thời gian 30-59 ngày quá hạn không tệ hơn, tỷ lệ nợ, số lần trễ 90 ngày, số thời gian 60-89 ngày quá hạn không tệ hơn có giá trị max cao bất thường.

RangeIndex: 101503 entries, 0 to 101502

Cột dữ liệu (tổng 11 cột):

Bảng 3- 4: Thông tin về tập test

#	*Cột*	*Số lượng Non-Null*	*Dạng dữ liệu*
0	Thời gian trễ hạn nghiêm trọng trong 2 năm	0 non-null	float64
1	Sử dụng quay vòng của hạn mức tín dụng không có bảo đảm	101503 non-null	float64
2	Tuổi	101503 non-null	int64
3	Số thời gian 30-59 ngày quá hạn không tệ hơn	101503 non-null	int64
4	Tỷ lệ nợ	101503 non-null	float64
5	Thu nhập hàng tháng	81400 non-null	float64
6	Số lượng hạn mức tín dụng mở và các khoản cho vay	101503 non-null	int64
7	Số lần trễ 90 ngày	101503 non-null	int64
8	Đánh số các khoản vay hoặc dòng bất động sản	101503 non-null	int64
9	Số thời gian 60-89 ngày quá hạn không tệ hơn	101503 non-null	int64
10	Số người phụ thuộc	98877 non-null	float64

Dạng dữ liệu: float64(5), int64(6)

Bộ nhớ được sử dụng: 8.5 MB

Ở file test, chúng tôi thấy dữ liệu ở dưới dạng int và float, và feature thu nhập hàng tháng và số người phụ thuộc có dữ liệu bị thiếu. Có 101503 mẫu dữ liệu chưa được gán nhãn.

Bảng 3- 5: Sử dụng lệnh describe() để kiểm tra mô tả thống kế bao gồm những thống kê tóm tắt xu hướng trung tâm, sự phân tán và hình dạng phân phối của tập dữ liệu test với 5 feature

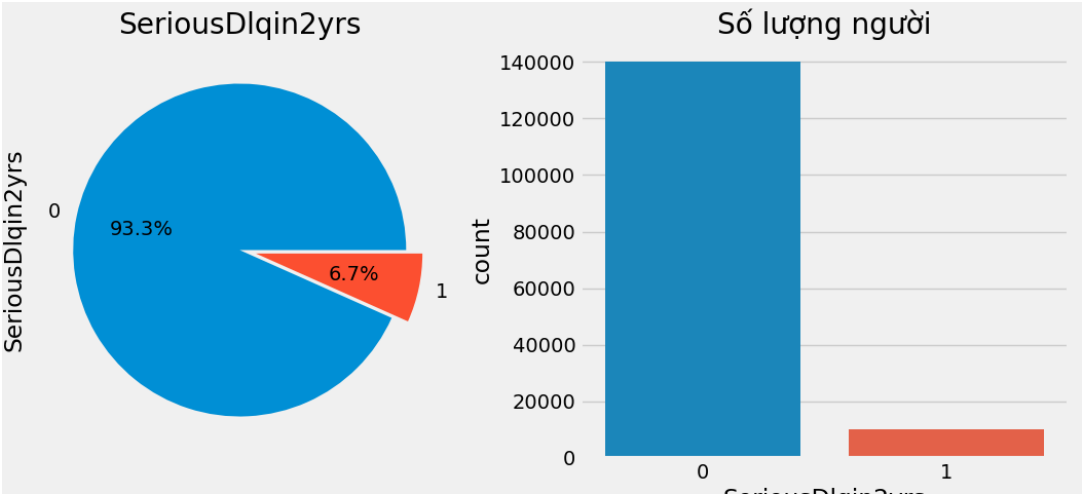
	Thời gian trễ hạn nghiêm	*Sử dụng quay vòng của hạn mức tín dụng*	*Tuổi*	*Số thời gian 30-59 ngày*	*Tỷ lệ nợ*

	trọng trong 2 năm	*không có bảo đảm*		*quá hạn không tệ hơn*	
số lượng	0.0	101503	101503	101503	101503
giá trị trung bình	NaN	5.31	52.40	0.45	344.47
độ lệch chuẩn	NaN	196.15	14.77	4.53	1632.59
min	NaN	0.00	21.00	0	0
25%	NaN	0.03	41.00	0	0.17
50%	NaN	0.15	52.00	0	0.36
75%	NaN	0.56	63.00	0	0.85
max	NaN	21821	104.00	98	268326

Bảng 3- 6: Sử dụng lệnh describe() để kiểm tra mô tả thống kế bao gồm những thống kê tóm tắt xu hướng trung tâm, sự phân tán và hình dạng phân phối của tập dữ liệu test với 6 feature

	Thu nhập hàng tháng	*Số lượng hạn mức tín dụng mở và các khoản cho vay*	*Số lần trễ 90 ngày*	*Đánh số các khoản vay hoặc dòng bất động sản*	*Số thời gian 60-89 ngày quá hạn không tệ hơn*	*Số người phụ thuộc*
số lượng	81400	101503	101503	101503	101503	98877
giá trị trung bình	6855.03	8.45	0.29	1.01	0.27	0.76
độ lệch chuẩn	36508.6	5.14	4.51	1.11	4.50	1.13
min	0	0	0	0	0	0
25%	3408.60	5	0	0	0	0
50%	5400	8	0	1	0	0
75%	8200	11	0	2	0	1
max	7727000	85	98	37	98	43

Ở tập test, chúng tôi thấy vấn đề tương tự file train.



Hình 3- 1: Số lượng nhãn 0 và 1 trong file train

Sau đó chúng tôi kiểm tra số lượng nhãn đã gán ở trên file train với label 1 là số người vỡ nợ và label 0 là số người không bị vỡ nợ. Với số lượng là:

- 0: 139974
- 1: 10026

Tỷ lệ của các trường hợp ngoại lệ 0 và 1 được tìm thấy là 93,3% đến 6,7%, xấp xỉ tỷ lệ 14: 1. Do đó, tập dữ liệu rất mất cân đối. Không thể dựa vào điểm chính xác để dự đoán thành công mô hình.

3.2.1. Tiền xử lí dữ liệu

Ở đây chúng tôi thấy được trong các cột số thời gian 30-59 ngày quá hạn không tệ hơn, số thời gian 60-89 ngày quá hạn không tệ hơn và số lần trễ 90 ngày, có phạm vi quá hạn vượt quá 90 phổ biến trên cả 3 feature.Có một số giá trị cao bất thường cho tỷ lệ nợ và sử dụng quay vòng của hạn mức tín dụng không có bảo đảm. Những feature trên đều có giá trị ngoại lai. Trước tiên chúng tôi sẽ xem xem giá trị ngoại lai đó là những giá trị nào:

- Những giá trị duy nhất trong những giá trị '30-59 ngày' nhiều hơn hoặc bằng 90: [96 98]
- Những giá trị duy nhất trong những giá trị '60-89 ngày' khi '30-59 ngày' nhiều hơn hoặc bằng 90: [96 98]
- Những giá trị duy nhất trong những giá trị '90 ngày' khi '30-59 ngày' nhiều hơn hoặc bằng 90: [96 98]
- Những giá trị duy nhất trong những giá trị '60-89 ngày' khi '30-59 ngày' ít hơn 90: [0 1 2 3 4 5 6 7 8 9 11]

- Những giá trị duy nhất trong những giá trị '90 ngày' khi '30-59 ngày' ít hơn 90: [0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17]
- Tỷ lệ của lớp 1 với những giá trị đặc biệt 96,98: 54.65 %

Có thể thấy điều sau rằng khi các bản ghi trong cột 'số thời gian 30-59 ngày quá hạn không tệ hơn ' lớn hơn 90, các cột khác ghi lại số lần các khoản thanh toán quá hạn X ngày cũng có cùng giá trị. Chúng tôi sẽ phân loại những nhãn này là nhãn đặc biệt vì tỷ lệ lớp cao bất thường ở mức 54,65%.

Các giá trị 96 và 98 này có thể được xem là lỗi kế toán. Do đó, sẽ thay thế chúng bằng giá trị lớn nhất trước 96, tức là 13, 11 và 17, hoặc là loại bỏ nó. Vì có ít dữ liệu nên chúng tôi sẽ chọn phương pháp thay thế chúng.

Và thay thế chúng ta sẽ có kết quả như sau:

- Các giá trị duy nhất trong 30-59Days [0 1 2 3 4 5 6 7 8 9 10 11 12 13]
- Các giá trị duy nhất trong 60-89Days [0 1 2 3 4 5 6 7 8 9 11]
- các giá trị duy nhất trong 90Days [0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17]

Và tương tự với file test kiểm tra giá trị ngoại lai đó là những giá trị nào:

- Những giá trị duy nhất trong những giá trị ‘30-59 ngày’ nhiều hơn hoặc bằng 90: [96 98]
- Những giá trị duy nhất trong những giá trị ‘60-89 ngày’ khi '30-59 ngày' nhiều hơn hoặc bằng 90: [96 98]
- Những giá trị duy nhất trong những giá trị ‘90 ngày’ khi '30-59 ngày' nhiều hơn hoặc bằng 90: [96 98]
- Những giá trị duy nhất trong những giá trị ‘30-59 ngày’ ít hơn 90: [0 1 2 3 4 5 6 7 8 9 10 11 12 19]
- Những giá trị duy nhất trong những giá trị ‘60-89 ngày’ khi '30-59 ngày' ít hơn 90: [0 1 2 3 4 5 6 7 8 9]
- Những giá trị duy nhất trong những giá trị ‘90 ngày’ khi ‘30-59 ngày’ ít hơn 90: [0 1 2 3 4 5 6 7 8 9 10 11 12 16 17 18]

Và thay thế chúng ta sẽ có kết quả như sau:

- Các giá trị duy nhất trong ‘30-59 ngày’ [0 1 2 3 4 5 6 7 8 9 10 11 12 19]
- Các giá trị duy nhất trong ‘60-89 ngày’ [0 1 2 3 4 5 6 7 8 9]

- các giá trị duy nhất trong ‘90 ngày’ [0 1 2 3 4 5 6 7 8 9 10 11 12 16 17 18]

Sau khi xử lí xong các feature trên chúng tôi xử lí 2 feature có giá trị cao bất thường là tỷ lệ nợ và sử dụng quay vòng của hạn mức tín dụng không có bảo đảm.

Với feature tỷ lệ nợ chúng tôi sẽ kiểm tra xem nó có giá trị tăng nhanh bắt đầu từ đâu bằng cách tính quantile.

Tính quantile trong feature tỷ lệ nợ trên file train:

- 75.0 % quantile của tỷ lệ nợ: 0.86825377325
- 80.0 % quantile của tỷ lệ nợ: 4.0
- 81.0 % quantile của tỷ lệ nợ: 14.0
- 85.0 % quantile của tỷ lệ nợ: 269.1499999999942
- 90.0 % quantile của tỷ lệ nợ: 1267.0
- 95.0 % quantile của tỷ lệ nợ: 2449.0
- 97.5 % quantile của tỷ lệ nợ: 3489.024999999994
- 99.0 % quantile của tỷ lệ nợ: 4979.040000000037

Giá trị bắt đầu tăng nhanh từ quantile 81%, vì vậy sẽ phải kiểm tra giá trị ngoại lai từ quantile 81%, nhưng vì dữ liệu chỉ có 150000 mẫu nên sẽ xem xét từ 95% để phân tích.

Bảng 3- 7: Mô tả những dữ liệu trong feature thời gian trễ hạn nghiêm trọng trong 2 năm và thu nhập hàng tháng trên file train có quantile trên feature Tỷ lệ Nợ lớn hơn hoặc bằng 95%

Thời gian trễ hạn nghiêm trọng trong 2 năm	***Thu nhập hàng tháng***
số lượng	7501.000000	379.000000
giá trị trung bình	0.055193	0.084433
độ lệch chuẩn	0.228371	0.278403
min	0.000000	0.000000
25%	0.000000	0.000000
50%	0.000000	0.000000
75%	0.000000	0.000000
max	1.000000	1.000000

Ở đây quan sát được: Trong số 7501 khách hàng có tỷ lệ nợ lớn hơn 95%, tức là số lần nợ cao hơn thu nhập của họ, chỉ có 379 khách hàng có giá trị thu nhập hàng tháng. Giá trị tối đa cho thu nhập hàng tháng là 1 và tối thiểu là 0, liệu có phải là lỗi nhập dữ liệu.

Tiến hành kiểm tra xem liệu giá trị thời gian trễ hạn nghiêm trọng trong 2 năm và thu nhập hàng tháng có bằng nhau hay không.

Số người có giá trị thu nhập hàng tháng bằng với giá trị thời gian trễ hạn nghiêm trọng trong 2 năm với quantile trên feature tỷ lệ nợ lớn hơn hoặc bằng 95% là 331 mẫu.

Do đó, nghi ngờ là đúng và có 331 trong số 379 hàng có thu nhập hàng tháng bằng với vi phạm nghiêm trọng trong 2 năm. Do đó, sẽ loại bỏ 331 giá trị ngoại lệ này khỏi phân tích của mình vì các giá trị hiện tại của chúng không hữu ích cho mô hình dự đoán. Dữ liệu sau khi loại bỏ 331 mẫu còn 149669 mẫu.

Tiếp theo chúng tôi sẽ xử lí feature sử dụng quay vòng của hạn mức tín dụng không có bảo đảm, trường này về cơ bản thể hiện tỷ lệ số tiền nợ theo hạn mức tín dụng của khách hàng. Tỷ lệ cao hơn 1 được coi là một sai số nghiêm trọng. Tỷ lệ bằng 10 về mặt chức năng cũng có vẻ khả thi, hãy xem có bao nhiêu khách hàng trong số này có sử dụng quay vòng của hạn mức tín dụng không có bảo đảm lớn hơn 10.

Bảng 3- 8: Thông tin về khách hàng có sử dụng quay vòng của hạn mức tín dụng không có bảo đảm lớn hơn 10 với 5 feature.

	Thời gian trễ hạn nghiêm trọng trong 2 năm	***Sử dụng quay vòng của hạn mức tín dụng không có bảo đảm***	***Tuổi***	***Số thời gian 30-59 ngày quá hạn không tệ hơn***	***Tỷ lệ nợ***
số lượng	241.00	241.00	241.00	241.00	173.00
giá trị trung bình	0.07	3564.02	50.63	0.18	8467.67
độ lệch chuẩn	0.25	5123.80	14.56	0.57	6564.06
min	0.00	11.38	24.00	0.00	0.00
25%	0.00	941.00	39.00	0.00	4500.00
50%	0.00	2012.00	48.00	0.00	7000.00
75%	0.00	4116.00	62.00	0.00	10091.00
max	1.00	50708.00	87.00	3.00	44472.00

Bảng 3- 9: Thông tin về khách hàng có sử dụng quay vòng của hạn mức tín dụng không có bảo đảm lớn hơn 10 với 6 feature.

	Thu nhập hàng tháng	***Số lượng hạn mức tín dụng mở và các***	***Số lần trễ 90 ngày***	***Đánh số các khoản vay hoặc***	***Số thời gian 60-89 ngày quá hạn***	***Số người phụ thuộc***

		khoản cho vay		*dòng bất động sản*	*không tệ hơn*	
số lượng	173.00	241.00	241.00	241.00	241.00	228.00
giá trị trung bình	8467.67	5.76	0.07	1.18	0.08	0.68
độ lệch chuẩn	6564.06	3.11	0.57	1.06	0.55	1.03
min	0.00	1.00	0.00	0.00	0.00	0.00
25%	4500.00	4.00	0.00	0.00	0.00	0.00
50%	7000.00	5.00	0.00	1.00	0.00	0.00
75%	10091.00	7.00	0.00	2.00	0.00	1.00
max	44472.00	21.00	8.00	9.00	7.00	4.00

Sau đó chúng tôi kiểm tra xem những giá trị của feature sử dụng quay vòng của hạn mức tín dụng không có bảo đảm nào làm ảnh hưởng ít hay nhiều đến feature thời gian trễ hạn nghiêm trọng trong 2 năm bằng cách tính giá tri trung bình của feature thời gian trễ hạn nghiêm trọng trong 2 năm theo giá trị của feature sử dụng quay vòng của hạn mức tín dụng không có bảo đảm. Với Sử dụng quay vòng của hạn mức tín dụng không có bảo đảm >= i

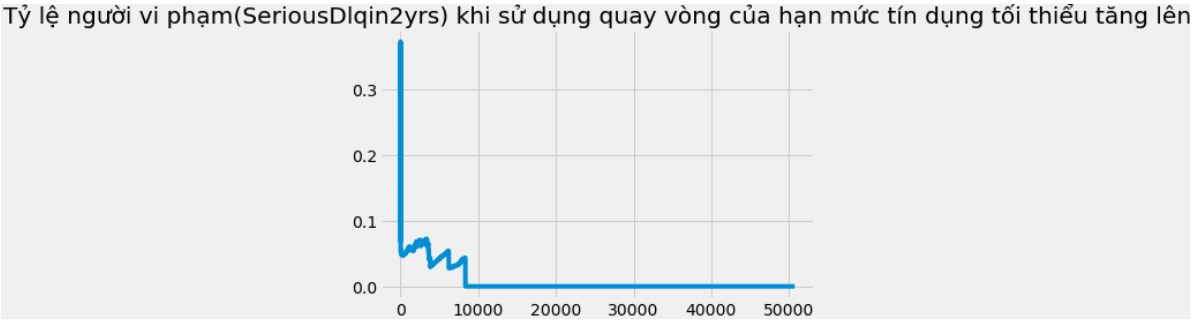
Và có kết quả là:

Bảng 3- 10: Ảnh hưởng của feature sử dụng quay vòng của hạn mức tín dụng không có bảo đảm lên feature thời gian trễ hạn nghiêm trọng trong 2 năm

	GIÁ TRỊ TRUNG BÌNH CỦA FEATURE THỜI GIAN TRỄ HẠN NGHIÊM TRỌNG TRONG 2 NĂM
0	0.0669677755580648
1	0.3722255548890222
2	0.14555256064690028
3	0.09931506849315068
4	0.08679245283018867
5	0.07874015748031496
6	0.07692307692307693
7	0.0778688524590164
8	0.07407407407407407
9	0.07053941908713693
10	0.07053941908713693
11	0.07053941908713693
12	0.06666666666666667
13	0.058823529411764705
14	0.058823529411764705

15	0.05531914893617021
16	0.05531914893617021
17	0.05531914893617021
18	0.05531914893617021
19	0.05555555555555555
20	0.05555555555555555
…	≈0.05
994	0.057803468208092484
995	0.057803468208092484
996	0.057803468208092484
997	0.057803468208092484
998	0.057803468208092484
999	0.057803468208092484
…	…

Chúng tôi thấy từ giá trị 13 trở đi giá trị của feature sử dụng quay vòng của hạn mức tín dụng không có bảo đảm không làm ảnh hưởng nhiều đến giá trị của feature thời gian trễ hạn nghiêm trọng trong 2 năm.



Hình 3- 2: Tỷ lệ người vi phạm(Thời gian trễ hạn nghiêm trọng trong 2 năm) khi sử dụng quay vòng của hạn mức tín dụng tối thiểu tăng lên

Tỷ lệ người vi phạm(thời gian trễ hạn nghiêm trọng trong 2 năm) có tổng số tiền sở hữu không vượt quá tổng hạn mức tín dụng: 0.06003334426587952. Nghĩa là sử dụng quay vòng của hạn mức tín dụng không có bảo đảm trong khoảng từ 0 đến 1.

Tỷ lệ người vi phạm(thời gian trễ hạn nghiêm trọng trong 2 năm) có tổng số tiền sở hữu không vượt quá hoặc bằng 13 lần tổng hạn mức tín dụng: 0.0669807469668275.

Bảng 3- 11: Miêu tả về khách hàng có sử dụng quay vòng của hạn mức tín dụng không có bảo đảm lớn hơn 13 với 5 feature.

	Thời gian trễ hạn nghiêm trọng trong 2 năm	***Sử dụng quay vòng của hạn mức tín dụng không có bảo đảm***	***Tuổi***	***Số thời gian 30-59 ngày quá hạn không tệ hơn***	***Tỷ lệ Nợ***

số lượng	238.00	238.00	238.00	238.00	238.00
giá trị trung bình	0.05	3608.79	50.63	0.16	579.11
độ lệch chuẩn	0.23	5140.42	14.61	0.52	1782.96
min	0.00	14.00	24.00	0.00	0.00
25%	0.00	951.00	39.00	0.00	0.21
50%	0.00	2023.00	48.00	0.00	0.39
75%	0.00	4128.75	62.00	0.00	82.25
max	1.00	50708.00	87.00	3.00	21395.00

Bảng 3- 12: Miêu tả về khách hàng có Sử dụng quay vòng của hạn mức tín dụng không có bảo đảm lớn hơn 13 với 6 feature.

	Thu nhập hàng tháng	***Số lượng hạn mức tín dụng mở và các khoản cho vay***	***Số lần trễ 90 ngày***	***Đánh số các khoản vay hoặc dòng bất động sản***	***Số thời gian 60-89 ngày quá hạn không tệ hơn***	***Số người phụ thuộc***
số lượng	170.00	238.00	238.00	238.00	238.00	225.00
giá trị trung bình	8520.04	5.68	0.07	1.18	0.07	0.69
độ lệch chuẩn	6606.88	3.03	0.58	1.06	0.54	1.04
min	0.00	1.00	0.00	0.00	0.00	0.00
25%	4608.00	4.00	0.00	0.00	0.00	0.00
50%	7000.00	5.00	0.00	1.00	0.00	0.00
75%	10178.75	7.00	0.00	2.00	0.00	1.00
max	44472.00	21.00	8.00	9.00	7.00	4.00

Sau đó chúng tôi đã loại bỏ những khách hàng có Sử dụng quay vòng của hạn mức tín dụng không có bảo đảm lớn hơn 13. Và dữ liệu còn lại 149431 mẫu

Xử Lí Missing và Null data:

Tiếp theo chúng tôi sẽ xử lí dữ liệu bị thiếu của trường thu nhập hàng tháng và trường số người phụ thuộc. Vì thu nhập hàng tháng là một giá trị số nguyên, chúng tôi sẽ thay thế các giá trị null bằng các giá trị trung vị (median). Số người phụ thuộc có thể được mô tả như một biến phân loại, do đó nếu khách hàng có NaN cho số người phụ thuộc, điều đó có nghĩa là họ không có bất kỳ người phụ thuộc nào. Do đó, chúng tôi điền chúng bằng các số 0.

Bảng 3- 13: Tỷ lên phần trăm dữu liệu bị mất của 2 feature trên file train

	Tỷ lệ phần trăm dữ liệu NaN
Thu nhập hàng tháng	19.850633

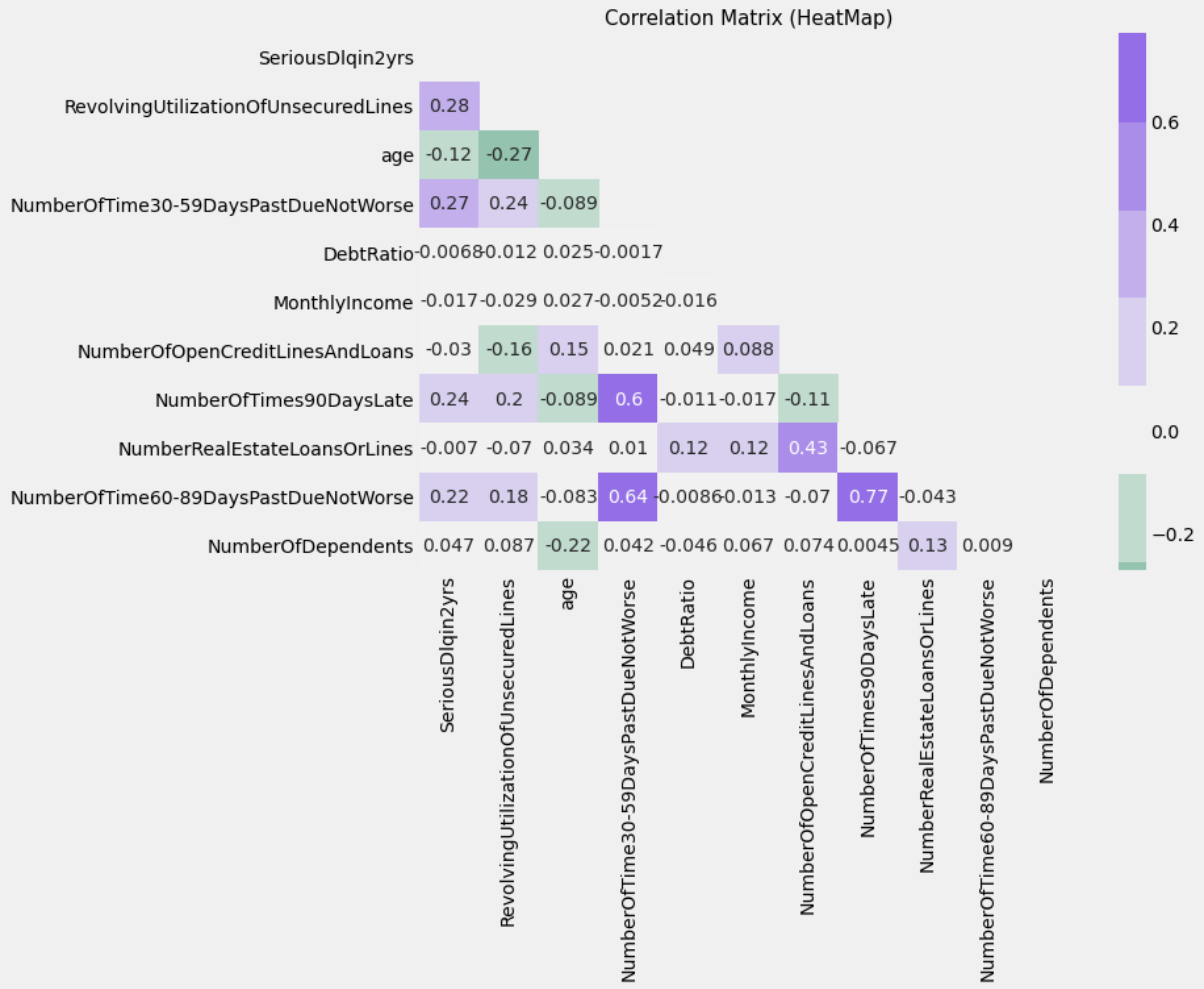
Số người phụ thuộc	2.617264

Bảng 3- 14: Tỷ lệ phần trăm dữ liệu bị thiếu trên tập test.

	Tỷ lệ phần trăm dữ liệu NaN
Thời gian trễ hạn nghiêm trọng trong 2 năm	100.000000
Thu nhập hàng tháng	19.805326
Số người phụ thuộc	2.587116

Và sử lí giá trị bị thiếu của tập test tương tự tập train. Không quan tâm đến feature thời gian trễ hạn nghiêm trọng trong 2 năm vì trường đó trong tập test chưa có dữ liệu.

Sau đó chúng tôi sẽ kiểm tra mối tương quan giữa các feature. Trên Hình 3-5 là ma trận tương quan (CORRELATION MATRIX) sử dụng công thức corr có ở phần 2.4.5 bằng lệnh corr() trong python.



Hình 3- 3: Ma trận ương quan của các feature trong tập train

Từ heatmap tương quan ở trên, chúng ta có thể thấy các giá trị tương quan nhất với thời gian trễ hạn nghiêm trọng trong 2 năm là số thời gian 30-59 ngày quá hạn không tệ hơn, số thời gian 60-89 ngày quá hạn không tệ hơn và số lần trễ 90 ngày, sử dụng quay vòng của hạn mức tín dụng không có bảo đảm.

Feature Engineering

Một phương pháp của deep learning, kết hợp tập train và tập test để thêm một số feature trên toàn tập dữ liệu và tiến hành phân tích sâu hơn. Sẽ chia chúng sau trước khi kiểm tra mô hình. Thêm vào một số feature:

- Thu nhập hàng tháng trên mỗi người (MonthlyIncomePerPerson): Thu nhập hàng tháng chia cho số người phụ thuộc
- Nợ hàng tháng (MonthlyDebt): Thu nhập hàng tháng nhân với tỷ lệ nợ
- Đã nghỉ hưu(isRetired): Người có thu nhập hàng tháng bằng 0 và tuổi lớn hơn 65
- RevolvingLines: Sự khác biệt giữa Số lượng Dòng Tín dụng Mở và Khoản cho vay (Số lượng hạn mức tín dụng mở và các khoản cho vay) và số Dòng Bất động sản và Khoản cho vay (Đánh số các khoản vay hoặc dòng bất động sản)
- hasRevolvingLines: Nếu RevolvingLines tồn tại thì bằng 1 nếu không thì bằng 0
- Có nhiều bất động sản (hasMultipleRealEstates): Nếu Số lượng Bất động sản lớn hơn hoặc bằng 2 thì giá trị bằng 1, nếu không thì giá trị bằng 0.
- Thu nhập chia cho phần nghìn (incomeDivByThousand): Thu nhập hàng tháng chia cho 1000. Có thể có nhiều khả năng là gian lận hoặc nó có thể là dấu hiệu người đó là trong một công việc mới và chưa được tăng lương theo phần trăm. Cả hai nhóm đều báo hiệu nguy cơ cao hơn.

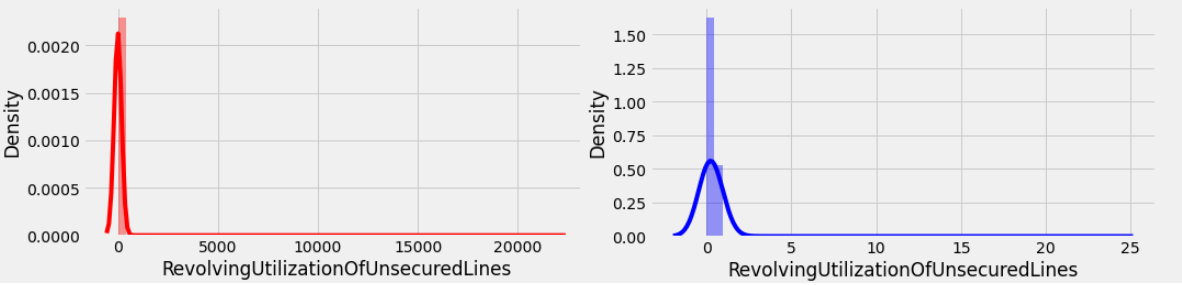
Và sau khi gộp tập train và tập test, rồi thêm feature thì chúng tôi kiểm tra final data thấy nó không có dữ liệu bị thiếu.

Bây giờ chúng tôi đã thêm các thuộc tính mới vào tập dữ liệu của mình. Tiếp theo, chúng tôi sẽ thực hiện kiểm tra độ lệch trên dữ liệu của mình bằng cách phân tích sự phân bố của các cột riêng lẻ và thực hiện chuyển đổi box cox để giảm độ lệch. Từ các biểu đồ phân phối ở dưới đây, chúng ta có thể thấy rằng phần lớn dữ liệu của chúng ta bị lệch theo một trong hai hướng. Chỉ có thể thấy tuổi gần với phân phối chuẩn. Hãy kiểm tra các giá trị độ lệch của mỗi cột.

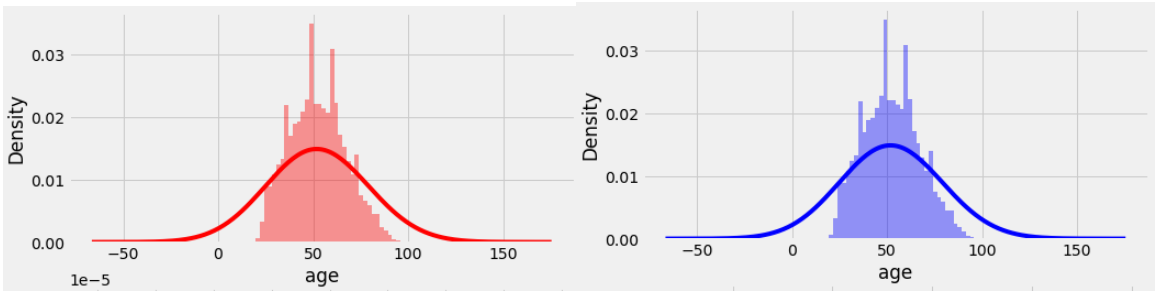
Độ lệch rất cao đối với tất cả các cột. Chúng tôi sẽ áp dụng Biến đổi Box Cox với λ = 0,15 để giảm độ lệch này.

Bảng 3- 15: Độ lệch của các feature

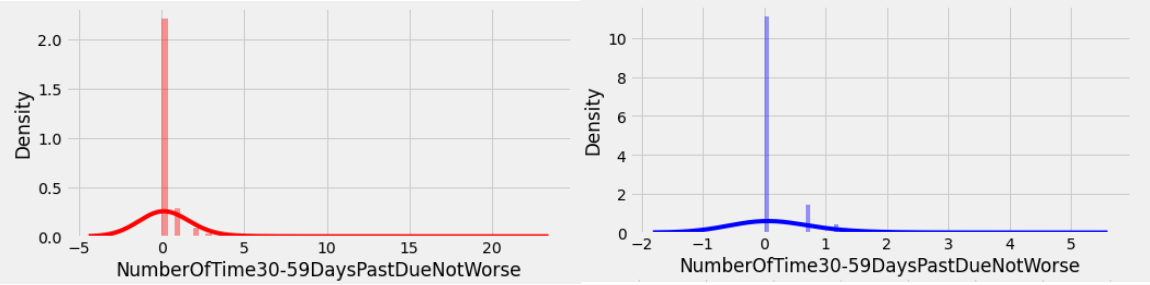
Độ lệch	***Trước khi chỉnh***	***Sau khi chỉnh***
Thu nhập hàng tháng	218.270205	-2.152376
Thu nhập chia cho phần nghìn	218.270205	0.708168
Thu nhập hàng tháng trên mỗi người	206.221804	-1.558107
Nợ hàng tháng	98.604981	1.817649
Tỷ lệ nợ	92.819627	1.958314
Sử dụng quay vòng của hạn mức tín dụng không có bảo đảm	91.721780	23.234640
Số lần trễ 90 ngày	15.097509	6.787000
Số thời gian 60-89 ngày quá hạn không tệ hơn	13.509677	6.602180
Số thời gian 30-59 ngày quá hạn không tệ hơn	9.773995	3.212010
Đánh số các khoản vay hoặc dòng bất động sản	3.217055	3.217055
Số người phụ thuộc	1.829982	0.947591
isRetired	1.564456	1.564456
RevolvingLines	1.364633	1.364633
Số lượng hạn mức tín dụng mở và các khoản cho vay	1.219429	1.219429
hasMutilpleRealEstates	1.008475	1.008475
hasRevolvingLines	-8.106007	-8.106007



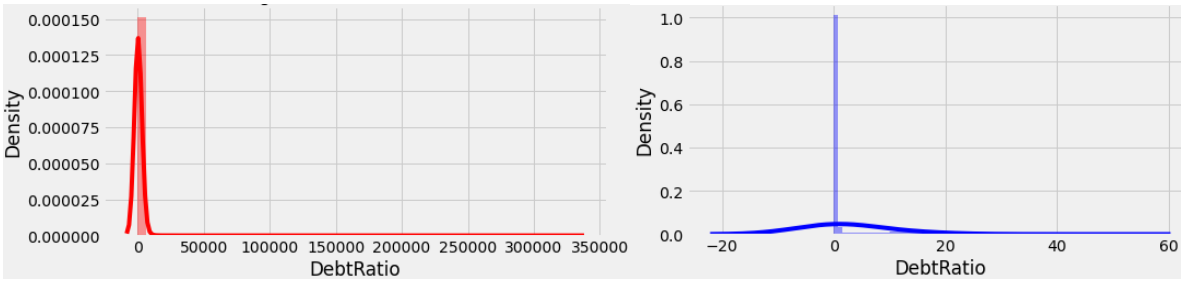
Hình 3- 4:Phân bố feature sử dụng quay vòng của hạn mức tín dụng không có bảo đảm khi chưa chỉnh độ lệch và đã chỉnh độ lệch



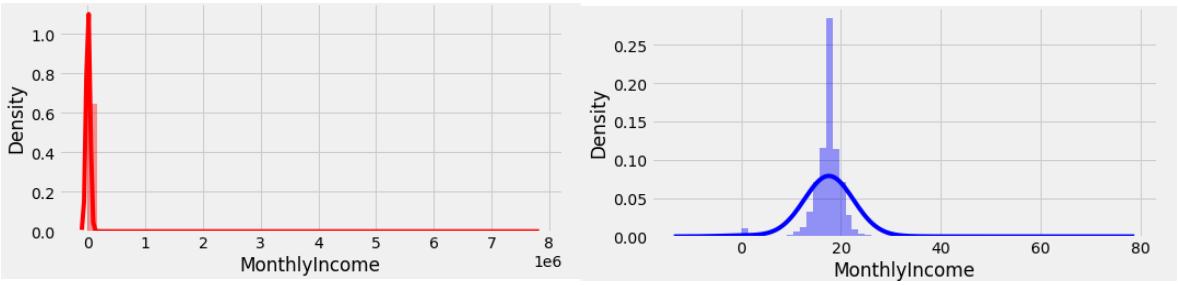
Hình 3- 5:Phân bố feature tuổi khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



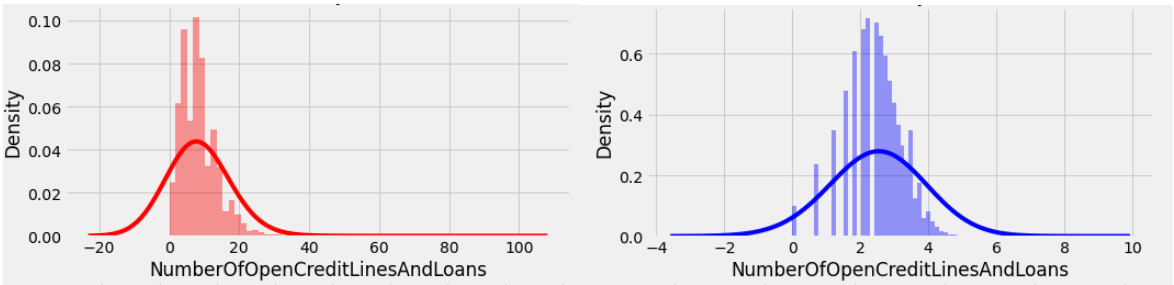
Hình 3- 6:Phân bố feature số thời gian 30-59 ngày quá hạn không tệ hơn khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



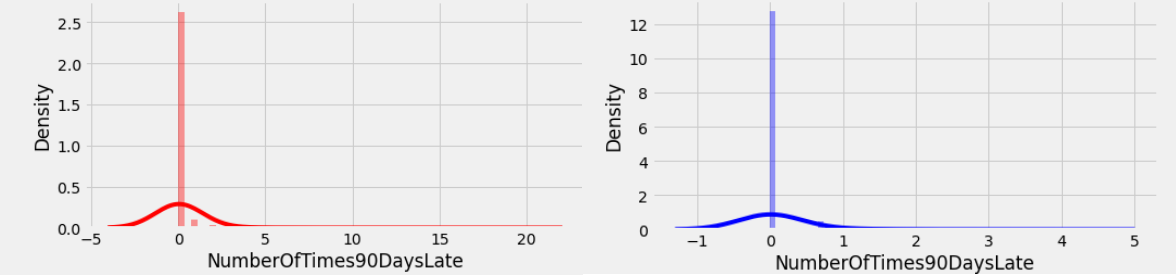
Hình 3- 7:Phân bố feature tỷ lệ nợ khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



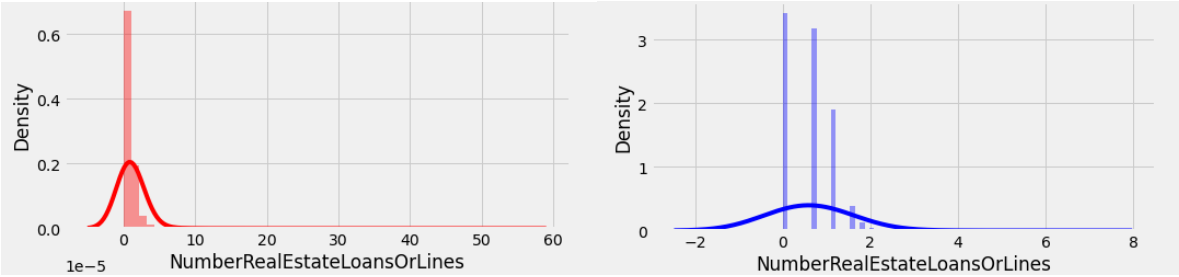
Hình 3- 8:Phân bố feature thu nhập hàng tháng khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



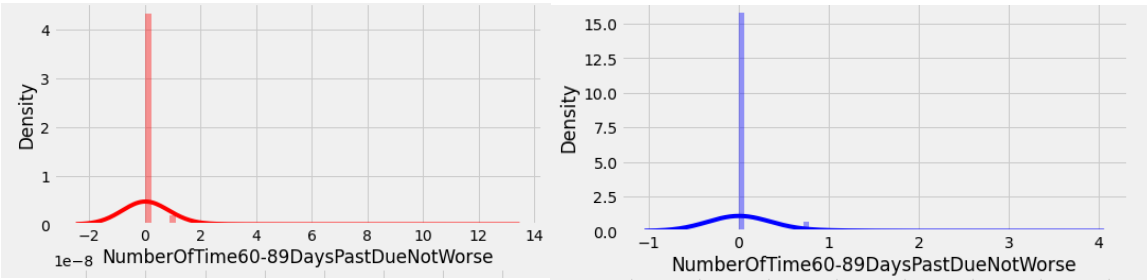
Hình 3- 9:Phân bố feature số lượng hạn mức tín dụng mở và các khoản cho vay khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



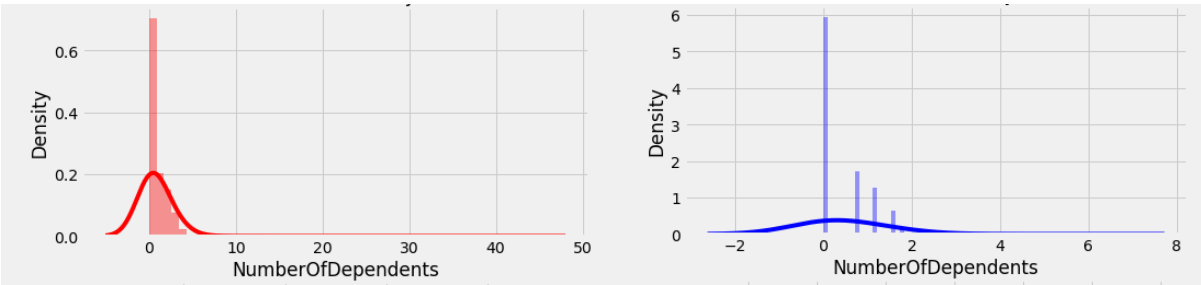
Hình 3- 10:Phân bố feature số lần trễ 90 ngày khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



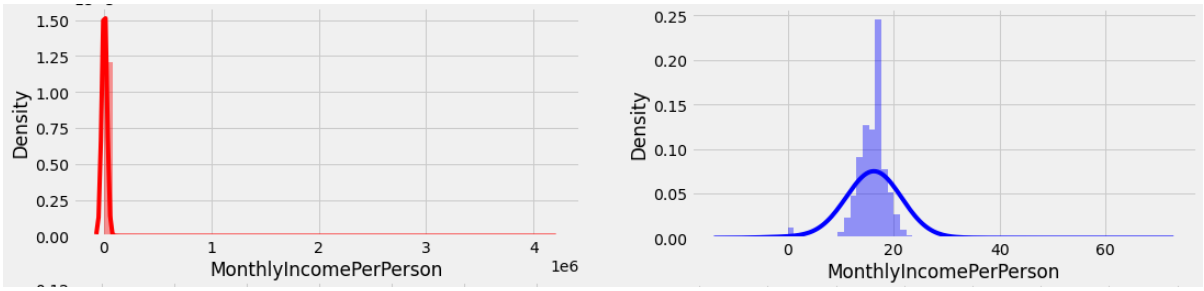
Hình 3- 11:Phân bố feature đánh số các khoản vay hoặc dòng bất động sản khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



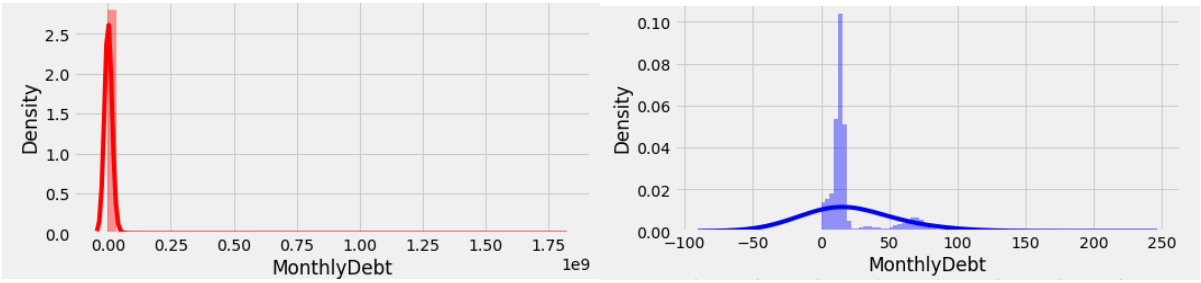
Hình 3- 12:Phân bố feature số thời gian 60-89 ngày quá hạn không tệ hơn khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



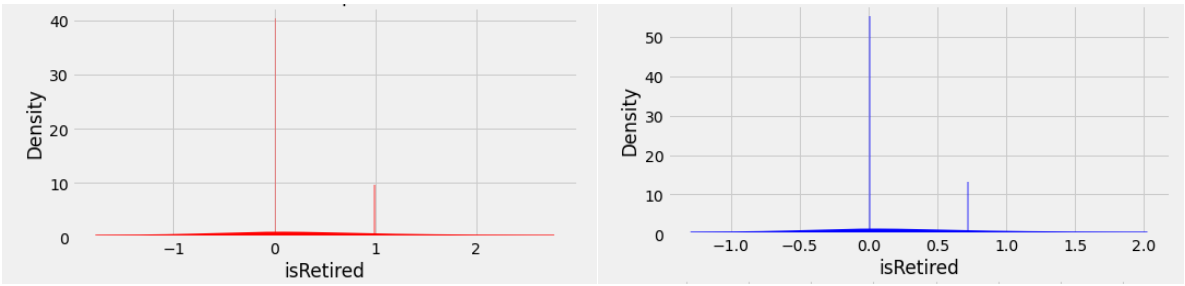
Hình 3- 13:Phân bố feature số người phụ thuộc khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



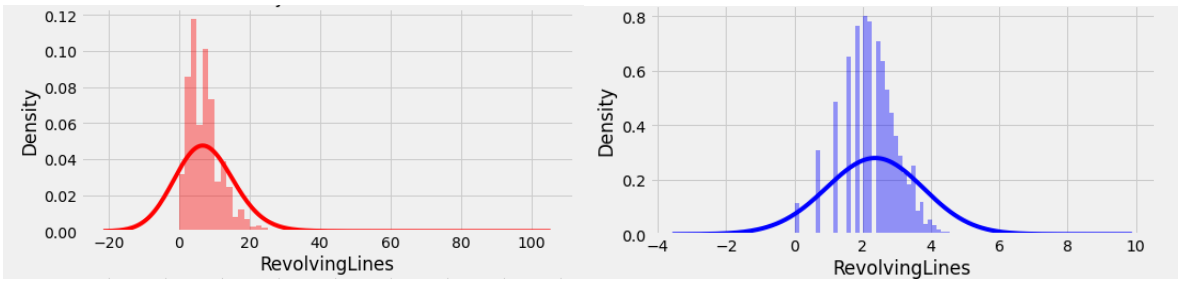
Hình 3- 14:Phân bố feature thu nhập hàng tháng trên mỗi người khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



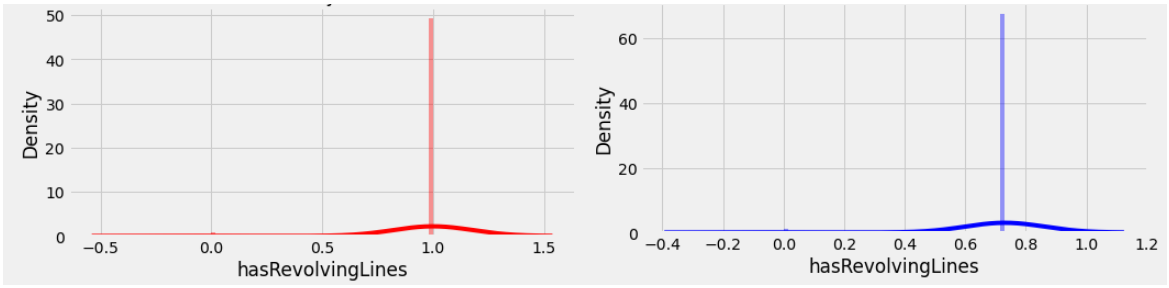
Hình 3- 15:Phân bố feature nợ hàng tháng khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



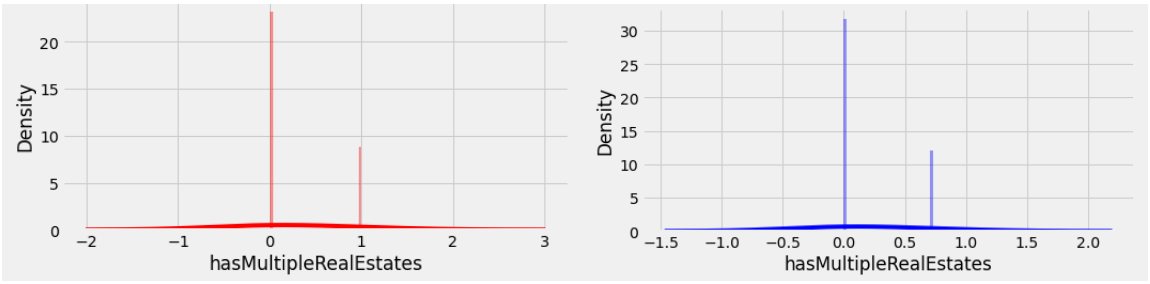
Hình 3- 16:Phân bố feature IsRetired khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



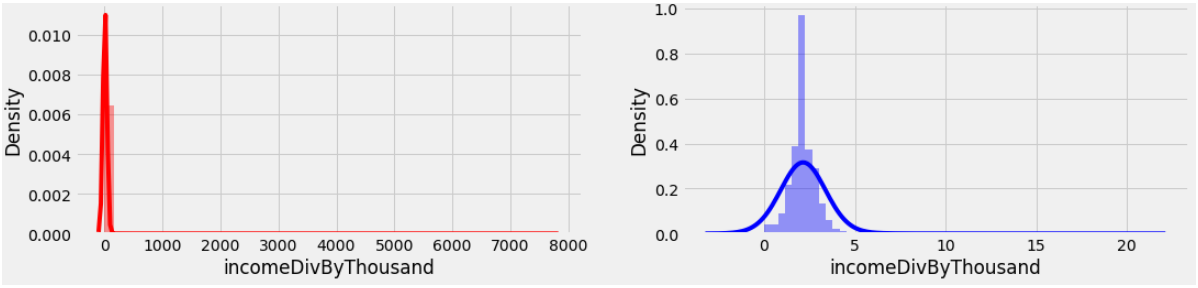
Hình 3- 17:Phân bố feature RevolvingLines khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



Hình 3- 18:Phân bố feature HasRevolvin Lines khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



Hình 3- 19:Phân bố feature HasMultipleRealEstates khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch



Hình 3- 20:Phân bố feature thu nhập chia cho phần nghìn khi chưa chỉnh độ lệch và sau khi chỉnh độ lệch

Độ lệch đã giảm ở quy mô cao hơn nhiều khi mà Chuyển đổi Box Cox được áp dụng.

3.2.2. Các Mô Hình

Sẽ chia train và val thành tỷ lệ 70-30.

Chia tập final ra thành trainDF với 149431 mẫu và 17 feature và testDF với 101503 mẫu với 17 feature.

Mô hình Decision tree regression and classfition

Lấy từ tập dữ liệu final train bỏ một số feature không quan trọng sau khi sử dụng mô hình LGBM tách thành xtrain, xtest, ytrain, ytest và dùng SMOTE để cân bằng dữ liệu để xử dụng mô hình DTR.

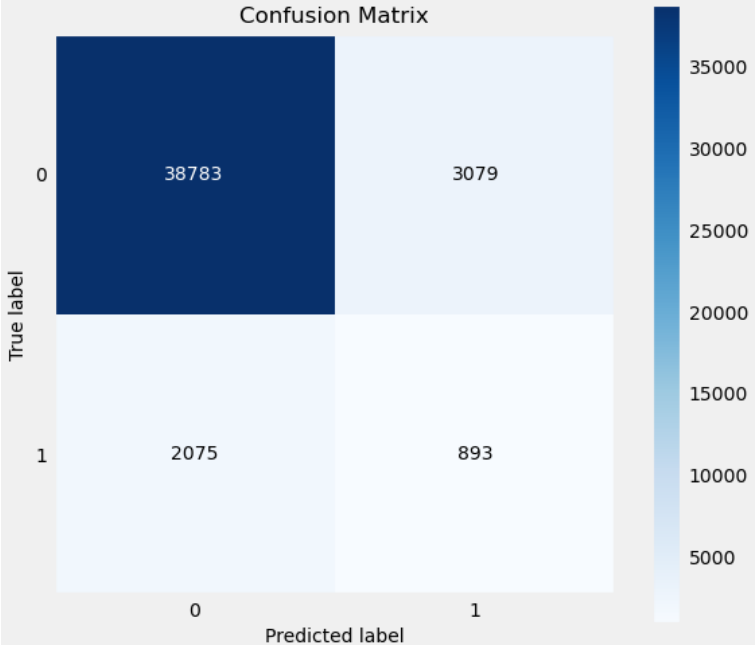
Phương pháp phổ biến để lấy mẫu các lớp thiểu số Kỹ thuật Oversampling cho nhóm thiểu số tổng hợp (Synthetic Minority Oversampling Technique-SMOTE), việc triển khai cơ bản của SMOTE sẽ không tạo ra bất kỳ sự phân biệt nào giữa các mẫu dễ và khó được phân loại bằng cách sử dụng quy tắc láng giềng gần nhất (nearest neighbors rule).

Dùng mô hình DTC vào mô hình và dự đoán đầu ra qua xtest bằng lệnh predict()

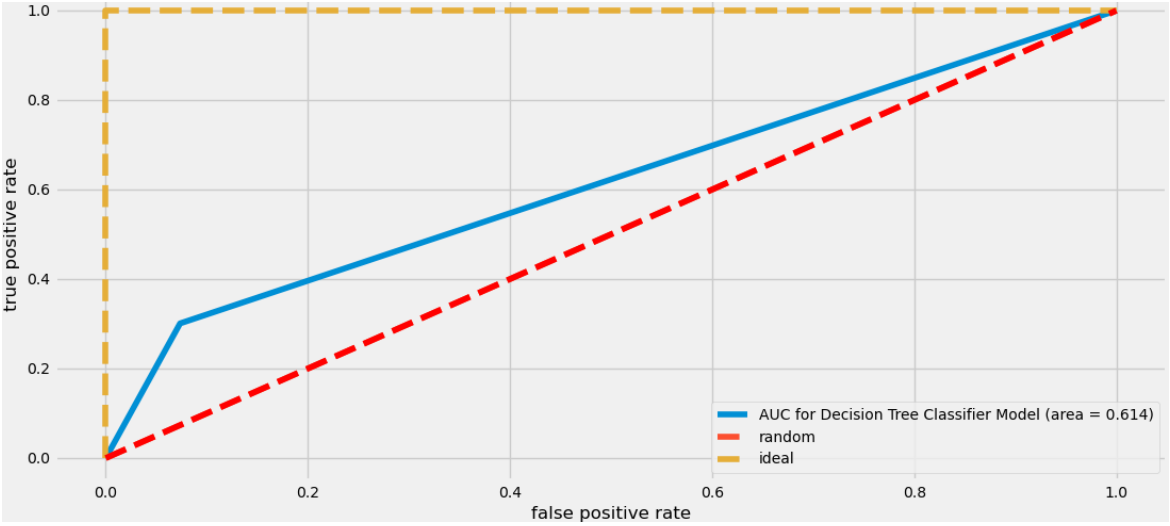
Bảng 3- 16:In ra precision recall f1-score và accuray của mô hình DTC

	precision	***recall***	***f1-score***	***support***
0	0.95	0.93	0.94	41862
1	0.22	0.29	0.25	2968
accuracy			0.88	44830
macro avg	0.59	0.61	0.60	44830
weighted avg	0.90	0.88	0.89	44830

Mô hình DTC có f1-score thấp và cả recall và precision đều thấp vì vậy mô hình này không dự đoán được nhiều label 1 thật. Sau đây là ma trận nhầm lẫn để chỉ rõ hơn về số lượng dự đoán ở các label:



Hình 3- 21: Ma trận nhầm lẫn trong mô hình DTC



Hình 3- 22: Đường cong ROC của mô hình DTC

Cho thấy tỷ lẻ TP với FP và AUC bằng 0.614, tức là có 61,4% nhãn 1 được dự đoán đúng.

Bảng 3- 17: DecisionTreeClassifierMetrics

Model	***MSE***	***RMSE***	***MAE***	***MSLE***	***RMSLE***	***Accuracy Train***	***Accuracy Test***	***F-Beta Score (β=2)***
Decision Tree Classsifier	11.52	3.39	11.52	5.54	2.35	99.96	88.49	27.57

Mô Hình LGBM

Sử dụng mô hình LightGBM với tham số là:

Bảng 3- 18:Tham số mô hình LightGBM

Tham số	***Thông số***
max_depth	2, 3, 4, 5
learning_rate	0.05, 0.1, 0.125, 0.15
colsample_bytree	0.2, 0.4, 0.6, 0.8, 1
n_estimators	400, 500, 600, 700, 800, 900
min_split_gain	0.15, 0.20, 0.25, 0.3, 0.35
min_child_weight	0.6, 0.7, 0.8, 0.9, 1
min_child_weight	6, 7, 8, 9, 10
scale_pos_weight	10, 15, 20
min_data_in_leaf	100, 200, 300, 400, 500, 600, 700, 800, 900
num_leaves	20, 30, 40, 50, 60, 70, 80, 90, 100.

Sau đó chúng tôi hiệu chỉnh tham số mô hình LGBM bằng RandomizedSearchCV với danh sách các tham số để thử bên trên, xác định chiến lược phân tách xác thực

chéo (cv) là 5 và trạng thái tạo số ngẫu nhiên giả được sử dụng để lấy mẫu đồng nhất ngẫu nhiên từ danh sách các giá trị có thể (random_state) là 2020.

Sau khi chạy xong công cụ ước tính ta lấy được tham số mô hình tốt nhất là:

Bảng 3- 19: Tham số tối ưu của mô hình LightGBM

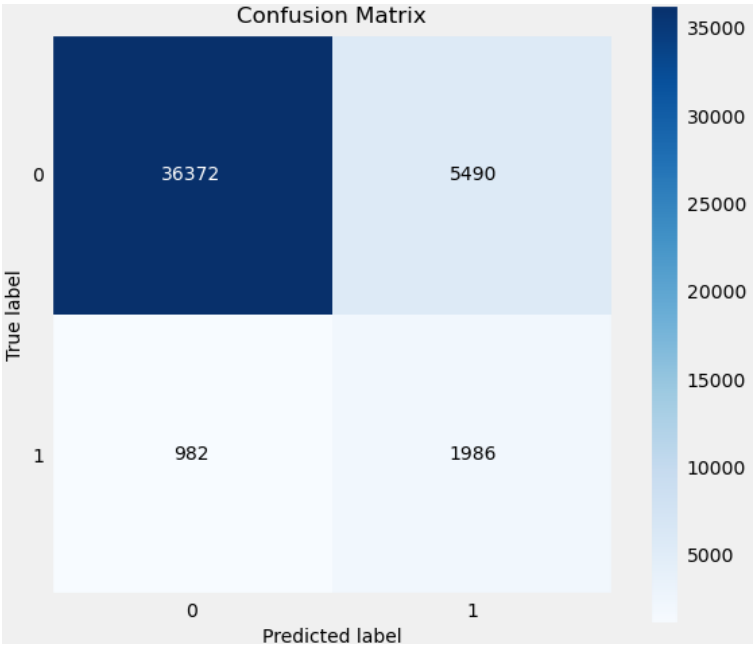
Tham số	Thông số
colsample_bytree	***0.4***
importance_type	***'gain'***
max_depth	***5***
min_child_weight	***6***
min_data_in_leaf	***600***
min_split_gain	***0.25***
n_estimators	***900***
num_leaves	***50***
objective	***'binary'***
random_state	***2020***
scale_pos_weight	***10***
subsample	***0.9***

Sau đó chúng tôi lưu lại tham số tốt nhất để dự đoán bằng lệnh predict_proba() trong LGBM.

Bảng 3- 20: Dự đoán feature Thời gian trễ hạn nghiêm trọng trong 2 năm cho xTest và in ra precision recall f1-score và accuray của mô hình LGBM

	precision	***recall***	***f1-score***	***support***
0	0.97	0.87	0.92	41862
1	0.27	0.67	0.38	2968
accuracy			0.86	44830
macro avg	0.62	0.77	0.65	44830
weighted avg	0.93	0.86	0.88	44830

Độ chính xác accuracy bằng 0.86 khá cao nhưng trong tập dữ liệu mất cân bằng chúng ta quan tâm đến f1-score hay recall hay precision nếu TP hiếm hay label 1 hiếm. Precision khác thấp điều đó cho thấy các mẫu được dự đoán là 1 không phải là label 1 thật và recall bằng 0.67 khá cao cho thấy dự đoán được nhiều label 1. Sau đây là ma trận nhầm lẫn sẽ chỉ rõ điều đó hơn:



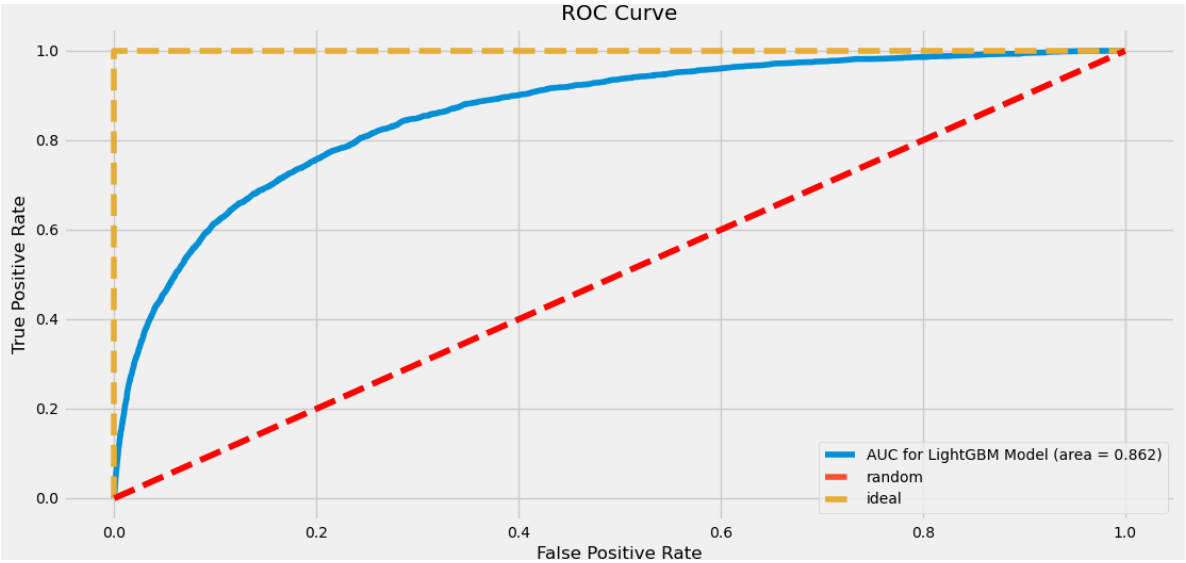
Hình 3- 23: ma trận nhầm lẫn của LGBM

Số liệu của mô hình LGBM dùng để so sánh xem mô hình nào là tốt hơn, và kiểm tra xem mô hình này tốt đến đâu:

Bảng 3- 21: LGBM Metrics

Mô Hình	***MSE***	***RMSE***	***MAE***	***MSLE***	***RMSLE***	***Accuracy Train***	***Accuracy Test***	***F-Beta Score (β=2)***
LightGBM	14.44	3.8	14.4	6.94	2.63	86.55	85.56	51.32

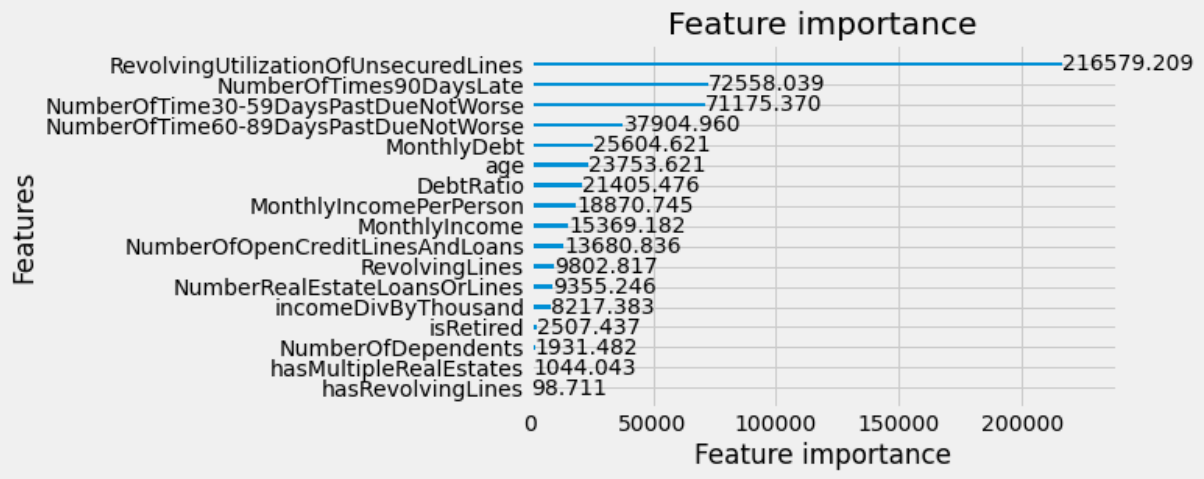
Và đường cong ROC về AUC của mô hình LGBM:



Hình 3- 24: ROC của mô hình LGBM

Cho thấy tỷ lẻ TP với FP và AUC bằng 0.862, tức là có 86,2% nhãn 1 được dự đoán đúng.

Các feature quan trọng:



Hình 3- 25: Các feature quan trọng của mô hình LGBM

Mô hình Logistic Regression

Lấy từ tập dữ liệu final train bỏ một số feature không quan trọng sau khi sử dụng mô hình LGBM tách thành x_train, x_test, y_train, y_test và dùng StandardScaler để chuẩn hóa dữ liệu dùng SMOTE để cân bằng dư liệu để xử dụng mô hình LR

StandardScaler() chuẩn hóa các thuộc tính bằng cách loại bỏ giá trị trung bình và chia tỷ lệ thành phương sai đơn vị. Điểm số tiêu chuẩn của một x mẫu được tính như sau:

$$z = \frac{(x - \bar{x})}{std} \tag{42}$$

trong đó $\bar{x}$ là giá trị trung bình của các mẫu train hoặc bằng 0 nếu with_mean = False và std là độ lệch chuẩn của các mẫu train.

Dùng mô hình LR với các tham số:

Tham số C làm tham số hiệu chỉnh của chúng tôi. Với $C = \frac{1}{\lambda}$. Lambda (λ) kiểm soát sự cân bằng giữa việc cho phép mô hình tăng độ phức tạp tùy thích với việc cố gắng giữ cho nó đơn giản.Ví dụ, nếu λ rất thấp hoặc 0, mô hình sẽ có đủ mạnh để tăng độ phức tạp của nó (overfit) bằng cách gán các giá trị lớn cho các trọng số cho mỗi tham số. Mặt khác, nếu chúng ta tăng giá trị của λ, mô hình sẽ có xu hướng không phù hợp, vì mô hình sẽ trở nên quá đơn giản. Tham số C sẽ hoạt động theo chiều ngược lại. Đối với các giá trị nhỏ của C, chúng tôi tăng cường độ hiệu chỉnh(chính quy hóa), điều này sẽ tạo ra các mô hình đơn giản phù hợp với dữ liệu. Đối với các giá trị lớn

của C, chúng tôi giảm sức mạnh của hiệu chỉnh C, điều này có nghĩa là mô hình được phép tăng độ phức tạp của nó.

Bảng 3- 22: Tham số mô hình Logistic Regression

Tham số	***Thông số***
C	0.001, 0.01, 0.1, 1.0, 10, 100
Penalty	L1, L2
Điểm đánh giá	Recall
cv	10
n_jobs	-1

Điều chỉnh các siêu tham số của công cụ ước tính bằng GridSearchCV. Tìm kiếm hoàn chỉnh trên các giá trị tham số được chỉ định cho một công cụ ước tính. Các thông số của ước lượng sử dụng để áp dụng các phương pháp này được tối ưu hóa bằng lưới tìm kiếm qua xác nhận trên một mạng lưới tham số được cho ở trên. ***Tham số tốt nhất được chỉ định bởi grid với các mô hình tuyết tính là các giá trị C bằng 1 và penalty là l2.***

Bảng 3- 23:*Dùng công cụ ước tính tốt nhất của GridSearchCV để dự đoán và xác nhận mô hình bằng cách so sánh các thông số của tập train và test*

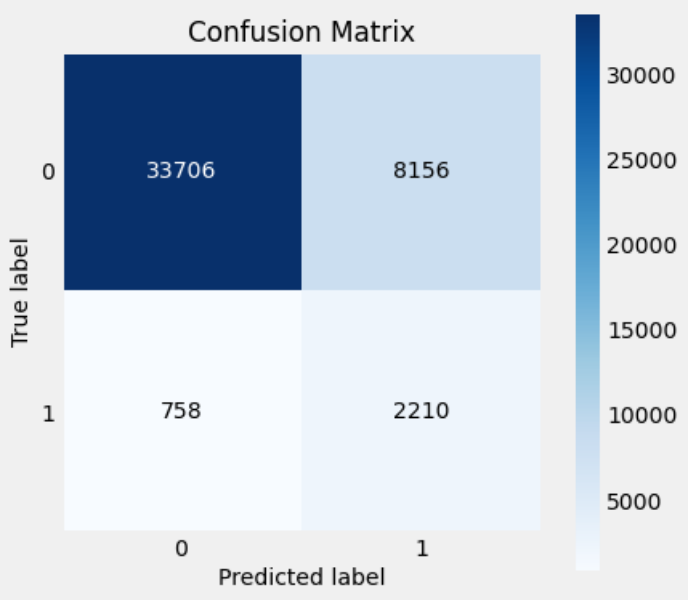
	Train	***Test***
Độ chính xác	0.8	0.801
Precision	0.215	0.213
Recall	0.743	0.745
F1_score	0.334	0.331

Bảng 3- 24: In ra precision recall f1-score và accuray của mô hình LR

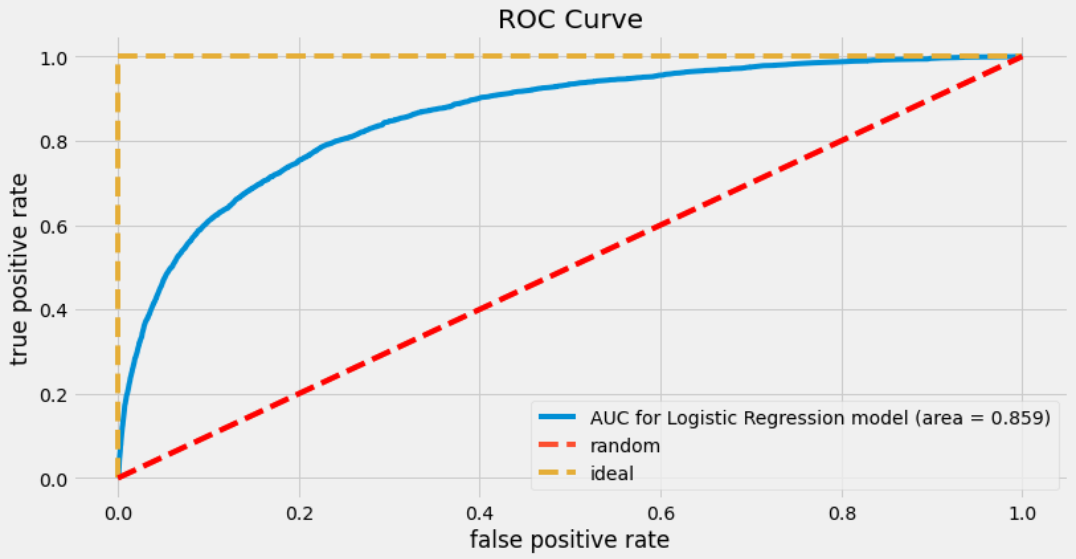
	Precision	***Recall***	***F1-score***	***Support***
0	0.98	0.81	0.88	41862
1	0.21	0.74	0.33	2968
Accuracy			0.80	44830
Macro avg	0.60	0.77	0.61	44830
Weighted avg	0.93	0.80	0.85	44830

Độ chính xác accuracy bằng 0.80 khá cao nhưng trong tập dữ liệu mất cân bằng chúng ta quan tâm đến f1-score hay recall hay precision nếu TP hiếm hay label 1 hiếm. Precision khác thấp điều đó cho thấy các mẫu được dự đoán là 1 không phải là label 1 thật và recall bằng 0.74 khá cao cho thấy dự đoán được nhiều label 1. Sau đây là ma trận nhầm lẫn sẽ chỉ rõ điều đó hơn:

Hình 3- 26: Ma trận nhầm lẫn của mô hình LR



Hình 3- 27: Đường cong ROC của mô hình LR



Cho thấy tỷ lệ TP với FP và AUC bằng 0.859, tức là có 85,9% nhãn 1 được dự đoán đúng.

Bảng 3- 25: LogisticRegressionMetrics

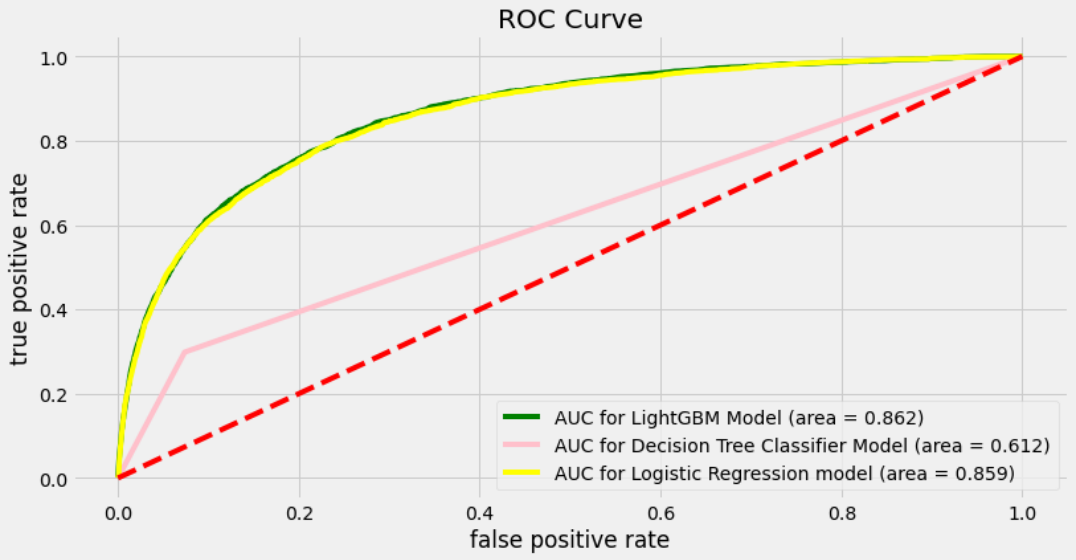
Mô Hình	***MSE***	***RMSE***	***MAE***	***MSLE***	***RMSLE***	***Accuracy Train***	***Accuracy Test***	***F-Beta Score (β=2)***
Logistic Regression	19.88	4.46	19.88	9.55	3.09	93.26	93.36	49.69

3.2.3. So sánh các mô hình

Hiệu quả về mặt thời gian của ba mô hình:

Model	*LightGBM*	*DecisionTreeClassifier*	*LogisticRegression*
Thời gian chạy (giây)	0,06	0,03	60,017

Mô hình Decision Tree có thời gian chạy nhanh nhất, thứ hai là LightGBM và Logistic Regression có thời gian chạy lâu nhất.



Hình 3- 28: Đường cong ROC của 3 mô hình LGBM, DTC và LR

Đường màu hồng là đường ROC của mô hình decision tree có recall positive, tức là điểm recall của nhãn 1 thấp đồng nghĩa với việc dự đoán lớp 1 đúng ít và với diện tích dưới đường cong ROC là AUC có kết quả là 0.614 cho thấy mô hình phân loại sai 38.6% lớp 1.

Đường màu vàng là đường ROC mô hình Logistic Regression recall positive, tức là điểm recall của nhãn 1 khá cao, mô hình này dự đoán lớp 1 đúng tương đối nhiều và với diện tích dưới đường cong ROC là AUC có kết quả là 0.859 thấy được mô hình này phân loại sai 14.1% lớp 1.

Đường màu xanh lục là đường ROC mô hình LGBM có recall positive, tức là điểm recall của nhãn 1 khá cao và cao nhất trong ba mô hình mô hình này dự đoán lớp 1 đúng tương đối nhiều và với diện tích dưới đường cong ROC là AUC có kết quả là 0.859 thấy được mô hình này phân loại sai 13.8% lớp 1.

Bảng 3- 26: Bảng so sánh metrics giữa ba mô hình LGBM, DTC và LR

Model	***LightGBM***	***DecisionTreeClassifier***	***LogisticRegression***
MSE	14.44	11.52	19.88

RMSE	3.8	3.39	4.46
MAE	14.44	11.52	19.88
MSLE	6.94	5.54	9.55
RMSLE	2.63	2.35	3.09
Accuracy Train	86.55	99.96	93.26
Accuracy Test	85.56	88.48	93.36
F-Beta Score (β=2)	51.32	25.57	49.69

Từ tính toán và kết quả trên bảng 3-26:

Thấy được MSE , MAE, MSLE của Decision Tree là thấp nhất vậy sai số giữa giá trị y thực và giá trị y dự đoán trên toàn tập là ít nhất và độ chính xác cao 99.96% mà điểm f lại thấp nhất 25.57%, điều đó cho thấy mô hình Decision Tree fit với tập train nhưng không fit với tập test do đó mô hình này bị overfit, và mô hình Decision Tree không phù hợp với bộ dữ liệu này.

Còn trong mô hình Logistic Regression MSE và MAE. MSLE tương dối thấp, nhưng khá cao so với ba mô hình còn lại, cũng có điểm f cao là 49.69%, do đó tập dữ liệu này tương đối phù hợp với mô hình Logistic Regression.

Với mô hình LGBM có MSE và MAE, MSLE ở mô hình LGBM tương đối thấp và cho số điểm f cao nhất trong ba mô hình là 51.32%, mô hình này tương đối phù hợp với bộ dữ liệu này.

KẾT LUẬN VÀ KIẾN NGHỊ

Kết luận

Chúng tôi làm việc với dữ liệu số liên tục, và đã giải quyết được vấn đề như:

- Xử lí dữ liệu ngoại lai (outlier) bằng cách thay thế và loại bỏ chúng.
- Xử lí dữ liệu bị thiếu bằng các thay thế bằng chúng bằng trung vị (median) của feature có dữ liệu bị thiếu.
- Phát hiện các dữ liệu cao bất thường, nguyên nhân là lỗi nhập dữ liệu, xử lí bằng các thay giá trị bằng giá trị gần nhất.

Và chúng tôi đã dùng một kĩ thuật là feature engineering lấy dữ liệu thô, trích xuất ra, tạo dữ liệu mới để bổ sung cho tập dữ liệu vừa xử lí, giúp tăng khả năng chính xác trong việc gán nhãn.

Trong nghiên cứu hiện tại, các mô hình học máy như Decision Tree Classifier, Logistic Regression và LightGBM cũng đã được nghiên cứu và phân tích số liệu của các mô hình cho thấy cả Logistic Regression, LightGBM và Decision Tree Classifier đều khá chính xác trong việc dự đoán phân loại chính xác cho mỗi người trong vấn đề phân loại rủi ro tín dụng của chúng tôi. Cụ thể hơn, hiệu suất tốt nhất đạt được dựa trên dữ liệu được lấy mẫu quá mức bằng phương pháp SMOTE trên tập train, như 86.55%, 99.96% và 93.26% và trên tập test 85.56%, 88.48% và 93.36% lần lượt là độ chính xác của thuật toán LightGBM, Decision Tree Classifier và Logistic Regression. Tuy nhiên, nếu chúng ta phải chọn một thuật toán học máy cho công việc và phân tích trong tương lai của mình, thì điều cần thiết là sử dụng Logistic Regression và LightGBM vì nó đã được kiểm tra là chính xác nhất tới 93.26% và 86.55% trên tập train, 93.36% và 85.56%, nếu yêu cầu về mặt thời gian thì mô hình phù hợp nhất là LightGBM vì thuật toán có thời gian chạy nhanh hơn so với Logistic Regression. Không lấy Decision Tree Classifier mặc dù mô hình này đặt được 99.96% trên tập train và tập test là 88.48%, thời gian chạy mất 0,03 giây, thời gian nhanh hơn và độ chính xác cao hơn mô hình LightGBM vì nó có điểm f thấp hơn các mô hình còn lại điều đó đồng nghĩa với việc nó dự đoán sai nhãn 1 với nhãn 0 nhiều hơn các mô hình khác, và nó chỉ dự đoán đúng nhiều trên tập train. Ngược lại mô hình LightGBM lại có lợi thế hơn trong việc dự đoán đúng các nhãn vì chỉ số f cao nhất trong các mô hình khác là 51.32%, xếp theo sau là mô hình Logistic Regression với điểm f là 49.69%. Cuối cùng, điều quan trọng là phải làm nổi bật rằng một số loại biến đổi trong tập dữ liệu gốc đã góp phần tạo ra dạng cuối cùng của tập dữ liệu cuối cùng được sử dụng để huấn luyện và thử nghiệm các mô hình. Đúng là, khi nói đến bộ dữ liệu không cân

bằng, làm việc trên dữ liệu gốc hoặc làm việc trên dữ liệu được lấy mẫu quá mức là một chủ đề gây ra một cuộc tranh luận giữa giữa các nhà nghiên cứu và nhà khoa học, và phụ thuộc vào nhiều yếu tố khác nhau. Phải nói rằng, sử dụng phương pháp SMOTE để lấy mẫu quá mức, lớp thiểu số (bất thường) có thể đạt được, nói chung, hiệu suất mô hình tốt hơn trong bài toán này. Tuy nhiên, điều này không có nghĩa là chúng tốt hơn cho mọi mục đích sử dụng trong các tác vụ học máy.

Kiến Nghị

Mặc dù học máy được coi là một công cụ hữu ích để phân tích rủi ro tín dụng và dự đoán mặc định, nhưng có một số hạn chế liên quan đến loại phân tích này. Hạn chế quan trọng nhất là chất lượng dữ liệu và cường độ dự đoán. Điều kiện tiên quyết cần thiết để xây dựng một mô hình tốt và đáng tin cậy là thu thập được khối lượng dữ liệu đại diện chất lượng cao. Xác định các đặc điểm có ảnh hưởng lớn đến việc vỡ nợ hay không là một trong những thách thức lớn nhất. Khi điều kiện kinh tế thay đổi liên tục và nhanh chóng, trong khi khách hàng mới, sản phẩm mới và xu hướng mới đang được giới thiệu, thì cần phải tính đến các tính năng, biến số và mối tương quan mới. Tất cả những hạn chế này đều thể hiện trong nghiên cứu này, vì bộ dữ liệu đã tồn tại qua lâu với số lượng đặc trưng (feature) ít và hạn chế và chỉ tiết lộ một phần nhỏ của tất cả các thông số này có thể góp phần vào việc vay hay không.

Dữ liệu cá nhân và việc sử dụng dữ liệu là một chủ đề rất nhạy cảm liên quan đến các doanh nghiệp và tổ chức tài chính. Khi thời gian thay đổi và có nhiều dữ liệu hơn, có thể thực hiện các nghiên cứu sâu hơn có tính đến các bộ dữ liệu lớn hơn và phức tạp hơn. Các tính năng mới và kỹ thuật mới có thể được sử dụng trong các dự án tương lai, vì máy học và trí tuệ nhân tạo nói chung đang phát triển nhanh chóng. Ngoài ra, các quyết định do các tổ chức tài chính đưa ra bằng cách sử dụng thuật toán học máy và kết quả của chúng có thể được sử dụng làm đầu vào cho nghiên cứu trong tương lai, dẫn đến một loại đánh giá mới dựa trên hồ sơ lịch sử cho sự đúng và sai từ các báo cáo cũ.

TÀI LIỆU THAM KHẢO

[1] "Give Me Some Credit | Kaggle." https://www.kaggle.com/c/GiveMeSomeCredit/data (accessed Jun. 27, 2021).

[2] R. Z. Li, S. L. Pang, and J. M. Xu, "Neural network credit-risk evaluation model based on back-propagation algorithm," *Proc. 2002 Int. Conf. Mach. Learn. Cybern.*, vol. 4, no. November, pp. 1702–1706, 2002, doi: 10.1109/icmlc.2002.1175325.

[3] X. Y. Hu and Y. L. Tang, "Ann-based credit risk identificaion and control for commercial banks," *Proc. 2006 Int. Conf. Mach. Learn. Cybern.*, vol. 2006, no. August, pp. 3110–3114, 2006, doi: 10.1109/ICMLC.2006.258400.

[4] Y. Demyanyk and I. Hasan, "Financial crises and bank failures: A review of prediction methods," *Omega*, vol. 38, no. 5, pp. 315–324, 2010, doi: 10.1016/j.omega.2009.09.007.

[5] J. M. Tomczak and M. Zie˛ba, "Classification Restricted Boltzmann Machine for comprehensible credit scoring model," *Expert Syst. Appl.*, vol. 42, no. 4, pp. 1789–1796, 2015, doi: 10.1016/j.eswa.2014.10.016.

[6] B. Baesens, R. Setiono, C. Mues, and J. Vanthienen, "Using neural network rule extraction and decision tables for credit-risk evaluation," *Manage. Sci.*, vol. 49, no. 3, pp. 312–329, 2003, doi: 10.1287/mnsc.49.3.312.12739.

[7] B. Huang, Q. P. Zhang, and Y. Q. Hu, "Research on credit risk management of the state-owned commercial bank," *2005 Int. Conf. Mach. Learn. Cybern. ICMLC 2005*, no. August, pp. 4038–4043, 2005, doi: 10.1109/icmlc.2005.1527644.

[8] R. Vedala and B. R. Kumar, "An application of Naive Bayes classification for credit scoring in e-lending platform," *Proc. - 2012 Int. Conf. Data Sci. Eng. ICDSE 2012*, pp. 81–84, 2012, doi: 10.1109/ICDSE.2012.6282321.

[9] O. J. Okesola, K. O. Okokpujie, A. A. Adewale, S. N. John, and O. Omoruyi, "An Improved Bank Credit Scoring Model: A Naïve Bayesian Approach," *Proc. - 2017 Int. Conf. Comput. Sci. Comput. Intell. CSCI 2017*, pp. 228–233, 2018, doi: 10.1109/CSCI.2017.36.

[10] D. J. Hand and W. E. Henley, "Statistical classification methods in consumer credit scoring: A review," *J. R. Stat. Soc. Ser. A Stat. Soc.*, vol. 160, no. 3, pp. 523–541, Sep. 1997, doi: 10.1111/j.1467-985X.1997.00078.x.

[11] A. Szwabe and P. Misiorek, *Decision trees as interpretable bank credit scoring models*, vol. 928. Springer International Publishing, 2018.

[12] A. Economics and V. Xxvi, "A Deep Neural Network (DNN) based

classification model in application to loan default prediction," *Theor. Appl. Econ.*, vol. XXVI, no. 4, pp. 75–84, 2019.

[13] M. Malik and L. C. Thomas, "Modelling credit risk of portfolio of consumer loans," *J. Oper. Res. Soc.*, vol. 61, no. 3, pp. 411–420, 2010, doi: 10.1057/jors.2009.123.

[14] R. A. McDonald, M. Sturgess, K. Smith, M. S. Hawkins, and E. X. M. Huang, "Non-linearity of scorecard log-odds," *Int. J. Forecast.*, vol. 28, no. 1, pp. 239–247, Jan. 2012, doi: 10.1016/j.ijforecast.2011.01.001.

[15] H. McNab, *Principles and practice of consumer credit risk management*. 2000.

[16] N. Sarlija, M. Bensic, and M. Zekic-Susac, "Comparison procedure of predicting the time to default in behavioural scoring," *Expert Syst. Appl.*, vol. 36, no. 5, pp. 8778–8788, Jul. 2009, doi: 10.1016/j.eswa.2008.11.042.

[17] A. C. Müller and S. Guido, *Introduction to Machine Learning with Python*. 2020.

[18] S. Bhatia, P. Sharma, R. Burman, S. Hazari, and R. Hande, "Credit Scoring using Machine Learning Techniques," *Int. J. Comput. Appl.*, vol. 161, no. 11, pp. 1–4, Mar. 2017, doi: 10.5120/ijca2017912893.

[19] K. Bijak and L. C. Thomas, "Does segmentation always improve model performance in credit scoring?," *Expert Syst. Appl.*, vol. 39, no. 3, pp. 2433–2442, 2012, doi: 10.1016/j.eswa.2011.08.093.

[20] R. Anderson, "The Credit Scoring Toolkit - Theory and Practice for Retail Credit Risk Management and Decision Automation," pp. 1–790, 2007.

[21] D. J. Hand, "Modelling consumer credit risk," *IMA J. Manag. Math.*, vol. 12, no. 2, pp. 139–155, 2001, doi: 10.1093/imaman/12.2.139.

[22] "FRB Speech, Greenspan -- Banking -- October 7, 2002." https://www.federalreserve.gov/boarddocs/speeches/2002/20021007/default.htm (accessed May 06, 2021).

[23] LYN C. THOMAS, "Consumer Credit Models: Pricing, Profit, and Portfolios," *J. Chem. Inf. Model.*, vol. 53, no. 9, pp. 1689–1699, 2013.

[24] S. H. Li, D. C. Yen, W. H. Lu, and C. Wang, "Identifying the signs of fraudulent accounts using data mining techniques," *Comput. Human Behav.*, vol. 28, no. 3, pp. 1002–1013, May 2012, doi: 10.1016/j.chb.2012.01.002.

[25] H. joo Lee and S. Cho, "Focusing on non-respondents: Response modeling with novelty detectors," *Expert Syst. Appl.*, vol. 33, no. 2, pp. 522–530, 2007, doi: 10.1016/j.eswa.2006.05.016.

[26] Y. Zhao, B. Li, X. Li, W. Liu, and S. Ren, "Customer Churn Prediction using improved one-class Support Vector Machine," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3584 LNAI, pp. 300–306, 2005, doi: 10.1007/11527503_36.

[27] J. S. Thomas, “A methodology for linking customer acquisition to customer retention,” *J. Mark. Res.*, vol. 38, no. 2, pp. 262–268, 2001, doi: 10.1509/jmkr.38.2.262.18848.

[28] S. Finlay, “Multiple classifier architectures and their application to credit risk assessment,” *Eur. J. Oper. Res.*, vol. 210, no. 2, pp. 368–378, 2011, doi: 10.1016/j.ejor.2010.09.029.

[29] L. Mosley and D. A. Singer, “The global financial crisis: Lessons and opportunities for international political economy,” *Int. Interact.*, vol. 35, no. 4, pp. 420–429, 2009, doi: 10.1080/03050620903328993.

[30] G. Wims, D. Martens, and M. De Backer, “Network Models of Financial Contagion : A Definition and Literature,” *Ghent Univ. Fac. Econ. Bus. Adm.*, no. July 2011, pp. 1–49, 2011.

[31] BCBS, “Basel- 1: International Convergence of Capital Measurement and Capital Standards,” no. July, p. 28, 1988.

[32] J. Holman, “A Flawed Solution: The difficulties of mandating a leverage ratio in the united states,” *South. Calif. Law Rev.*, vol. 84, no. 3, pp. 713–750, 2011.

[33] J. Breeden, L. Thomas, and J. McDonald, “Stress-testing retail loan portfolios with dual-time dynamics,” *J. Risk Model Valid.*, vol. 2, no. 2, pp. 43–62, 2008, doi: 10.21314/jrmv.2008.033.

[34] Basel Committee on Banking Supervision, “The Internal ratings-based approach,” *Bank Int. Settlments*, no. May, pp. 1–108, 2001.

[35] P.-N. Tan, M. Steinbach, A. Karpatne, and V. Kumar, *Introduction to Data Mining*. 2019.

[36] “2. Over-sampling — Version 0.8.0.” .

[37] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software, 1984.

[38] P. Bruce, A. Bruce, and P. Gedeck, *Practical Statistics for Data Scientists*. 2017.

[39] Scikit Learn, “sklearn.model_selection.GridSearchCV — scikit-learn 0.24.2 documentation,” 2020. .

[40] Z. Zhang and C. Jung, “GBDT-MO: Gradient Boosted Decision Trees for Multiple Outputs.”

[41] “Welcome to LightGBM’s documentation! — LightGBM 3.2.1.99 documentation.” .

[42] “sklearn.model_selection.RandomizedSearchCV — scikit-learn 0.24.2 documentation.” .

[43] “3.3. Metrics and scoring: quantifying the quality of predictions — scikit-learn

0.24.1 documentation.” .

[44] “sklearn.metrics.fbeta_score — scikit-learn 0.24.2 documentation.” .

[45] G. Fowler, “cql: Flat-file database query language,” *WTEC’94 Proc. USENIX Winter 1994 Tech. Conf. USENIX Winter 1994 Tech. Conf.*, 1994.

[46] “What Is a Relational Database | Oracle.” .