

ĐẠI HỌC QUỐC GIA HÀ NỘI  
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN  
KHOA TOÁN - CƠ - TIN HỌC

Tạ Quang Tùng

ỨNG DỤNG ENSEMBLE LEARNING TRONG  
RỦI RO TÍN DỤNG NGÂN HÀNG

KHÓA LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

Ngành: Toán tin  
(Chương trình đào tạo chuẩn)

Hà Nội, 2025

ĐẠI HỌC QUỐC GIA HÀ NỘI  
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN  
KHOA TOÁN - CƠ - TIN HỌC

Tạ Quang Tùng

ỨNG DỤNG ENSEMBLE LEARNING TRONG  
RỦI RO TÍN DỤNG NGÂN HÀNG

KHÓA LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY  
Ngành: Toán tin  
(Chương trình đào tạo chuẩn)

Cán bộ hướng dẫn: Hoàng Thị Phương Thảo

Hà Nội, 2025

# LỜI CẢM ƠN

Lời đầu tiên, em xin phép được gửi lời cảm ơn chân thành nhất tới các thầy cô trong Khoa Toán - Cơ - Tin học, trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Hà Nội đã giảng dạy và truyền cảm hứng cho em trong suốt quá trình học tập để phát triển bản thân. Những bài giảng, sự chỉ bảo tận tâm và kinh nghiệm quý báu từ thầy cô chính là hành trang vô cùng quan trọng giúp em có được nền tảng vững chắc để thực hiện khóa luận này.

Tiếp theo, em cũng xin gửi lời cảm ơn sâu sắc tới cô Hoàng Thị Phương Thảo - Tiến sĩ bộ môn Xác suất thống kê đang làm việc và giảng dạy tại Khoa Toán - Cơ - Tin học, trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Hà Nội. Với sự kiên nhẫn, tận tâm và những góp ý quý báu từ cô, em đã có thể từng bước hoàn thiện đề tài khóa luận tốt nghiệp, mở rộng kiến thức và trau dồi thêm kỹ năng nghiên cứu khoa học tại trường.

Mặc dù đã nỗ lực hoàn thiện trong khả năng của mình, nhưng do thời gian nghiên cứu có hạn và kiến thức còn chưa thực sự sâu rộng, khóa luận tốt nghiệp chắc chắn không thể tránh khỏi những thiếu sót. Em rất mong nhận được sự đóng góp, nhận xét của các quý thầy cô và bạn đọc để có thể tiếp tục hoàn thiện và nâng cao hơn nữa chất lượng của công trình nghiên cứu cho khóa luận tốt nghiệp này.

Em xin chân thành cảm ơn!

Hà Nội, ngày ... tháng 05 năm 2025  
Sinh viên

**Tạ Quang Tùng**

# Mục lục

<b>Lời cảm ơn</b>	<b>1</b>
<b>1 Giới thiệu chung về bài toán:</b>	<b>5</b>
1.1 Khái niệm tín dụng: . . . . .	5
1.2 Khái niệm rủi ro tín dụng: . . . . .	5
1.3 Khái niệm chấm điểm thẻ tín dụng ngân hàng: . . . . .	6
1.4 Lý do chọn đề tài: . . . . .	7
1.5 Mục tiêu khóa luận: . . . . .	8
<b>2 Tổng quan về bài toán và bộ dữ liệu</b>	<b>9</b>
2.1 Lộ trình thực hiện dự đoán sự phê duyệt và chấm điểm của thẻ tín dụng: . . . . .	9
2.1.1 Quy trình phân tích và dự đoán mô hình: . . . . .	9
2.1.2 Tiêu chí phê duyệt trong thẻ tín dụng ngân hàng: . . . . .	11
2.1.3 Tiêu chí chấm điểm thẻ tín dụng: . . . . .	11
2.2 Tổng quan về bộ dữ liệu: . . . . .	12
2.2.1 Bộ dữ liệu: . . . . .	12
2.2.2 Trường thông tin của bộ dữ liệu: . . . . .	12
2.2.3 Khởi tạo biến mục tiêu: . . . . .	13
2.3 Chuẩn bị dữ liệu huấn luyện mô hình: . . . . .	14
<b>3 Khai phá và phân tích dữ liệu</b>	<b>15</b>
3.1 Phân tích dữ liệu: . . . . .	15
3.1.1 Phân tích đơn biến: . . . . .	16
3.1.2 Phân tích hai biến: . . . . .	17
3.2 Xử lý và làm sạch dữ liệu các đặc trưng: . . . . .	19
3.2.1 Xử lý ngoại lệ Outliers (Data Cleaning): . . . . .	20
3.2.2 Lựa chọn đặc trưng (Feature Selection): . . . . .	21
3.2.3 Chuyển đổi dữ liệu đặc trưng (Data Transformation): . . . . .	22
3.2.4 Khắc phục giảm thiểu độ lệch sai số (Bias & Error Mitigation): . . . . .	22
3.2.5 Tiến hành xử lý đặc trưng (Feature Engineering): . . . . .	22
3.3 Tiền xử lý dữ liệu (Data Preprocessing): . . . . .	26
3.4 Lựa chọn và huấn luyện mô hình: . . . . .	26
<b>4 Phương pháp Ensemble Learning</b>	<b>28</b>
4.1 Mô hình Gradient Boosting Classifier: . . . . .	29
4.1.1 Khái niệm Gradient Boosting: . . . . .	29
4.1.2 Quy trình thuật toán: . . . . .	29
4.1.3 Mô hình Gradient Boosting: . . . . .	31
4.1.4 Ví dụ minh họa: . . . . .	32

4.2	Mô hình Bagging Classifier: . . . . .	34
4.2.1	Khái niệm Bootstrap Aggregating: . . . . .	34
4.2.2	Quy trình thuật toán: . . . . .	35
4.2.3	Mô hình Bagging: . . . . .	37
4.2.4	Ví dụ minh họa: . . . . .	37
4.3	Mô hình Stacking Classifier: . . . . .	39
4.3.1	Mô hình tuyến tính Logistic Regression: . . . . .	40
4.3.2	Mô hình cây quyết định Decision Tree: . . . . .	41
4.3.3	Mô hình rừng ngẫu nhiên Random Forest: . . . . .	42
4.3.4	Mô hình ExtraTrees Classifier: . . . . .	43
4.4	Mô hình Gradient Stacking Classifier: . . . . .	45
4.4.1	Khái niệm Gradient Stacking Classifier: . . . . .	45
4.4.2	Quy trình thuật toán: . . . . .	46
<b>5</b>	<b>Kết quả thực nghiệm</b>	<b>47</b>
5.1	So sánh giữa các mô hình: . . . . .	47
5.2	Lý do lựa chọn Gradient Stacking Classifier: . . . . .	48
5.2.1	Ưu điểm: . . . . .	48
5.2.2	Nhược điểm: . . . . .	48
5.3	Kết quả thu được từ mô hình Gradient Stacking Classifier: . . . . .	48
5.3.1	Mối tương quan giữa các đặc trưng: . . . . .	48
5.3.2	Ma trận nhầm lẫn: . . . . .	49
5.3.3	Đường cong ROC: . . . . .	51
5.4	Triển khai và xây dựng giao diện mô hình với Streamlit Web Interface: . . . . .	52
5.4.1	Ngôn ngữ lập trình Python: . . . . .	52
5.4.2	Triển khai mô hình với Streamlit: . . . . .	53
5.4.3	Ngôn ngữ lập trình mở rộng Cython: . . . . .	53
5.5	Sản phẩm đề tài khóa luận tốt nghiệp: . . . . .	54
	<b>Kết luận</b>	<b>59</b>
	<b>Tài liệu tham khảo</b>	<b>60</b>

# Danh sách hình vẽ

Hình 1. Minh họa tín dụng trong hoạt động ngân hàng. . . . .	5
Hình 2. Minh họa rủi ro tín dụng ngân hàng. . . . .	6
Hình 3. Minh họa chấm điểm thẻ tín dụng ngân hàng. . . . .	7
Hình 4. Quy trình Dự đoán phê duyệt thẻ tín dụng . . . . .	9
Hình 5. Hồ sơ thông tin cá nhân cơ bản của ứng viên . . . . .	13
Hình 6. Thông tin tín dụng và trạng thái vay nợ của ứng viên	13
Hình 7. Chia tập dữ liệu thành tập huấn luyện và tập kiểm tra	14
Hình 8. Biểu đồ thể hiện mối tương quan giữa hai biến với nhau	18
Hình 9. Mối tương quan giữa Family member count và Chil- dren count . . . . .	18
Hình 10. Mối tương quan giữa Age và Employment length . .	19
Hình 11. Biểu đồ hộp thể hiện khoảng tứ phân vị . . . . .	21
Hình 12. Một số phương pháp trong Ensemble Learning . . . .	28
Hình 13. Minh họa thuật toán Gradient Boosting . . . . .	29
Hình 14. Minh họa thuật toán Bagging (Bootstrap Aggregating)	35
Hình 15. Minh họa thuật toán Decision Tree . . . . .	37
Hình 16. Minh hoạt các thuật toán trong Stacking . . . . .	39
Hình 17. Minh họa thuật toán Gradient Stacking Classifier . .	45
Hình 18. Ma trận tương quan các đặc trưng bằng Heatmap . .	49
Hình 19. Minh họa về ma trận nhầm lẫn tổng quát . . . . .	50
Hình 20. Ma trận nhầm lẫn của Gradient Stacking Classifier .	50
Hình 21. Minh họa đường cong ROC trên tập dữ liệu . . . . .	51
Hình 22. Framework Streamlit của Python . . . . .	53
Hình 23. So sánh tốc độ xử lý dữ liệu giữa Cython và Python	54
Hình 24. Đánh giá sự chấp thuận và chấm điểm thẻ tín dụng ngân hàng . . . . .	58

# 1 Giới thiệu chung về bài toán:

## 1.1 Khái niệm tín dụng:

Tín dụng là một hình thức giao dịch tài chính phổ biến trong lĩnh vực ngân hàng - tài chính, thể hiện mối quan hệ vay mượn giữa bên cho vay và bên đi vay. Theo đó, bên cho vay (thường là ngân hàng hoặc tổ chức tài chính) chuyển giao cho bên vay một khoản tiền hoặc tài sản trong một khoảng thời gian nhất định, kèm theo cam kết hoàn trả đầy đủ cả gốc và lãi theo các điều kiện đã thỏa thuận từ trước.

Tín dụng đóng vai trò quan trọng trong việc luân chuyển vốn trong nền kinh tế, thúc đẩy tiêu dùng, đầu tư và phát triển sản xuất kinh doanh. Có nhiều loại hình tín dụng khác nhau như tín dụng tiêu dùng, tín dụng thương mại, tín dụng ngân hàng, trong đó phổ biến nhất là hình thức tín dụng do các ngân hàng thương mại cung cấp cho cá nhân và doanh nghiệp nhằm đáp ứng nhu cầu vốn ngắn hạn hoặc dài hạn.



Hình 1: Minh họa tín dụng trong hoạt động ngân hàng.

## 1.2 Khái niệm rủi ro tín dụng:

Rủi ro tín dụng (Credit Risk) là một trong những loại rủi ro tài chính phổ biến nhất trong hoạt động ngân hàng và tài chính, phản ánh khả năng bên vay không thực hiện đúng nghĩa vụ thanh toán đối với bên cho vay theo thỏa thuận đã ký kết. Cụ thể, đó là nguy cơ mà khách hàng, sau khi được cấp tín dụng (bao gồm vay tiền mặt hoặc sử dụng thẻ tín dụng), không có khả năng hoặc không muốn hoàn trả khoản nợ cả gốc và lãi đúng hạn, dẫn đến tổn thất tài chính cho tổ chức tín dụng.

Trong việc sử dụng thẻ tín dụng ngân hàng, rủi ro tín dụng càng trở nên đáng lo ngại khi người dùng không kiểm soát được hành vi chi tiêu và mất khả năng thanh toán khoản tiền đã chi. Trong đó, có các loại rủi ro tín dụng như sau:

- Rủi ro tín dụng (Credit Risk): xảy ra khi người dùng không có khả năng hoàn trả lại khoản tiền đã chi tiêu trong thẻ tín dụng ngân hàng
- Rủi ro gian lận (Fraud Risk): phát sinh khi thẻ tín dụng bị giả mạo hoặc thông tin tài khoản bị đánh cắp
- Rủi ro giao dịch (Transaction Risk): xảy ra trong quá trình xử lý các giao dịch thanh toán không thành công
- Rủi ro bảo mật (Security Risk): liên quan đến việc bảo vệ dữ liệu thẻ tín dụng, thông tin cá nhân và hệ thống thanh toán trong các cuộc tấn công mạng

Và còn một số loại rủi ro tín dụng trong lĩnh vực tài chính - ngân hàng khác nữa. Việc nhận diện và quản lý hiệu quả các dạng rủi ro tín dụng là điều thiết yếu để đảm bảo sự an toàn và phát triển bền vững cho các tổ chức tài chính cũng như cho hệ thống tài chính nói chung.



Hình 2: Minh họa rủi ro tín dụng ngân hàng.

Ngoài ra, việc phân tích và đánh giá rủi ro thẻ tín dụng ngân hàng góp phần và là cơ sở để tổ chức ngân hàng - tài chính thống kê và chấm điểm thẻ tín dụng cho khách hàng.

### **1.3 Khái niệm chấm điểm thẻ tín dụng ngân hàng:**

Chấm điểm thẻ tín dụng ngân hàng là quá trình đánh giá mức độ tín nhiệm của một cá nhân hoặc tổ chức dựa trên các tiêu chí tài chính



và phi tài chính nhằm đưa ra quyết định có nên cấp thẻ tín dụng hay không, cũng như xác định hạn mức tín dụng phù hợp. Hệ thống chấm điểm này thường được xây dựng dựa trên các mô hình thống kê hoặc học máy, sử dụng thông tin từ lịch sử tín dụng, thu nhập, độ ổn định nghề nghiệp, tần suất giao dịch, lịch sử thanh toán, số lượng nợ hiện tại và nhiều yếu tố khác có liên quan đến hành vi tài chính của khách hàng.

Mục tiêu của chấm điểm thẻ tín dụng là giúp ngân hàng giảm thiểu rủi ro tín dụng bằng cách phân loại khách hàng theo các mức độ rủi ro khác nhau. Những khách hàng có điểm tín dụng cao được coi là có khả năng hoàn trả tốt và do đó dễ dàng được cấp thẻ và hạn mức cao hơn. Ngược lại, những người có điểm tín dụng thấp sẽ bị hạn chế cấp thẻ, hoặc được áp dụng các điều kiện nghiêm ngặt hơn. Việc áp dụng hệ thống chấm điểm một cách khoa học và tự động hóa giúp ngân hàng ra quyết định nhanh chóng, minh bạch và giảm thiểu sự phụ thuộc vào đánh giá chủ quan từ cán bộ tín dụng.



Hình 3: Minh họa chấm điểm thẻ tín dụng ngân hàng.

Trong bối cảnh phát triển của công nghệ tài chính và dữ liệu lớn, chấm điểm tín dụng ngày càng trở thành một công cụ không thể thiếu trong quản lý rủi ro tín dụng và nâng cao hiệu quả hoạt động cấp tín dụng tại các ngân hàng thương mại.

## 1.4 Lý do chọn đề tài:

Trong bối cảnh nền kinh tế ngày càng phát triển và nhu cầu vay tiêu dùng, sử dụng thẻ tín dụng gia tăng, các tổ chức tín dụng, đặc biệt là ngân hàng, đang ngày càng đối mặt với nhiều rủi ro liên quan đến khả năng thanh toán của khách hàng. Trong đó, rủi ro tín dụng luôn là một trong những vấn đề cốt lõi ảnh hưởng trực tiếp đến chất lượng tài sản

và sự an toàn tài chính của ngân hàng. Để quản trị hiệu quả rủi ro này, việc xây dựng hệ thống chấm điểm tín dụng nhằm đánh giá mức độ tin cậy của khách hàng là cực kỳ quan trọng.

Ngày nay, với sự phát triển mạnh mẽ của trí tuệ nhân tạo và học máy, các mô hình dự đoán hiện đại đã được ứng dụng ngày càng rộng rãi trong lĩnh vực tài chính – ngân hàng. Đặc biệt, các phương pháp Ensemble Learning như Gradient Boosting, Bagging, Random Forest, ... đã cho thấy hiệu quả vượt trội trong việc cải thiện độ chính xác của mô hình phân loại và dự báo. Việc ứng dụng những kỹ thuật học máy tiên tiến này vào bài toán chấm điểm tín dụng không chỉ giúp tăng độ chính xác trong đánh giá rủi ro mà còn hỗ trợ ra quyết định cấp tín dụng nhanh chóng và hiệu quả hơn.

Xuất phát từ thực tiễn đó, em lựa chọn đề tài "Ứng dụng Ensemble Learning trong rủi ro tín dụng ngân hàng" với mong muốn được tìm hiểu, nghiên cứu sâu hơn về lĩnh vực phân tích dữ liệu tài chính kết hợp với học máy, đồng thời đóng góp một phần nhỏ trong việc xây dựng các hệ thống hỗ trợ đánh giá tín dụng hiệu quả, an toàn và khoa học hơn.

## **1.5 Mục tiêu khóa luận:**

Khóa luận tốt nghiệp này hướng tới mục tiêu xây dựng mô hình dự đoán rủi ro tín dụng của khách hàng dựa trên dữ liệu đăng ký cấp thẻ hồ sơ tín dụng của khách hàng, từ đó hỗ trợ đưa ra quyết định cấp tín dụng một cách hiệu quả và khách quan. Cụ thể, khóa luận sẽ được áp dụng và so sánh hiệu suất của các mô hình học máy gồm: Gradient Boosting Classifier, Bagging Classifier, Hồi quy Logistic, Cây quyết định (Decision Tree) và Rừng ngẫu nhiên (Random Forest), ... để đánh giá xác suất rủi ro tín dụng của khách hàng, từ đó có thể có số điểm tín dụng phù hợp với từng hồ sơ. Cuối cùng, khóa luận hướng đến phát triển một ứng dụng hỗ trợ đánh giá thẻ tín dụng, cho phép người dùng nhập thông tin cơ bản để kiểm tra khả năng được chấp thuận cấp thẻ tín dụng, đồng thời không làm ảnh hưởng tiêu cực đến điểm tín dụng hiện tại của họ.

Tuy còn khó khăn trong việc xin dữ liệu tín dụng thực tế của khách hàng từ các ngân hàng nhưng mà em vẫn có thể mô phỏng và thống kê được khả năng rủi ro tín dụng và chấm điểm thẻ tín dụng đó thông qua bộ dữ liệu từ trang Kaggle - một trang website chuyên cung cấp các bộ dữ liệu cộng đồng miễn phí.

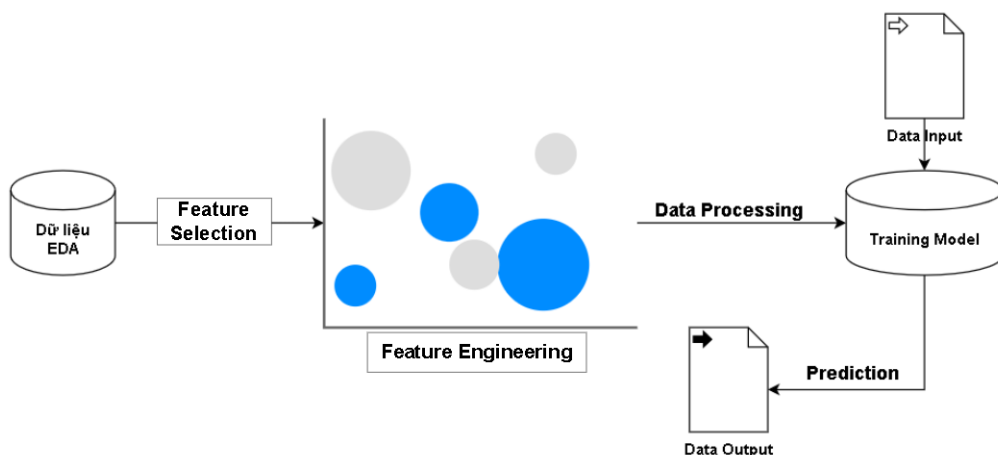
## 2 Tổng quan về bài toán và bộ dữ liệu

### 2.1 Lộ trình thực hiện dự đoán sự phê duyệt và chấm điểm của thẻ tín dụng:

#### 2.1.1 Quy trình phân tích và dự đoán mô hình:

Trong quá trình xây dựng mô hình học máy để dự đoán xác suất khả năng phê duyệt thẻ tín dụng và chấm điểm thẻ tín dụng của khách hàng, ta cần hiểu được quy trình làm việc và các bước phân tích rõ ràng nhằm đảm bảo độ chính xác, hiệu quả và khả năng triển khai thực tế của mô hình. Dưới đây là quy trình dự đoán phê duyệt thẻ tín dụng và chấm điểm thẻ của khách hàng:

1. Khai phá và phân tích dữ liệu (EDA - Exploratory Data Analysis)
2. Thực hiện lựa chọn đặc trưng (Feature Selection)
3. Tiến hành xử lý đặc trưng (Feature Engineering)
4. Tiền xử lý dữ liệu (Data Preprocessing)
5. Huấn luyện mô hình (Model Training)
6. Lựa chọn mô hình (Model Selection)
7. Xây dựng giao diện Website cho mô hình bằng Streamlit
8. Triển khai mô hình (Deploy the Model)



Hình 4: Quy trình Dự đoán phê duyệt thẻ tín dụng

Bước đầu tiên là khai phá và phân tích dữ liệu (Exploratory Data Analysis - EDA). Ở giai đoạn này, các nhà phân tích sẽ tiến hành khảo

sát tổng quan bộ dữ liệu, tìm hiểu phân phối của các thuộc tính, phát hiện các giá trị ngoại lai (outliers), dữ liệu thiếu (missing values), cũng như mối tương quan giữa các biến độc lập và biến mục tiêu. Việc hiểu rõ dữ liệu là nền tảng quan trọng để quyết định các bước xử lý tiếp theo.

Sau khi đã hiểu rõ về dữ liệu, ta thực hiện lựa chọn đặc trưng (Feature Selection) nhằm loại bỏ những thuộc tính không liên quan hoặc gây nhiễu, từ đó giảm độ phức tạp của mô hình và tăng hiệu quả dự đoán. Quá trình này có thể được thực hiện thủ công hoặc sử dụng các phương pháp thống kê, học máy để đánh giá độ quan trọng của từng đặc trưng.

Tiếp theo là bước xử lý đặc trưng (Feature Engineering) – nơi các thuộc tính mới được tạo ra dựa trên các đặc trưng hiện có nhằm tăng tính biểu diễn của mô hình. Việc kết hợp, tách hoặc biến đổi các đặc trưng này giúp mô hình học được nhiều thông tin hơn từ cùng một tập dữ liệu.

Khi đã hoàn thiện các đặc trưng, dữ liệu sẽ được đưa vào bước tiền xử lý (Data Preprocessing). Giai đoạn này bao gồm chuẩn hóa dữ liệu, mã hóa dữ liệu phân loại (encoding), xử lý giá trị thiếu, cân bằng tập dữ liệu (nếu mất cân đối giữa các lớp)... để đảm bảo dữ liệu ở trạng thái tốt nhất trước khi huấn luyện.

Sau đó, mô hình được đưa vào giai đoạn huấn luyện (Model Training). Tại đây, các thuật toán học máy được áp dụng để học từ dữ liệu và tìm ra quy luật phân loại hoặc hồi quy nhằm dự đoán xác suất phê duyệt thẻ tín dụng. Một số thuật toán được sử dụng như Gradient Boosting, Bagging, Logistic Regression, v.v.

Để tìm ra mô hình phù hợp nhất, bước lựa chọn mô hình (Model Selection) được tiến hành. Các mô hình được đánh giá dựa trên các chỉ số như Accuracy, Precision, Recall, F1-score hoặc ROC-AUC nhằm đảm bảo lựa chọn được mô hình có hiệu suất cao và phù hợp với mục tiêu đề ra. Cuối cùng, mô hình được triển khai (Deploy the Model) lên hệ thống thực tế để phục vụ người dùng hoặc tích hợp vào hệ thống ngân hàng. Việc triển khai đảm bảo rằng mô hình hoạt động liên tục, chính xác, bảo mật và sẵn sàng mở rộng khi cần thiết.

Toàn bộ quy trình trên hướng tới mục tiêu cuối cùng: dự đoán xác suất phê duyệt thẻ tín dụng của khách hàng một cách chính xác, minh bạch, mà không làm ảnh hưởng đến điểm tín dụng hiện tại của họ, đồng thời hỗ trợ ngân hàng trong việc ra quyết định nhanh chóng và tối ưu.

### 2.1.2 Tiêu chí phê duyệt trong thẻ tín dụng ngân hàng:

Ứng dụng dựa trên dữ liệu cá nhân (điểm tín dụng, thu nhập, tỷ lệ nợ, lịch sử giao dịch thẻ, ...) để dự đoán kết quả phê duyệt hoặc không phê duyệt của thẻ tín dụng dựa trên một số thông tin hứa ích sau:

- Thu nhập ổn định (Income): thu nhập hàng tháng đủ cao để đáp ứng tiêu chuẩn tối thiểu
- Số lượng thành viên trong gia đình (Family Member Headcount): lượng chi tiêu chi phí tính trên đầu người
- Thời gian làm việc (Employment Length): thu nhập sẽ dựa trên kinh nghiệm làm việc lâu dài
- Một số thông tin khác bán quan trọng như: Children Count, Account Age, Is High Risk, Phone, Email, ...

=> Đáp ứng các nhu cầu do tổ chức tài chính, ngân hàng đặt ra.

### 2.1.3 Tiêu chí chấm điểm thẻ tín dụng:

Sau khi đã tính toán được mức độ rủi ro phê duyệt của thẻ tín dụng, ta có thể chấm điểm thẻ tín dụng đó của khách hàng thông qua một số tiêu chí sau:

- Mức thang điểm thẻ tín dụng của khách hàng luôn bắt đầu từ 100
- Tỷ lệ phần trăm của mức độ rủi ro đánh giá trực tiếp đến việc chấm điểm của thẻ tín dụng, tức là độ rủi ro càng cao thì điểm tín dụng càng thấp và ngược lại

Công thức tính điểm thẻ tín dụng dựa trên xác suất rủi ro tín dụng từ dự đoán các mô hình theo dạng cơ bản Log Odds Scaling:

$$Score = 100 + B \times \log\left(\frac{P(0)}{P(1)}\right)$$

trong đó:

- Điểm cơ sở Base Score là 100, đây là điểm tín dụng cơ bản mà mọi khách hàng đều có trước khi xét đến yếu tố rủi ro
- $P(0)$  và  $P(1)$  lần lượt là xác suất rủi ro tín dụng ứng với khả năng không rủi ro và rủi ro của khách hàng
- Tỷ lệ  $\frac{P(0)}{P(1)}$  phản ánh mức độ an toàn tín dụng của khách hàng

- Hệ số điều chỉnh  $B$  giúp điều chỉnh mức độ thay đổi điểm tín dụng ngân hàng, biết rằng:
  - Nếu  $10 \leq B \leq 50$  thì điểm tín dụng thay đổi dựa trên mô hình ổn định
  - Nếu  $50 \leq B \leq 100$  thì điểm tín dụng thay đổi dựa trên mô hình nhạy cảm về độ rủi ro

## 2.2 Tổng quan về bộ dữ liệu:

### 2.2.1 Bộ dữ liệu:

Bộ dữ liệu Credit Card Approval Prediction được lấy từ trang Kaggle cung cấp hai file.csv là *application\_record.csv* và *credit\_record.csv* cho việc phân tích và thống kê dữ liệu học máy. Dựa vào bộ dữ liệu trên, ta có thể dự đoán khả năng rủi ro tín dụng của khách hàng, đồng thời đánh giá điểm thẻ tín dụng trên hồ sơ của khách hàng đó.

Link datasets: [kaggle.com/datasets/credit-card-approval-prediction](https://kaggle.com/datasets/credit-card-approval-prediction)

Hồ sơ đăng ký thẻ tín dụng của ứng viên gồm một số đặc trưng quan trọng như: Gender (giới tính), Age (độ tuổi), Marital Status (tình trạng hôn nhân), Children Count (số lượng con cái), Employment Length (thời gian làm việc), Income (thu nhập), Education Level (trình độ học vấn), Has A Car (sở hữu xe), Has A Property (sở hữu tài sản), Account Age (độ tuổi tài khoản).

### 2.2.2 Trường thông tin của bộ dữ liệu:

Để có được cái nhìn tổng thể về khách hàng và đánh giá điểm thẻ tín dụng được, ta cần phải gộp hai tập dữ liệu này lại dựa trên một khóa chung (gọi là ID). Việc này giúp kết nối thông tin cá nhân với lịch sử tín dụng, từ đó hỗ trợ quá trình phân tích và ra quyết định chính xác hơn. Biết rằng, bộ dữ liệu có khoảng 20 thuộc tính và có hơn 46000 điểm dữ liệu (bản ghi).

1. Hồ sơ thông tin cơ bản của ứng viên:

application_record.csv		
Feature name	Explanation	Remarks
ID	Client number	
CODE_GENDER	Gender	
FLAG_OWN_CAR	Is there a car	
FLAG_OWN_REALTY	Is there a property	
CNT_CHILDREN	Number of children	
AMT_INCOME_TOTAL	Annual income	
NAME_INCOME_TYPE	Income category	
NAME_EDUCATION_TYPE	Education level	
NAME_FAMILY_STATUS	Marital status	
NAME_HOUSING_TYPE	Way of living	
DAYS_BIRTH	Birthday	Count backwards from current day (0), -1 means yesterday
DAYS_EMPLOYED	Start date of employment	Count backwards from current day(0). If positive, it means the person currently unemployed.
FLAG_MOBIL	Is there a mobile phone	
FLAG_WORK_PHONE	Is there a work phone	
FLAG_PHONE	Is there a phone	
FLAG_EMAIL	Is there an email	
OCCUPATION_TYPE	Occupation	
CNT_FAM_MEMBERS	Family size	

Hình 5: Hồ sơ thông tin cá nhân cơ bản của ứng viên

## 2. Hồ sơ tín dụng và trạng thái của ứng viên đó:

credit_record.csv		
Feature name	Explanation	Remarks
ID	Client number	
MONTHS_BALANCE	Record month	The month of the extracted data is the starting point, backwards, 0 is the current month, -1 is the previous month, and so on
STATUS	Status	0: 1-29 days past due 1: 30-59 days past due 2: 60-89 days overdue 3: 90-119 days overdue 4: 120-149 days overdue 5: Overdue or bad debts, write-offs for more than 150 days C: paid off that month X: No loan for the month

Hình 6: Thông tin tín dụng và trạng thái vay nợ của ứng viên

### 2.2.3 Khởi tạo biến mục tiêu:

Giả sử khách hàng có khoản nợ trên 60 ngày thì coi là "khách hàng xấu". Khi đó, biến mục tiêu là biến nhị phân thỏa mãn:

$$IsHighRisk = \begin{cases} 1 & \text{nếu khách hàng xấu} \\ 0 & \text{nếu khách hàng tốt} \end{cases}$$

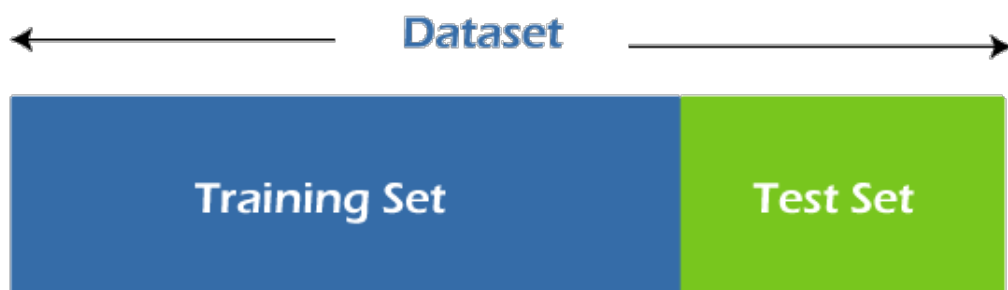
Quy trình khởi tạo biến mục tiêu:

1. Tính độ tuổi của tài khoản cho mỗi khách hàng
2. Xác định khách hàng có rủi ro tín dụng cao theo cột STATUS (tình trạng thanh toán theo từng tháng)
  - Nếu STATUS = 0 thì khách hàng không có nợ quá hạn
  - Nếu STATUS > 0 thì khách hàng có khả năng rủi ro tín dụng cao mức STATUS
3. Thống kê dữ liệu vào cột mục tiêu "IsHighRisk"

*Lưu ý: Việc khởi tạo thêm biến mục tiêu trên giúp đề tài khóa luận được đưa về bài toán phân loại học máy có giám sát (Supervised Learning) vì trong bài toán có dữ liệu đầu vào như các thuộc tính: Income, Family member count, Employment length, ... và có nhãn đầu ra IsHighRisk trong quá trình huấn luyện mô hình.*

## 2.3 Chuẩn bị dữ liệu huấn luyện mô hình:

- Dữ liệu gốc gồm 20 trường thông tin khác nhau được phân tách thành 2 tập là tập train và tập test. Trong đó, tập train chiếm 80% và tập test chiếm 20%
- Thống kê một số thông tin quan trọng từ bộ dữ liệu như: count, mean, std, max-min, khoảng tứ phân vị, ... qua các dạng biểu đồ khác nhau.



Hình 7: Chia tập dữ liệu thành tập huấn luyện và tập kiểm tra

Trong quá trình xây dựng mô hình học máy, việc chia dữ liệu thành 80% cho tập huấn luyện và 20% cho tập kiểm tra là một lựa chọn phổ biến nhằm cân bằng giữa việc học và đánh giá mô hình. Tập huấn luyện đủ lớn giúp mô hình học được các đặc trưng quan trọng từ dữ liệu, trong khi tập kiểm tra đủ đa dạng để đánh giá hiệu suất mô hình trên dữ liệu chưa từng thấy.



### 3 Khai phá và phân tích dữ liệu

Trong tập dữ liệu không có biến mục tiêu thể hiện "khách hàng tốt" và "khách hàng xấu". Ta cần quy trình phân tích Vintage Analysis để tạo biến mục tiêu từ *credit\_record*

*Chú ý: Vintage Analysis là phương pháp thống kê nhằm phân tích dữ liệu dựa theo nhóm thời gian để theo dõi hiệu suất của các nhóm khách hàng.*

#### 3.1 Phân tích dữ liệu:

Phân tích dữ liệu là một bước quan trọng và không thể thiếu trong bất kỳ quy trình xây dựng mô hình học máy nào. Mục tiêu của bước này là hiểu rõ cấu trúc, phân bố và mối quan hệ giữa các đặc trưng trong tập dữ liệu, từ đó giúp đưa ra các quyết định hợp lý về tiền xử lý, lựa chọn đặc trưng và lựa chọn mô hình. Trong đề tài khóa luận này, quá trình phân tích dữ liệu được triển khai qua các bước cụ thể sau:

1. Xử lý thông tin dữ liệu bị mất mát Job Title
2. Tính số lượng và tần suất xuất hiện của các đặc trưng thuộc tính
3. Mô tả thông tin dữ liệu thống kê của từng đặc trưng:
  - Tính số lượng và tần suất của từng lớp trong đặc trưng
  - Mô tả thông tin dữ liệu thống kê của từng đặc trưng
4. Vẽ biểu đồ trực quan hóa dữ liệu:
  - Biểu đồ tròn (pie plot): trực quan hóa tỷ lệ phân bố các lớp trong đặc trưng
  - Biểu đồ cột (bar plot): so sánh số lượng các lớp trong đặc trưng
  - Biểu đồ hộp (box plot): phân tích phân phối và các giá trị ngoại lai trong đặc trưng
  - Biểu đồ histogram (hist plot): phân tích phân bố các đặc trưng
5. Phân tích đơn biến, đa biến

Toàn bộ quá trình phân tích dữ liệu trên giúp tạo nền tảng vững chắc trước khi bước sang giai đoạn tiền xử lý dữ liệu và huấn luyện mô hình, đảm bảo rằng dữ liệu được làm sạch sẽ và sẵn sàng cho các bước xử lý tiếp theo.

### 3.1.1 Phân tích đơn biến:

Phân tích đơn biến (Univariate Analysis) là kỹ thuật phân tích cơ bản cho dữ liệu thống kê, dữ liệu chỉ có một biến và đo lường khía cạnh về lượng của dữ liệu đo mà không xem xét tới mối quan hệ giữa nhiều biến khác nhau.

Điều này đặc biệt hữu ích trong giai đoạn tiền xử lý và khám phá dữ liệu (Exploratory Data Analysis - EDA), giúp nhà phân tích làm quen với dữ liệu, phát hiện bất thường và hỗ trợ ra quyết định cho các bước phân tích tiếp theo, chẳng hạn như lựa chọn phương pháp xử lý dữ liệu phù hợp hoặc xác định biến quan trọng trong mô hình học máy.

- Mục tiêu:
  - Mô tả, tóm tắt dữ liệu và tìm ra kiểu mẫu trong dữ liệu đó
  - Tìm các giá trị trung bình (mean), mode, giá trị trung vị (median), độ lệch chuẩn (standard deviation), ....
- Các kỹ thuật:
  - Summary Statistics (Thống kê): tóm tắt một tập hợp quan sát để lấy ra thông tin đơn giản
  - Frequency distribution table (Bảng phân phối tần suất): số lần giá trị đó xuất hiện trong tổng thể dữ liệu đang xét
  - Vẽ biểu đồ trực quan hóa dữ liệu
- Phân loại kiểu biến:
  - Biến rời rạc: Gender, Marital status, Dwelling, Employment status, Education level, Has a car, Has a work phone, Has a phone, Has an email, Job title
  - Biến liên tục: Age, Children count, Income, Employment length, Account age
  - Biến mục tiêu: Is high risk
- Các đặc trưng quan trọng trong bộ dữ liệu: "Age", "Material status", "Income", "Employment length", "Account age" vì chúng là một trong những nhân tố đóng góp vào việc dự đoán và thống kê tỷ lệ phần trăm các ứng viên đó có rủi ro tín dụng cao hay không (IsHighRisk).

- Các đặc trưng trong bộ dữ liệu tham gia phân tích đơn biến:

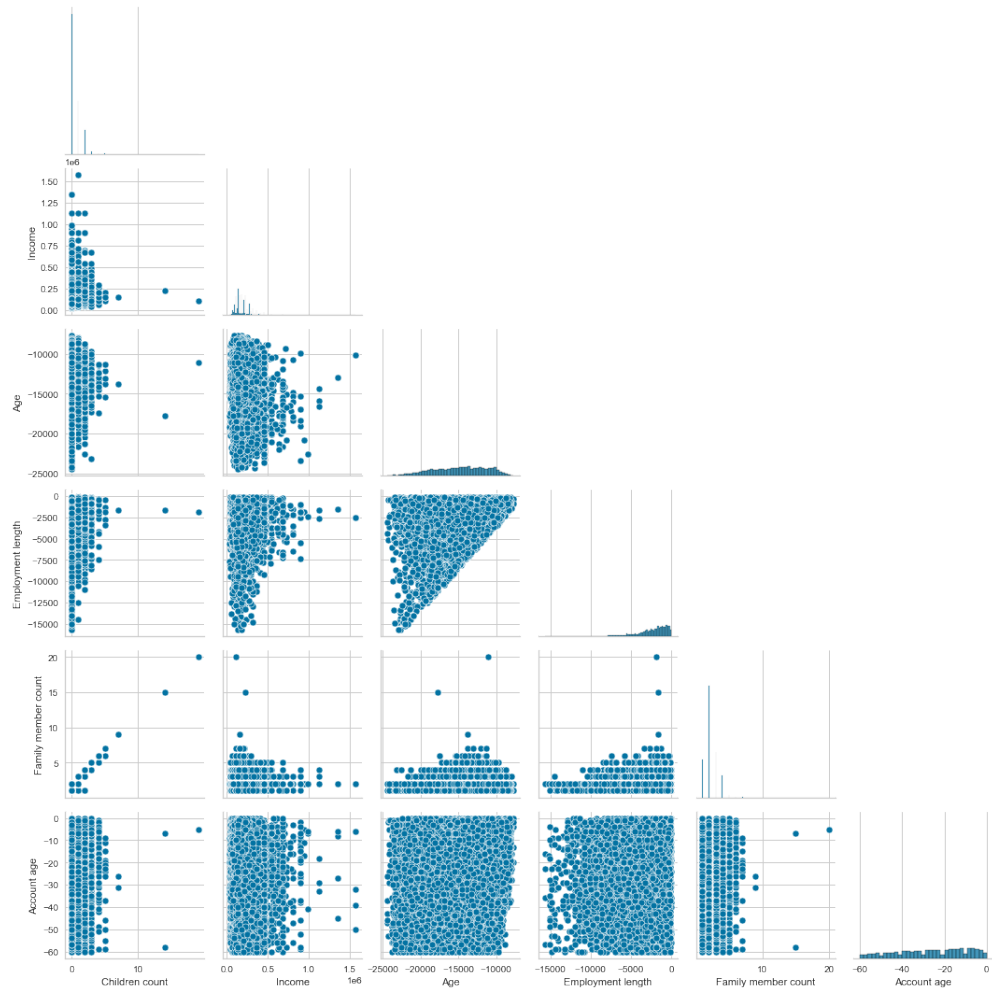
Đặc trưng	Ý nghĩa
"Gender"	Giới tính khách hàng
"Age"	Độ tuổi khách hàng
"Marital status"	Tình trạng hôn nhân
"Family member count"	Số lượng thành viên trong gia đình
"Children count"	Số lượng con cái
"Dwelling"	Loại hình nhà ở
"Income"	Thu nhập
"Employment status"	Trạng thái việc làm
"Education level"	Trình độ học vấn
"Employment length"	Thời gian làm việc
"Has a car"	Có xe riêng không?
"Has a property"	Có sở hữu tài sản không?
"Has a work phone"	Có số điện thoại cơ quan không?
"Has a mobile phone"	Có số điện thoại di động không?
"Has a phone"	Có số điện thoại nhà không?
"Has an email"	Có địa chỉ email không?
"Account age"	Độ tuổi của tài khoản
"Job title"	Chức danh công việc (dữ liệu mất mát)
"Is high risk"	Độ rủi ro cao không?

### 3.1.2 Phân tích hai biến:

Phân tích hai biến (Bivariate Analysis) là một phần của phân tích đa biến, đây là phương pháp phân tích dữ liệu tập trung vào việc kiểm tra mối quan hệ và sự tương quan giữa hai biến số trong cùng một tập dữ liệu. Đây là bước quan trọng trong phân tích dữ liệu để xác định xem liệu có mối liên hệ như thế nào giữa các biến.

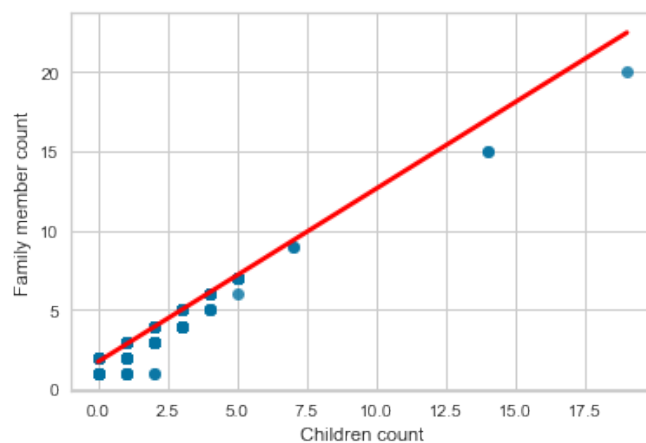
Trong bài toán này, ta cần phân tích đa biến giữa các đặc trưng khác như "Children count", "Income", "Age", "Employment length", "Family member count", "Account Age" để thấy rõ được mối quan hệ và tính tương quan giữa từng cặp biến với nhau.

Cụ thể, ta có biểu đồ mối quan hệ giữa các cặp biến như sau:



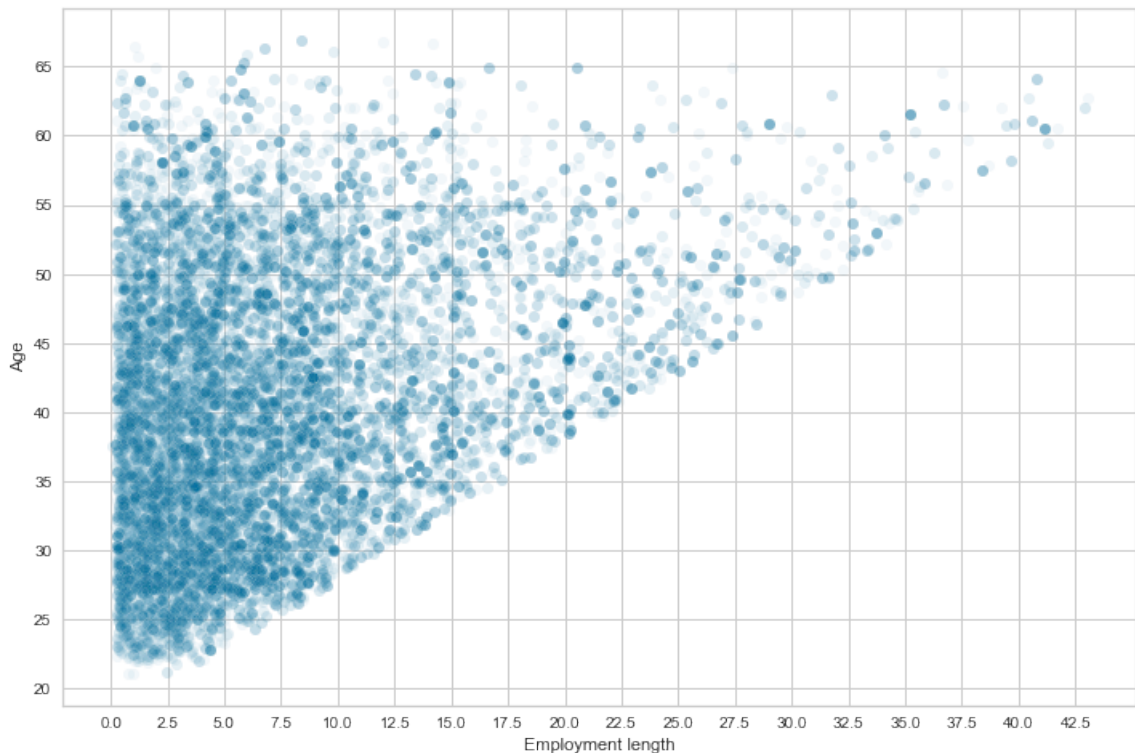
Hình 8: Biểu đồ thể hiện mối tương quan giữa hai biến với nhau

Trong biểu đồ trên, ta thấy có mối quan hệ tuyến tính dương giữa "Family Member Count" với "Children Count", tuy nhiên điều này xảy ra vấn đề đa cộng tuyến tính (Multicollinearity) không thích hợp để huấn luyện mô hình. Do đó, ta cần loại bỏ đi một trong hai đặc trưng này để đảm bảo việc training model.



Hình 9: Mối tương quan giữa Family member count và Children count

Tương tự đối với mối quan hệ giữa hai đặc trưng "Age" và "Employment Length" cũng là có mối quan hệ tuyến tính qua biểu đồ Scatter Plot nên cũng sẽ cần loại bỏ đi một trong hai đặc trưng đi để đảm bảo quá trình huấn luyện mô hình diễn ra tốt nhất.



Hình 10: Mối tương quan giữa Age và Employment length

*Chú ý: Đa cộng tuyến tính là hiện tượng xảy ra khi hai hay nhiều biến độc lập trong mô hình hồi quy tuyến tính có mối tương quan chặt chẽ với nhau. Điều này làm cho việc ước lượng hệ số hồi quy trở nên không chính xác, dẫn đến khó khăn trong việc đánh giá tác động riêng biệt của từng biến đến biến phụ thuộc.*

### 3.2 Xử lý và làm sạch dữ liệu các đặc trưng:

Sau khi tiến hành phân tích đơn biến và đa biến, ta có cái nhìn tổng quan về đặc điểm phân phối cũng như mối quan hệ giữa các biến trong bộ dữ liệu. Bước tiếp theo sau đây là thực hiện xử lý và làm sạch dữ liệu đặc trưng, nhằm loại bỏ các giá trị thiếu, nhiễu hoặc không hợp lệ, đồng thời chuẩn hóa và mã hóa các biến phù hợp để sẵn sàng đưa vào quá trình huấn luyện mô hình học máy. Việc xử lý dữ liệu đúng cách đóng vai trò quan trọng trong việc nâng cao hiệu quả và độ chính xác của mô hình sau này.

- Xử lý ngoại lệ outliers: Family Member Count, Employment Length, Income
- Các đặc trưng cần loại bỏ: ID, Has A Mobile Phone, Account Age, Children Count, Job Title
- Chuyển đổi giá trị dữ liệu ngày: Age, Employment Length
- Thay đổi giá trị với điều kiện thỏa mãn: Employment Length
- Khắc phục giảm thiểu độ lệch sai số: Age, Income
- Sử dụng Binary Encoding: Has a work phone, Has a phone, Has an email
- Sử dụng One-Hot Encoding: Gender, Material Status, Employment Status, Dwelling, Has A Work Phone, Has A Phone, Has An Email, Has A Car, Has A Property
- Sử dụng Ordinal Encoding: Education Level
- Áp dụng Min-Max scaling: Age, Employment Length, Income

trong đó đặc trưng *Is High Risk* cần chuyển đổi dữ liệu thành dạng số và cân bằng bằng dữ liệu với SMOTE.

### 3.2.1 Xử lý ngoại lệ Outliers (Data Cleaning):

Outliers là những điểm dữ liệu có khác biệt đáng kể trong tập dữ liệu nói chung. Outliers có thể làm nhiễu trong quá trình training model trong khoảng thời gian nhất định khiến cho kết quả dự đoán sai khác nhiều so với kết quả thực tế.

Xét tập dữ liệu với  $X[\text{feature}] = \{x_1, x_2, \dots, x_n\}$  được xếp theo thứ tự tăng dần đối với feature đó, ta có bảng tương ứng sau:

Khoảng	$[a_1, a_2)$	$[a_2, a_3)$	$[a_3, a_4)$	...	$[a_p, a_{p+1})$	...
Số lượng	$m_1$	$m_2$	$m_3$	...	$m_p$	...

feature = ["Family member count", "Income", "Employment length"]

1. Tìm vị trí nhóm chứa tứ phân vị thứ r:

$$R = \frac{rn}{4} \text{ nằm trong khoảng } [a_p, a_{p+1})$$

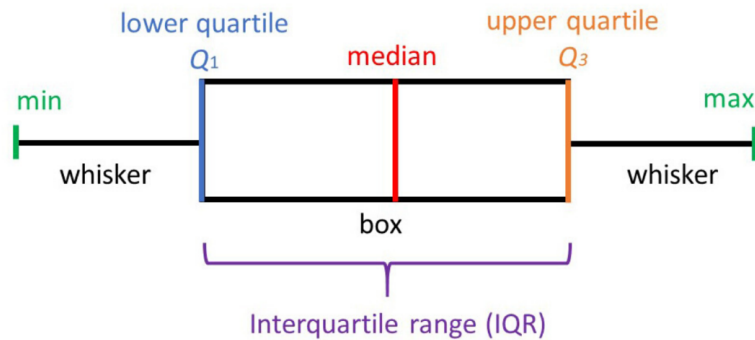
2. Công thức tứ phân vị thứ r:

$$Q_r = a_p + \frac{R - (m_1 + m_2 + \dots + m_{p-1})}{m_p} (a_{p+1} - a_p)$$

Ta có:

- $Q_1$  là giá trị phân vị thứ 25% ( $r = 1$ )
- $Q_3$  là giá trị phân vị thứ 75% ( $r = 3$ )
- $IQR = Q_3 - Q_1$  là khoảng tứ phân vị

*Phương án: Loại bỏ các mẫu dữ liệu ra khỏi tập dữ liệu nếu chúng nằm ngoài khoảng  $[Q_1 - 3 \times IQR, Q_3 + 3 \times IQR]$*



Hình 11: Biểu đồ hộp thể hiện khoảng tứ phân vị

### 3.2.2 Lựa chọn đặc trưng (Feature Selection):

Tại bước lựa chọn các đặc trưng phù hợp cho mô hình, ta cần phải loại bỏ đi các đặc trưng không hữu ích trong quá trình huấn luyện và dự đoán như là: "ID", "Has a mobile phone", "Children count", "Job title", "Account age".

Lý do loại bỏ các đặc trưng này đi vì:

- ID: Số thứ tự cho các mẫu dữ liệu
- Has a mobile phone: Việc có hay không Mobile Phone không hữu ích cho việc huấn luyện mô hình
- Children count: mối quan hệ tuyến tính dương mạnh với đặc trưng "Family member count" nên chỉ lấy một trong hai đặc trưng đó
- Job title: chứa nhiều dữ liệu mất mát
- Account age: đã được sử dụng để khởi tạo biến mục tiêu "Is High Risk" nên cần loại bỏ để tránh overfitting

*Overfitting là hiện tượng mô hình học máy có độ chính xác cao đối với bộ dữ liệu huấn luyện nhưng độ chính xác thấp đối với bộ dữ liệu mới.*

### 3.2.3 Chuyển đổi dữ liệu đặc trưng (Data Transformation):

Trong quá trình tiền xử lý dữ liệu, hai đặc trưng "Employment length" (số ngày làm việc) và "Age" (tuổi) ban đầu được lưu dưới dạng giá trị âm, do dữ liệu gốc tính theo số ngày trước thời điểm hiện tại. Do đó, để thuận tiện cho việc phân tích và huấn luyện mô hình, hai cột này được chuyển đổi về giá trị dương bằng cách lấy trị tuyệt đối.

$$|Value| = \begin{cases} Value & \text{nếu } Value \geq 0 \\ -Value & \text{nếu } Value < 0 \end{cases}$$

Ngoài ra, trong cột "Employment length", một số khách hàng đã nghỉ hưu có giá trị được gán là 365243 ngày – đây là giá trị đặc biệt đại diện cho trường hợp không còn lao động. Tuy nhiên, để tránh ảnh hưởng đến mô hình, giá trị này được thay thế bằng 0, biểu thị cho thời gian làm việc hiện tại là không còn.

### 3.2.4 Khắc phục giảm thiểu độ lệch sai số (Bias & Error Mitigation):

Một số đặc trưng như "Income" (Thu nhập) và "Age" (Tuổi) trong tập dữ liệu có phân phối bị lệch (skewed), điều này có thể ảnh hưởng đến hiệu suất của các mô hình học máy. Do đó, cần thực hiện bước chuẩn hóa phân phối để dữ liệu có xu hướng gần với phân phối chuẩn hơn. Cụ thể, nhóm dữ liệu sử dụng căn bậc ba (cube root transformation) để làm giảm độ lệch phân phối của các cột này như sau:

$$newValue = \sqrt[3]{oldValue}$$

trong đó:

- newValue là giá trị sau khi giảm thiểu sai số của thuộc tính
- oldValue là giá trị thực tế ban đầu của bộ dữ liệu

### 3.2.5 Tiến hành xử lý đặc trưng (Feature Engineering):

#### 1. Mã hóa Binary Encoding:

Binary Encoding là một kỹ thuật mã hóa dữ liệu phân loại (categorical data) bằng cách kết hợp giữa Ordinal Encoding và One-Hot Encoding. Cụ thể, mỗi giá trị phân loại đầu tiên được chuyển thành một số nguyên (theo thứ tự xuất hiện), sau đó số nguyên này được chuyển sang dạng nhị phân. Cuối cùng, mỗi bit trong biểu diễn nhị phân đó được tách thành một cột riêng biệt.



Trong khóa luận tốt nghiệp này, ta cần mã hóa dữ liệu Yes-No của các thuộc tính "Has a work phone", "Has a phone", "Has an email" về dạng nhị phân (1 - 0) để thuận tiện cho quá trình huấn luyện mô hình học máy. Cụ thể như sau:

$$newValue = \begin{cases} 0 & \text{nếu oldValue} = N \\ 1 & \text{nếu oldValue} = Y \end{cases}$$

trong đó:

- newValue là giá trị mới của thuộc tính sau khi mã hóa nhị phân
- oldValue là giá trị thực tế ban đầu của bộ dữ liệu

## 2. Mã hóa One-Hot Encoding:

One-Hot Encoding là một phương pháp biến đổi mã hóa các biến phân loại thành dạng nhị phân để máy học có thể xử lý được. Hầu hết các thuật toán Machine Learning không thể làm việc trực tiếp với dữ liệu dạng chữ (text) mà cần phải chuyển đổi chúng thành dạng số. Do đó, One-Hot Encoding là một kỹ thuật quan trọng để biểu diễn dữ liệu phi số thành dạng số mà không làm mất đi ý nghĩa của chúng.

Các thuộc tính cần mã hóa: : "Gender", "Material status", "Employment status", "Dwelling", "Has a work phone", "Has a phone", "Has an email", "Has a car", "Has a property".

Cho bảng dữ liệu sau:

Feature 1	Feature 2	Feature 3
$A_1$	$B_1$	$C_1$
$A_2$	$B_2$	$C_1$
$A_3$	$B_2$	$C_1$
...	...	...

Giả sử, ta có các giá trị của từng đặc trưng Feature 1 =  $\{A_1, A_2, A_3\}$ , Feature 2 =  $\{B_1, B_2\}$ , Feature 3 =  $\{C_1\}$ . Khi đó, để One-Hot Encoding các đặc trưng đó thì ta sẽ tạo thêm các cột sau:

- Feature 1: isA1, isA2, isA3
- Feature 2: isB1, isB2
- Feature 3: isC1

Khi đó, bảng dữ liệu mới được mã hóa One-Hot Encoding như sau:

isA1	isA2	isA3	isB1	isB2	isC1
1	0	0	1	0	1
0	1	0	0	1	1
0	0	1	0	1	1
...	...	...	...	...	...

### 3. Mã hóa Ordinal Encoding:

Ordinal Encoding (mã hóa thứ tự) là một kỹ thuật mã hóa dữ liệu phân loại có thứ tự hoặc mức độ xếp hạng. Khác với One-Hot Encoding, phương pháp này chuyển đổi các biến phân loại có trật tự logic thành các số nguyên, thay vì tạo ra nhiều cột nhị phân.

Trong khóa luận tốt nghiệp này, thuộc tính "Education level" được mã hóa nhằm chuyển đổi dữ liệu mã hóa có tính xếp hạng, dễ dàng phục vụ cho quá trình huấn luyện và xây dựng mô hình học máy.

Cho bảng dữ liệu sau biết rằng  $A > B > C > D > \dots$ , ta có:

X	Feature
$X_1$	A
$X_2$	B
$X_3$	D
$X_4$	C
...	...

Ta gán giá trị tương ứng cho các giá trị của đặc trưng (A - 0, B - 1, C - 2, D - 3, ...). Khi đó, bảng dữ liệu mới sẽ được mã hóa Ordinal Encoding như sau:

X	Feature
$X_1$	0
$X_2$	1
$X_3$	3
$X_4$	2
...	...

### 4. Kỹ thuật Min - Max Scaling:

Min - Max Scaling là một trong những kỹ thuật chuẩn hóa dữ liệu phổ biến nhất trong học máy, được sử dụng để đưa tất cả các đặc trưng (features) về một khoảng giá trị nhất định, thường có giá trị từ 0 đến 1. Mục tiêu chính của phương pháp này là đảm bảo rằng

mọi thuộc tính trong dữ liệu đầu vào đều được thu nhỏ để có cùng thang đo, giúp cải thiện hiệu quả và độ chính xác của các thuật toán học máy.

Các thuộc tính cần được chuẩn hóa: "Age", "Income", "Employment length".

Công thức Min - Max Scaling như sau:

$$X_{scaler} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

trong đó:

- $X$  là giá trị gốc của thuộc tính đang xét
- $X_{min}$  là giá trị nhỏ nhất của thuộc tính đó trong tập dữ liệu
- $X_{max}$  là giá trị lớn nhất của thuộc tính đó trong tập dữ liệu
- $X_{scaler}$  là giá trị sau khi được chuẩn hóa Min - Max

## 5. Cân bằng dữ liệu với SMOTE:

SMOTE (Synthetic Minority Over-sampling Technique) là một kỹ thuật cân bằng dữ liệu khi bị lệch nhãn. Kỹ thuật này giúp tăng số lượng mẫu của nhóm thiểu số (minority class) bằng cách tạo ra dữ liệu tổng hợp mới dựa trên các điểm hiện có, thay vì chỉ sao chép dữ liệu cũ.

Cách thức hoạt động của SMOTE:

- (a) Xác định nhóm thiểu số (Is high risk = 1) và nhóm đa số (Is high risk = 0), trong đó SMOTE sẽ tạo thêm dữ liệu cho nhóm thiểu số mà không làm thay đổi nhóm đa số
- (b) Tạo dữ liệu tổng hợp mới bằng phương pháp nội suy:
  - Với mỗi điểm dữ liệu trong nhóm thiểu số (Is high risk = 1), SMOTE sẽ tìm ra  $k$  điểm lân cận gần nhất theo thuật toán K-Nearest Neighbors (KNN)
  - Chọn ngẫu nhiên một trong những hàng xóm này và tạo ra một điểm mới nằm trên đoạn nối giữa điểm gốc và điểm được chọn
  - Điểm mới này có giá trị nằm giữa hai điểm cũ, được nội suy bằng cách như sau:

$$X_{NewSample} = X_{minority} + \lambda(X_{neighbor} - X_{minority}) \quad \forall \lambda \in [0, 1]$$

- (c) Kết hợp dữ liệu gốc và dữ liệu mới đã tổng hợp trên để tạo thành tập dữ liệu cân bằng

### 3.3 Tiền xử lý dữ liệu (Data Preprocessing):

1. Chuẩn hóa dữ liệu dạng ngày đối với đặc trưng "Employment length" và "Age" về dạng số nguyên dương
2. Thay đổi giá trị của đặc trưng "Employment length" đối với người nghỉ hưu về 0
3. Giảm thiểu độ lệch đối với đặc trưng "Income" và "Age" bằng cách lấy căn bậc 3 giá trị nhằm để kéo chúng về gần phân phối chuẩn hơn
4. Chuyển đổi dữ liệu các đặc trưng "Has a work phone", "Has a phone", "Has an email" về dạng nhị phân (Y - 1, N - 0)
5. Sử dụng One-Hot Encoding đối với các đặc trưng phân loại: "Gender", "Marital status", "Dwelling", "Employment status", "Has a car", "Has a property", "Has a work phone", "Has a phone", "Has an email"
6. Sử dụng Ordinal Encoding cho đặc trưng "Education level" vì đặc trưng này có tính thứ tự xếp hạng
7. Chuẩn hóa các đặc trưng "Age", "Income", "Employment length" bằng Min-Max scaling để cho thuật toán thực hiện được dễ dàng đối với các giá trị nhỏ mà vẫn giữ nguyên tỷ lệ ban đầu của dữ liệu
8. Chuyển đổi các giá trị của biến mục tiêu "Is high risk" về dạng Numerical và sử dụng SMOTE để cân bằng dữ liệu

Sau quá trình tiền xử lý dữ liệu (Data Preprocessing), ta thu được bộ dữ liệu mới phù hợp dùng để training mô hình.

### 3.4 Lựa chọn và huấn luyện mô hình:

Trong đề tài khóa luận này sử dụng một số mô hình học máy như Gradient Boosting Classifier, Bagging Classifier, Logistic Regression, Decision Tree, Random Forest, ... Sau khi thử nghiệm từng mô hình trên, em đã có ý tưởng thiết kế và xây dựng được một mô hình mới và có đặt tên là Gradient Stacking Classifier.

Mô hình Gradient Stacking Classifier này được xây dựng với ba mô hình cơ sở (base learners) bao gồm Decision Tree, Random Forest, Extra Trees và đảm nhiệm vai trò học đặc trưng từ dữ liệu huấn luyện. Các mô hình này sẽ tạo ra các dự đoán ban đầu, sau đó được sử dụng làm đầu vào cho một mô hình tầng trên (meta-learner) là Gradient Boosting Classifier.

Mô hình tầng trên này học cách kết hợp các dự đoán của các mô hình cơ sở để tạo ra dự đoán cuối cùng.

### Gradient Stacking Classifier

Lớp	Precision	Recall	F1-score	Support
0	0.99	0.99	0.99	23272
1	0.99	0.99	0.99	23272
<b>Accuracy</b>			0.99	46544
<b>Macro avg</b>	0.99	0.99	0.99	46544
<b>Weighted avg</b>	0.99	0.99	0.99	46544

Kết quả đánh giá trên tập kiểm tra cho thấy mô hình Gradient Stacking Classifier đạt được độ chính xác cao hơn so với các mô hình thành phần đơn lẻ, từ đó chứng minh hiệu quả của việc kết hợp nhiều mô hình với khả năng học đa dạng trong việc cải thiện hiệu suất tổng thể của hệ thống dự đoán. Việc ứng dụng Stacking không chỉ giúp tận dụng ưu điểm của từng mô hình mà còn giảm thiểu các sai số riêng lẻ, góp phần mang lại kết quả ổn định và tin cậy hơn cho bài toán đặt ra.

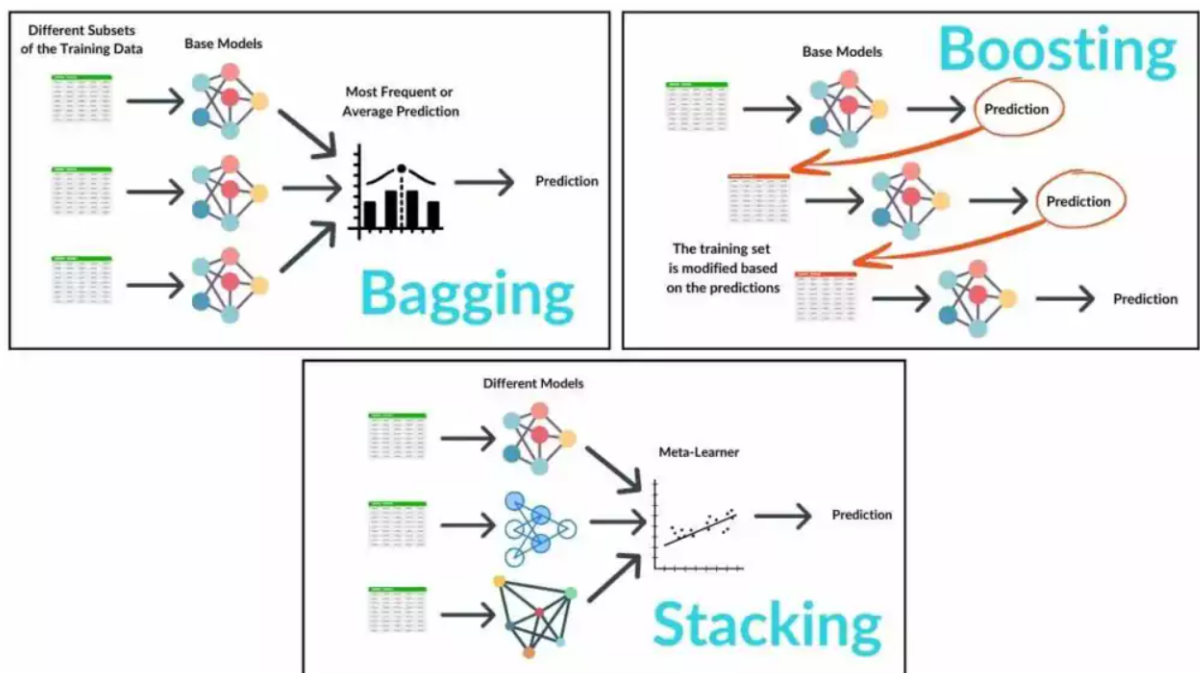
## 4 Phương pháp Ensemble Learning

Phương pháp Ensemble Learning là phương pháp kết hợp một số mô hình với nhau được sử dụng nhằm để tăng độ chính xác trên tập dữ liệu.

Ý nghĩa của việc kết hợp (Combine) các mô hình khác nhau bởi vì các mô hình khác nhau có khả năng thực hiện các công việc khác nhau (subtasks) và khi kết hợp các mô hình đó một cách hợp lý thì tạo thành Combined Model có khả năng cải thiện hiệu suất tổng thể so với việc sử dụng các mô hình đơn lẻ.

Có 3 phương pháp Ensemble Learning:

- **Bagging (đóng bao)**: xây dựng số lượng lớn các models cùng loại trên những subsamples khác nhau từ tập Training Dataset một cách song song độc lập nhằm đưa ra dự đoán tốt hơn.
- **Boosting (tăng cường)**: xây dựng số lượng lớn các models cùng loại. Tuy nhiên quá trình huấn luyện trong phương pháp này diễn ra tuần tự theo chuỗi (Sequence), tức là trong chuỗi này mỗi model sau sẽ dựa trên những sai số (Errors) của model trước.
- **Stacking (xếp chồng)**: xây dựng một số models khác loại và một mô hình Supervisor. Mô hình này sẽ học cách kết hợp kết quả dự báo của một số mô hình một cách tốt nhất.



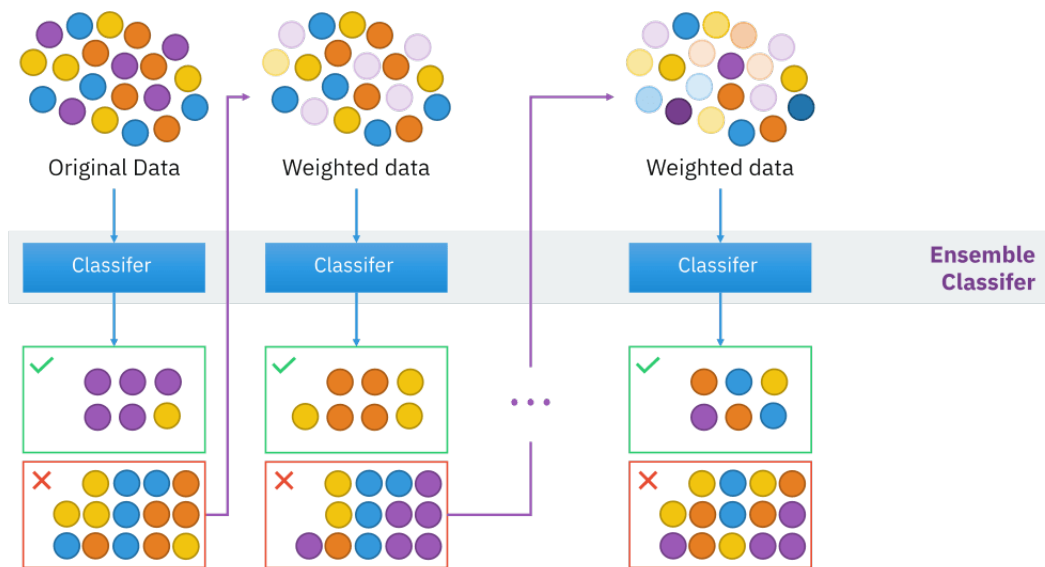
Hình 12: Một số phương pháp trong Ensemble Learning

## 4.1 Mô hình Gradient Boosting Classifier:

### 4.1.1 Khái niệm Gradient Boosting:

Gradient Boosting là một thuật toán phân loại mô hình học máy thuộc nhóm Ensemble Learning (Boosting, Bagging, Stacking) theo phương pháp Boosting. Đây là kỹ thuật kết hợp nhiều mô hình yếu (Weak Learners) để tạo ra một mô hình mạnh mẽ hơn nhằm cải thiện độ chính xác trong bài toán phân loại. Trong đó các mô hình yếu thường là cây quyết định (Decision Tree).

Thuật toán Gradient Boosting hoạt động dựa trên ý tưởng tối ưu hóa hàm mất mát (Loss Function) bằng cách sử dụng phương pháp Gradient Descent. Cụ thể, mô hình được xây dựng một cách tuần tự, trong đó mỗi mô hình mới được huấn luyện nhằm giảm sai số còn lại (Residual Error) từ các dự đoán của những mô hình trước đó. Qua từng bước, mô hình ngày càng được cải thiện chính xác hơn, hướng đến việc giảm thiểu sai số tổng thể.



Hình 13: Minh họa thuật toán Gradient Boosting

### 4.1.2 Quy trình thuật toán:

Cho bộ dữ liệu Datasets có  $X = (X_1, X_2, \dots, X_n)^T$  và  $Y = (y_1, y_2, \dots, y_n)^T$  tương ứng như sau:

$$X = \begin{pmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \dots & \dots & \dots & \dots \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{pmatrix}; Y = \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix} \in \{0, 1\}^n$$

Biết rằng bộ dữ liệu trên thỏa mãn ánh xạ:

$$f: \begin{matrix} R^{n \times m} \\ (X_1, X_2, \dots, X_n) \end{matrix} \longrightarrow \begin{matrix} R^n \\ F(X_1, X_2, \dots, X_n) \end{matrix}$$

*Chú ý: Mỗi vector  $X_i = (x_{i1}, x_{i2}, \dots, x_{im})$  đại diện cho đặc trưng  $X_i$  trong bộ dữ liệu với giá trị dự đoán  $y_i$  tương ứng.*

Các bước của thuật toán đối với mẫu dữ liệu  $X_i$ :

1. Khởi tạo mô hình cơ bản (Initial Model): mô hình dự đoán giá trị ban đầu  $F_t(X_i)$  cho tất cả các điểm dữ liệu:

$$F_0(X_i) = \log\left(\frac{\hat{p}_i^{(0)}}{1 - \hat{p}_i^{(0)}}\right) \text{ với } \hat{p}_i^{(0)} = \frac{1}{n} \sum_{j=1}^n y_j$$

2. Thực hiện các vòng lặp của thuật toán cho đến khi Loss-Function đạt GTNN: (tại bước thứ t)

- Tính toán Gradient Descent đối với điểm dữ liệu đang xét:

$$g_i^{(t)} = \frac{\partial L(y_i, F_t(X_i))}{\partial F_t(X_i)} = \hat{p}_i^{(t-1)} - y_i$$

– Với  $\hat{p}_i^{(t-1)} = \frac{1}{1 + e^{-F_{t-1}(X_i)}}$  là xác suất dự đoán tại bước t - 1

– Với  $y_i \in \{0, 1\}$  là nhãn thực tế

- Huấn luyện mô hình dạng cây quyết định  $h_t(X)$ , trong đó  $h_t(X_i)$  để dự đoán Gradient  $g_i^{(t)}$  theo phương pháp Friedman MSE như sau:

$$FriedmanMSE = \frac{1}{n_1} \sum_{i \in I_1} (g_i^{(t)} - \overline{g_1})^2 + \frac{1}{n_2} \sum_{i \in I_2} (g_i^{(t)} - \overline{g_2})^2$$

biết rằng:

- $n_1, n_2$  là số lượng mẫu tại nhánh trái và nhánh phải
- $I_1, I_2$  là chỉ số của các mẫu thuộc về nhánh con trái và phải
- $\overline{g_1}, \overline{g_2}$  là giá trị trung bình của gradient  $g_i^{(t)}$  trong mỗi nhánh trái và phải

Cách tìm điểm chia tối ưu cho cây quyết định:

- (a) Xét từng điểm chia cho các đặc trưng (m đặc trưng) trong bộ dữ liệu bằng cách lấy lần lượt trung bình hai điểm gần nhau nhất sau khi đã sắp xếp số liệu tại thuộc tính đó



- (b) Trong danh sách gồm các điểm chia đó, ta chọn điểm chia có Friedman MSE đạt GTNN
- (c) Giả sử, cây quyết định trên có điểm chia thỏa mãn thì ta sẽ lặp lại các bước trên với  $m - 1$  đặc trưng còn lại tại mỗi nhánh của cây

Sau khi thực hiện thuật toán Decision Tree theo phương pháp Friedman MSE như trên, ta tìm được cây quyết định  $h_t(X)$  với các điểm chia theo các đặc trưng như trên và  $h_t(X_i) \approx g_i^{(t)}$

- Cập nhật mô hình:

- Hàm mất mát (Loss Function):

$$L(y_i, F_t(X_i)) = -y_i \log(\hat{p}_i^{(t)}) - (1 - y_i) \log(1 - \hat{p}_i^{(t)})$$

- Hệ số tối ưu  $\gamma_t$  để cập nhật mô hình:

$$\gamma_t = \arg \min_{\gamma} \sum_{i=1}^n L(y_i, F_{t-1}(X_i) + \gamma \cdot \eta h_t(X_i))$$

- Cập nhật lại mô hình:

$$F_t(X_i) = F_{t-1}(X_i) + \eta \gamma_t h_t(X_i) \quad \forall \eta \in (0, 1)$$

=> Dự đoán xác suất của mẫu  $i$  tại bước  $t$  là  $\hat{p}_i^{(t)} = \frac{1}{1 + e^{-F_t(X_i)}}$

3. Thuật toán dừng khi  $|L(y_i, F_t(X_i)) - L(y_i, F_{t-1}(X_i))| < \epsilon$  với  $\epsilon$  là ngưỡng hội tụ
4. Sau  $t$  vòng lặp thì mô hình cuối cùng  $F_t(X_i)$  có thể dùng để dự đoán xác suất và phân loại đối với mẫu  $i$ :

$$\hat{y}_i = \begin{cases} 1 & \text{nếu } \hat{p}_i^{(t)} \geq 0.5 \\ 0 & \text{nếu ngược lại} \end{cases}$$

### 4.1.3 Mô hình Gradient Boosting:

Trong quá trình huấn luyện mô hình, thuật toán Gradient Boosting sẽ giúp mô hình tìm được hệ số tối ưu  $\gamma = \gamma_t$  nhằm có cơ sở để dự đoán chính xác hơn. Hệ số này đóng vai trò quan trọng trong việc điều chỉnh mức độ ảnh hưởng của từng cây quyết định vào mô hình tổng thể.

Khi đó, mô hình của thuật toán GBC:

$$F(X_i) = F_0(X_i) + \gamma \cdot \sum_{t=1}^T \eta h_t(X_i) \quad \forall \eta \in (0, 1)$$

trong đó:

- Với  $F_0(X_i) = \log(\frac{\bar{p}}{1-\bar{p}})$  và  $\bar{p} = \frac{1}{n} \sum_{j=1}^n y_j$
- T là số vòng lặp tối đa của thuật toán với cây quyết định  $h_t(X_i)$
- Dự đoán giá trị bằng hàm Sigmoid là  $\hat{y}_i = \frac{1}{1+e^{-F(X_i)}} = \begin{cases} 1 & \text{nếu } \hat{y}_i \geq 0.5 \\ 0 & \text{nếu ngược lại} \end{cases}$

#### 4.1.4 Ví dụ minh họa:

Cho bộ dữ liệu Dataset gồm 5 khách hàng ( $n = 5, m = 4$ ) tương ứng như sau:

$$X = \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \end{pmatrix} = \begin{pmatrix} 1 & 3 & 2 & 8 \\ 2 & 1 & 1 & 5 \\ 3 & 2 & 4 & 6 \\ 4 & 5 & 3 & 2 \\ 5 & 1 & 2 & 7 \end{pmatrix}; Y = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix} \in \{0, 1\}^5$$

Các bước thực hiện thuật toán đối với mẫu dữ liệu:

1. Khởi tạo mô hình cơ bản cho tất cả các điểm dữ liệu: ( $i \in [1, 5]$ )

- $\hat{p}_i^{(0)} = \frac{1}{5} \sum_{j=1}^5 y_j = \frac{0+0+1+1+1}{5} = \frac{3}{5} = 0.6$
- $F_0(X_i) = \log(\frac{\hat{p}_i^{(0)}}{1-\hat{p}_i^{(0)}}) = \log(\frac{0.6}{1-0.6}) = 0.4$

$X_i$	$y_i$	$p_i^{(0)}$	$F_0(X_i)$
$X_1$	0	0.6	0.4
$X_2$	0	0.6	0.4
$X_3$	1	0.6	0.4
$X_4$	1	0.6	0.4
$X_5$	1	0.6	0.4

2. Thực hiện các vòng lặp của thuật toán:

- Tính xác suất dự đoán tại bước 0:

$$\hat{p}_i^{(0)} = \frac{1}{1 + e^{-F_0(X_i)}} = \frac{1}{1 + e^{-0.4}} = 0.6$$

- Tính toán Gradient Descent đối với các điểm dữ liệu:

$$g_i^{(1)} = \hat{p}_i^{(0)} - y_i = 0.6 - y_i$$

i	$y_i$	$g_i^{(1)} = \hat{p}_i^{(0)} - y_i$
1	0	$0.6 - 0 = 0.6$
2	0	$0.6 - 0 = 0.6$
3	1	$0.6 - 1 = -0.4$
4	1	$0.6 - 1 = -0.4$
5	1	$0.6 - 1 = -0.4$

- Huấn luyện mô hình mới dạng cây quyết định  $h_1(X)$  và ta có giá trị  $h_1(X_i)$  nhằm dự đoán Gradient  $g_i^{(1)}$ . Giả sử, theo như phương pháp Friedman MSE thì có thể chọn được điểm chia theo đây đủ các thuộc tính như sau:

i	$y_i$	$g_i^{(1)} = \hat{p}_i^{(0)} - y_i$	$h_1(X_i)$
1	0	0.6	0.6
2	0	0.6	0.6
3	1	-0.4	-0.4
4	1	-0.4	-0.4
5	1	-0.4	-0.4

- Hàm mất mát:

$$L(y_i, F_t(X_i)) = -y_i \log(\hat{p}_i^{(t)}) - (1 - y_i) \log(1 - \hat{p}_i^{(t)})$$

i	$y_i$	$p_i^{(0)}$	$L(y_i, F_0(X_i))$
1	0	0.6	0.9
2	0	0.6	0.9
3	1	0.6	0.5
4	1	0.6	0.5
5	1	0.6	0.5

- Hệ số tối ưu  $\gamma_1$  để cập nhật mô hình thỏa mãn:

$$\gamma_1 = \arg \min_{\gamma} \sum_{i=1}^n L(y_i, F_0(X_i) + \gamma \cdot \eta h_1(X_i))$$

Giả sử, ta chọn được hệ số tối ưu như sau:

$$\gamma_1 = \sum_{i=1}^5 L(y_i, F_0(X_i)) = 0.9 + 0.9 + 0.5 + 0.5 + 0.5 = 3.3$$

- Cập nhật mô hình:

$$F_1(X_i) = F_0(X_i) + \eta \gamma_1 h_1(X_i) \quad \forall \eta = 0.8$$

i	$\gamma_1$	$F_0(X_i)$	$h_1(X_i)$	$F_1(X_i)$
1	3.3	0.4	0.6	$0.4 + 0.8 \times 3.3 \times 0.6 = 1.984$
2	3.3	0.4	0.6	$0.4 + 0.8 \times 3.3 \times 0.6 = 1.984$
3	3.3	0.4	-0.4	$0.4 - 0.8 \times 3.3 \times 0.4 = -0.656$
4	3.3	0.4	-0.4	$0.4 - 0.8 \times 3.3 \times 0.4 = -0.656$
5	3.3	0.4	-0.4	$0.4 - 0.8 \times 3.3 \times 0.4 = -0.656$

3. Khi đó, ta có được dự đoán cuối cùng của mô hình:

$X_i$	$y_i$	$F_1(X_i)$	$p_i^{(1)} = \frac{1}{1+e^{-F_1(X_i)}}$	$\hat{y}_i$
$X_1$	0	1.984	0.9	0
$X_2$	0	1.984	0.9	0
$X_3$	1	-0.656	0.3	1
$X_4$	1	-0.656	0.3	1
$X_5$	1	-0.656	0.3	1

Ta có mô hình mới  $F_1(X_i)$  có thể dùng để dự đoán xác suất và phân loại đối với mẫu i:

$$\hat{y}_i = \begin{cases} 1 & \text{nếu } \hat{p}_i^{(1)} \geq 0.5 \\ 0 & \text{nếu ngược lại} \end{cases}$$

4. Kiểm tra tính hội tụ của thuật toán với ngưỡng hội tụ cho trước  $\epsilon = 0.01$  như sau:

$$L = |L(y_i, F_1(X_i)) - L(y_i, F_0(X_i))| < \epsilon$$

$X_i$	$y_i$	$L(y_i, F_0(X_i))$	$L(y_i, F_1(X_i))$	$L$
$X_1$	0	0.9	2.3	1.4
$X_2$	0	0.9	2.3	1.4
$X_3$	1	0.5	1.2	0.7
$X_4$	1	0.5	1.2	0.7
$X_5$	1	0.5	1.2	0.7

5. Lặp tại tất cả các bước làm trên cho đến khi tìm được hệ số tối ưu  $\gamma_t$  và số vòng lặp phù hợp để tìm ra được mô hình dự đoán chính xác nhất. Khi đó, ta có thể dự đoán cho dữ liệu mới dựa trên mô hình vừa xây dựng trên.

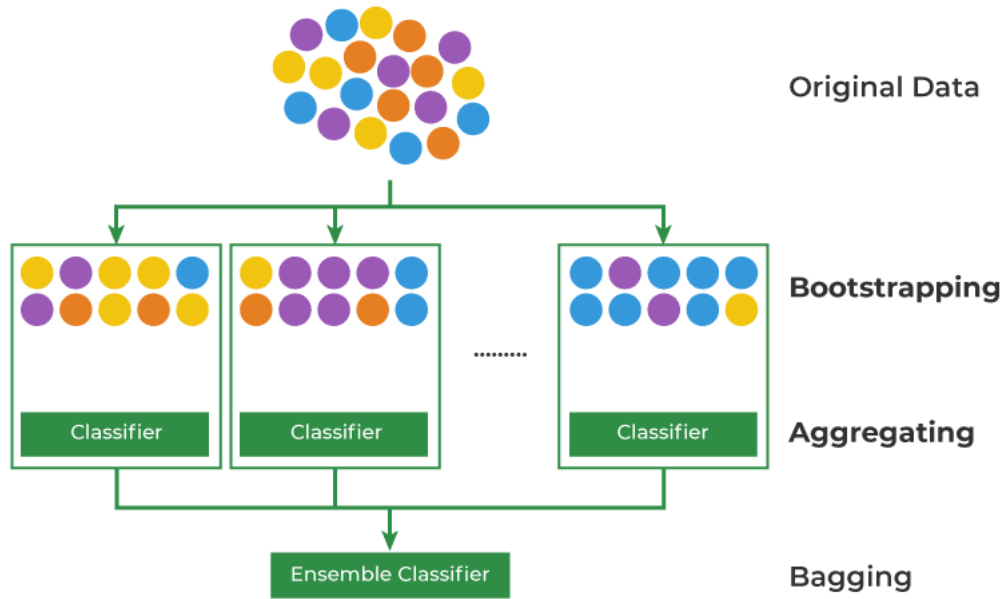
## 4.2 Mô hình Bagging Classifier:

### 4.2.1 Khái niệm Bootstrap Aggregating:

Bootstrap Aggregating (Bagging) là một trong những phương pháp phổ biến của nhóm học máy Ensemble Learning – học kết hợp (Boosting,

Bagging, Stacking), trong đó các mô hình thường là cây quyết định (Decision Trees), được huấn luyện trên các tập con khác nhau của tập dữ liệu gốc, và kết quả cuối cùng được xác định dựa trên biểu quyết đa số (voting) (đối với phân loại) hoặc trung bình (averaging) (đối với hồi quy).

Bagging hoạt động dựa trên nguyên tắc huấn luyện nhiều mô hình yếu (Weak Learners) một cách độc lập trên các tập dữ liệu khác nhau từ bộ dữ liệu cho trước, sau đó tổng hợp kết quả để tạo ra mô hình mạnh hơn.



Hình 14: Minh họa thuật toán Bagging (Bootstrap Aggregating)

#### 4.2.2 Quy trình thuật toán:

Cho bộ dữ liệu Datasets có  $X = (X_1, X_2, \dots, X_n)^T$  và  $Y = (y_1, y_2, \dots, y_n)^T$  tương ứng như sau:

$$X = \begin{pmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \dots & \dots & \dots & \dots \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{pmatrix}; Y = \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix} \in \{0, 1\}^n$$

Biết rằng bộ dữ liệu trên thỏa mãn ánh xạ:

$$f: \begin{matrix} R^{n \times m} \\ (X_1, X_2, \dots, X_n) \end{matrix} \longrightarrow \begin{matrix} R^n \\ F(X_1, X_2, \dots, X_n) \end{matrix}$$

*Chú ý:* Mỗi vector  $X_i = (x_{i1}, x_{i2}, \dots, x_{im})$  đại diện cho đặc trưng  $X_i$  trong bộ dữ liệu với giá trị dự đoán  $y_i$  tương ứng.

Từ bộ dữ liệu ban đầu, ta tạo ra  $T = 10$  tập dữ liệu con mới ngẫu nhiên tương ứng với 10 cây quyết định, trong đó mỗi cây quyết định đều chứa  $K$  điểm dữ liệu trong không gian. Khi đó, mỗi cây quyết định sẽ được thực hiện theo thuật toán Decision Tree như sau:

1. Với tập dữ liệu thứ  $t$  ( $1 \leq t \leq T$ ), ta có:

$$X_{Kt} = \begin{pmatrix} X_{i+1} \\ X_{i+2} \\ \dots \\ X_{i+K} \end{pmatrix} = \begin{pmatrix} x_{(i+1)1} & x_{(i+1)2} & \dots & x_{(i+1)m} \\ x_{(i+2)1} & x_{(i+2)2} & \dots & x_{(i+2)m} \\ \dots & \dots & \dots & \dots \\ x_{(i+K)1} & x_{(i+K)2} & \dots & x_{(i+K)m} \end{pmatrix}; Y_{Kt} = \begin{pmatrix} y_{i+1} \\ y_{i+2} \\ \dots \\ y_{i+K} \end{pmatrix}$$

biết rằng  $Y_{Kt} \in \{0, 1\}^K$

2. Huấn luyện mô hình  $t$  thành dạng cây quyết định (Decision Tree) dựa trên tập dữ liệu thứ  $t$  bằng phương pháp Gini Impurity:

$$Gini(D_{j;branch}) = 1 - \sum_{i=1}^C p_i^2$$

với  $C = 2$  là số lớp và  $p_i$  là tỷ lệ mẫu chung của từng lớp

- Xét từng điểm chia cho các đặc trưng ( $m$  đặc trưng) trong tập dữ liệu thứ  $t$  bằng cách lấy lần lượt trung bình giữa hai điểm gần nhau nhất sau khi đã sắp xếp số liệu
- Trong danh sách gồm các điểm chia, ta chọn điểm chia có Total Gini đạt GTNN thỏa mãn:

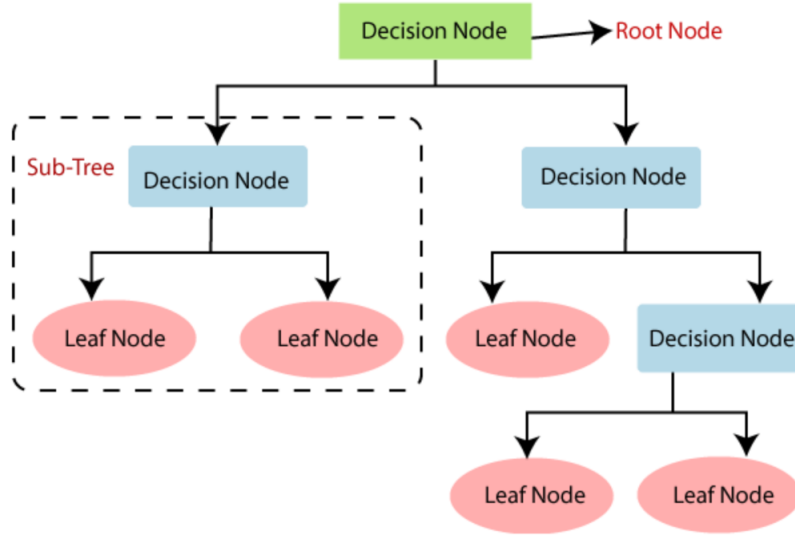
$$GiniTotal(D_j) = \frac{n}{k} * Gini(D_{j;right}) + \frac{k-n}{k} * Gini(D_{j;left})$$

trong đó:

- $n$  là số điểm được chia ở nhánh phải
- $k$  là số điểm tổng của cả hai nhánh thỏa mãn  $0 \leq k \leq K$
- Đặc trưng thứ  $j$  thỏa mãn  $1 \leq j \leq m$
- Giả sử, cây quyết định trên có điểm chia thỏa mãn thì ta sẽ lặp lại các bước trên với  $m - 1$  đặc trưng còn lại tại mỗi nhánh của cây

3. Sau khi thực hiện thuật toán Decision Tree như trên, ta tìm được cây quyết định thứ  $t$  với các điểm chia theo các đặc trưng như trên ta gọi là  $h_t(X_{Kt})$ . Khi đó, ta có  $T = 10$  cây quyết định có các giá trị như sau:

$$DecisionTree(X_{Kt}) = (h_1(X_{K1}), h_2(X_{K2}), \dots, h_T(X_{KT}))$$



Hình 15: Minh hoạ thuật toán Decision Tree

### 4.2.3 Mô hình Bagging:

Với các mô hình cây quyết định Decision Tree như trên, ta có thể dự đoán điểm dữ liệu mới  $X_{new} = (x_1, x_2, \dots, x_m)$  thuộc phân lớp rủi ro nào theo ánh xạ:

$$f: \begin{matrix} R^m \\ X_{new} \end{matrix} \longrightarrow \begin{matrix} R^T \\ DecisionTree(X_{new}) \end{matrix}$$

thỏa mãn  $DecisionTree(X_{new}) = (h_1(X_{new}), h_2(X_{new}), \dots, h_T(X_{new}))$  với  $T = 10$  và có giá trị dự đoán là  $y_{new}$ , trong đó:

$$y_{new} = \begin{cases} 1 & \text{nếu có nhiều hơn số lớp 1 trong } DecisionTree(X_{new}) \\ 0 & \text{nếu có nhiều hơn số lớp 0 trong } DecisionTree(X_{new}) \end{cases}$$

### 4.2.4 Ví dụ minh họa:

Cho bộ dữ liệu Dataset gồm 5 khách hàng ( $n = 5$ ,  $m = 4$ ) tương ứng như sau:

$$X = \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \end{pmatrix} = \begin{pmatrix} 1 & 3 & 2 & 8 \\ 2 & 1 & 1 & 5 \\ 3 & 2 & 4 & 6 \\ 4 & 5 & 3 & 2 \\ 5 & 1 & 2 & 7 \end{pmatrix}; Y = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix} \in \{0, 1\}^5$$

Các bước thực hiện thuật toán đối với mẫu dữ liệu:

1. Giả sử, ta cần tạo ra  $T = 3$  cây quyết định, trong đó mỗi cây sử dụng  $K = 3$  điểm dữ liệu ngẫu nhiên từ bộ dữ liệu ban đầu như sau:

- Tập con  $X_{K1}$  và nhãn  $Y_{K1}$ :

$$X_{K1} = \begin{pmatrix} 1 & 3 & 2 & 8 \\ 3 & 2 & 4 & 6 \\ 5 & 1 & 2 & 7 \end{pmatrix}; Y_{K1} = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$$

- Tập con  $X_{K2}$  và nhãn  $Y_{K2}$ :

$$X_{K2} = \begin{pmatrix} 2 & 1 & 1 & 5 \\ 4 & 5 & 3 & 2 \\ 3 & 2 & 4 & 6 \end{pmatrix}; Y_{K2} = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$$

- Tập con  $X_{K3}$  và nhãn  $Y_{K3}$ :

$$X_{K3} = \begin{pmatrix} 5 & 1 & 2 & 7 \\ 2 & 1 & 1 & 5 \\ 4 & 5 & 3 & 2 \end{pmatrix}; Y_{K3} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

2. Huấn luyện mô hình dạng cây quyết định theo phương pháp Gini Impurity nhằm chọn điểm chia tối ưu. Giả sử, ta chọn điểm chia như sau:

- Tập con  $X_{K1}$  và nhãn  $Y_{K1}$ :

$$h_1(X_{K1}) = \begin{cases} 0 & \text{nếu col1} < 3 \\ 1 & \text{nếu col1} \geq 3 \end{cases}$$

- Tập con  $X_{K2}$  và nhãn  $Y_{K2}$ :

$$h_2(X_{K2}) = \begin{cases} 0 & \text{nếu col2} < 2 \\ 1 & \text{nếu col2} \geq 2 \end{cases}$$

- Tập con  $X_{K3}$  và nhãn  $Y_{K3}$ :

$$h_3(X_{K3}) = \begin{cases} 0 & \text{nếu col3} < 2 \\ 1 & \text{nếu col3} \geq 2 \end{cases}$$

3. Dự đoán các điểm dữ liệu trên với cả 3 cây quyết định:

- $X_1 = (1, 3, 2, 8)$  có giá trị dự đoán cây quyết định là:

$$DecisionTree(X_1) = (0, 1, 1) \longrightarrow \hat{y}_1 = 1$$

- $X_2 = (2, 1, 1, 5)$  có giá trị dự đoán cây quyết định là:

$$DecisionTree(X_2) = (0, 0, 0) \longrightarrow \hat{y}_2 = 0$$



- $X_3 = (3, 2, 4, 6)$  có giá trị dự đoán cây quyết định là:

$$DecisionTree(X_3) = (1, 1, 1) \rightarrow \hat{y}_3 = 1$$

- $X_4 = (4, 5, 3, 2)$  có giá trị dự đoán cây quyết định là:

$$DecisionTree(X_4) = (1, 1, 1) \rightarrow \hat{y}_4 = 1$$

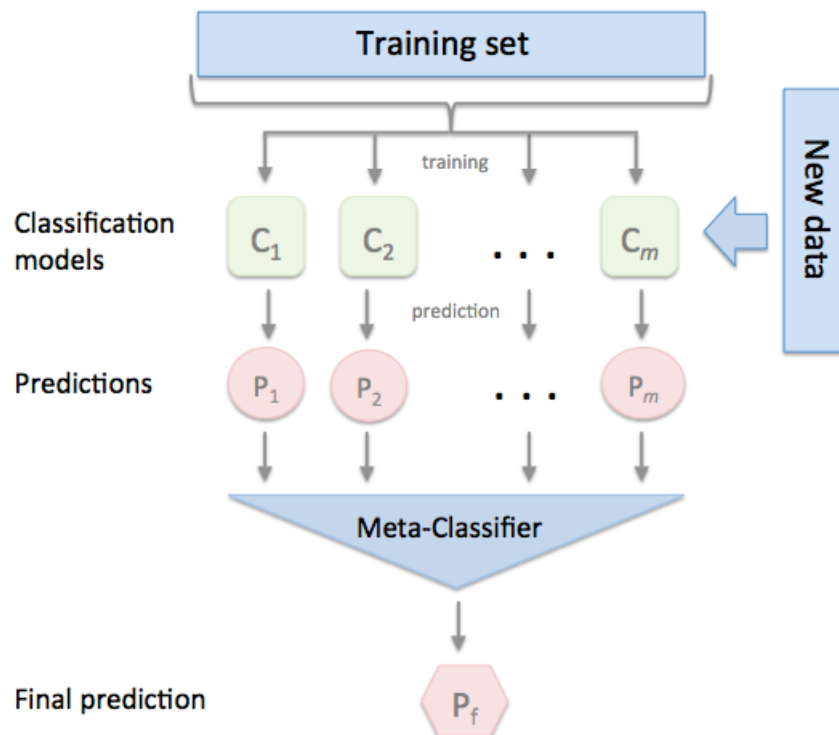
- $X_5 = (5, 1, 2, 7)$  có giá trị dự đoán cây quyết định là:

$$DecisionTree(X_5) = (1, 0, 1) \rightarrow \hat{y}_5 = 1$$

4. Tương tự, ta có thể dự đoán dữ liệu mới dựa trên mô hình Bagging Classifier vừa xây dựng ở phía trên.

### 4.3 Mô hình Stacking Classifier:

Stacking Classifier là một phương pháp học tập kết hợp mạnh mẽ trong Ensemble Learning (Boosting, Bagging, Stacking). Trong đó, nhiều mô hình khác loại được huấn luyện độc lập với nhau và có mô hình tổng hợp sẽ kết hợp kết quả của chúng để tạo ra dự đoán chính xác hơn nữa.



Hình 16: Minh hoạt các thuật toán trong Stacking

Stacking Classifier khai thác được sức mạnh của nhiều mô hình bằng cách kết hợp đa dạng các thuật toán giúp tận dụng tối đa ưu điểm của từng mô hình. Ngoài ra, so với các mô hình đơn lẻ thì Stacking còn giúp giảm nhiều sai số và cải thiện hiệu suất kể cả với các mô hình tuyến tính hay phi tuyến.

### 4.3.1 Mô hình tuyến tính Logistic Regression:

Cho bộ dữ liệu Datasets có  $X = (X_1, X_2, \dots, X_n)^T$  và  $Y = (y_1, y_2, \dots, y_n)^T$  tương ứng như sau:

$$X = \begin{pmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \dots & \dots & \dots & \dots \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{pmatrix}; Y = \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix} \in \{0, 1\}^n$$

Biết rằng bộ dữ liệu trên thỏa mãn ánh xạ:

$$f: \begin{matrix} R^{n \times m} \\ (X_1, X_2, \dots, X_n) \end{matrix} \longrightarrow \begin{matrix} R^n \\ F(X_1, X_2, \dots, X_n) \end{matrix}$$

*Chú ý: Mỗi vector  $X_i = (x_{i1}, x_{i2}, \dots, x_{im})$  đại diện cho đặc trưng  $X_i$  trong bộ dữ liệu với giá trị dự đoán  $y_i$  tương ứng.*

Mô hình Logistic Regression:

$$\hat{y}_i = \frac{1}{1 + e^{-z_i}} \quad \text{với} \quad z_i = X_i w^T + b = X_i \begin{pmatrix} w_1 \\ w_2 \\ \dots \\ w_m \end{pmatrix} + b$$

Hàm mất mát Log-Likelihood:

$$L(w, b) = - \sum_{i=1}^n [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

Ta cần tối ưu hóa hàm mất mát  $L(w, b)$  để tìm vector trọng số  $w$  và bias  $b$  phù hợp cho mô hình Logistic Regression với  $j \in [1, m]$ :

1. Tính Gradient của hàm mất mát với từng trọng số  $w_j$  và bias  $b$ :

$$\frac{\partial L}{\partial w_j} = \sum_{i=1}^n x_{ij}(\hat{y}_i - y_i); \quad \frac{\partial L}{\partial b} = \sum_{i=1}^n (\hat{y}_i - y_i)$$

2. Cập nhật lại trọng số của ma trận với  $\eta \in (0, 1)$ :

$$w_j \longleftarrow w_j - \eta \frac{\partial L}{\partial w_j}; \quad b \longleftarrow b - \eta \frac{\partial L}{\partial b}$$

3. Thuật toán dừng khi vector trọng số  $w$  và bias  $b$  không có thay đổi gì đáng kể, ta thu được mô hình Logistic Regression mới có dạng:

$$f(X_i) = \frac{1}{1 + e^{-z_i}} \quad \text{với} \quad z_i = X_i w^T + b$$

### 4.3.2 Mô hình cây quyết định Decision Tree:

Cho bộ dữ liệu Datasets có  $X = (X_1, X_2, \dots, X_n)^T$  và  $Y = (y_1, y_2, \dots, y_n)^T$  tương ứng như sau:

$$X = \begin{pmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \dots & \dots & \dots & \dots \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{pmatrix}; Y = \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix} \in \{0, 1\}^n$$

Biết rằng bộ dữ liệu trên thỏa mãn ánh xạ:

$$f: \begin{matrix} R^{n \times m} \\ (X_1, X_2, \dots, X_n) \end{matrix} \longrightarrow \begin{matrix} R^n \\ F(X_1, X_2, \dots, X_n) \end{matrix}$$

*Chú ý: Mỗi vector  $X_i = (x_{i1}, x_{i2}, \dots, x_{im})$  đại diện cho đặc trưng  $X_i$  trong bộ dữ liệu với giá trị dự đoán  $y_i$  tương ứng.*

Các bước của thuật toán đối với bộ dữ liệu:

1. Huấn luyện mô hình dạng cây quyết định (Decision Tree) dựa trên bộ dữ liệu bằng phương pháp Gini Impurity:

$$Gini(D_{j;branch}) = 1 - \sum_{i=1}^C p_i^2$$

với  $C = 2$  là số lớp và  $p_i$  là tỷ lệ mẫu chung của từng lớp

- Xét từng điểm chia cho các đặc trưng (m đặc trưng) trong bộ dữ liệu bằng cách lấy lần lượt trung bình giữa hai điểm gần nhau nhất sau khi đã sắp xếp số liệu
- Trong danh sách gồm các điểm chia, ta chọn điểm chia có Total Gini đạt GTNN thỏa mãn:

$$GiniTotal(D_j) = \frac{r}{n} * Gini(D_{j;right}) + \frac{n-r}{n} * Gini(D_{j;left})$$

trong đó:

- $r$  là số điểm được chia ở nhánh phải
- Đặc trưng thứ  $j$  thỏa mãn  $1 \leq j \leq m$
- Giả sử, cây quyết định trên có điểm chia thỏa mãn thì ta sẽ lặp lại các bước trên với  $m - 1$  đặc trưng còn lại tại mỗi nhánh của cây quyết định

2. Sau khi thực hiện thuật toán Decision Tree như trên, ta tìm được cây quyết định của bộ dữ liệu với các điểm chia theo các đặc trưng như trên gọi là  $DecisionTree(X)$ .

### 4.3.3 Mô hình rừng ngẫu nhiên Random Forest:

Về cơ bản, Random Forest là một dạng đặc biệt của Bagging Classifier, tuy nhiên tại bước chọn điểm chia tối ưu thì thuật toán sẽ chọn ngẫu nhiên tập con các đặc trưng để chia nhánh cho cây.

Cho bộ dữ liệu Datasets có  $X = (X_1, X_2, \dots, X_n)^T$  và  $Y = (y_1, y_2, \dots, y_n)^T$  tương ứng như sau:

$$X = \begin{pmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \dots & \dots & \dots & \dots \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{pmatrix}; Y = \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix} \in \{0, 1\}^n$$

Biết rằng bộ dữ liệu trên thỏa mãn ánh xạ:

$$f: \begin{matrix} R^{n \times m} \\ (X_1, X_2, \dots, X_n) \end{matrix} \longrightarrow \begin{matrix} R^n \\ F(X_1, X_2, \dots, X_n) \end{matrix}$$

*Chú ý: Mỗi vector  $X_i = (x_{i1}, x_{i2}, \dots, x_{im})$  đại diện cho đặc trưng  $X_i$  trong bộ dữ liệu với giá trị dự đoán  $y_i$  tương ứng.*

Từ bộ dữ liệu ban đầu, ta tạo ra T tập dữ liệu con mới ngẫu nhiên tương ứng với T cây quyết định, trong đó mỗi cây quyết định đều chứa K điểm dữ liệu trong không gian. Khi đó, mỗi cây quyết định sẽ được thực hiện theo thuật toán Decision Tree như sau:

1. Với tập dữ liệu thứ t ( $1 \leq t \leq T$ ), ta có:

$$X_{Kt} = \begin{pmatrix} X_{i+1} \\ X_{i+2} \\ \dots \\ X_{i+K} \end{pmatrix} = \begin{pmatrix} x_{(i+1)1} & x_{(i+1)2} & \dots & x_{(i+1)m} \\ x_{(i+2)1} & x_{(i+2)2} & \dots & x_{(i+2)m} \\ \dots & \dots & \dots & \dots \\ x_{(i+K)1} & x_{(i+K)2} & \dots & x_{(i+K)m} \end{pmatrix}; Y_{Kt} = \begin{pmatrix} y_{i+1} \\ y_{i+2} \\ \dots \\ y_{i+K} \end{pmatrix}$$

biết rằng  $Y_{Kt} \in \{0, 1\}^K$

2. Huấn luyện mô hình t thành dạng cây quyết định (Decision Tree) dựa trên tập dữ liệu thứ t bằng phương pháp Gini Impurity:

$$Gini(D_{j;branch}) = 1 - \sum_{i=1}^C p_i^2$$

với  $C = 2$  là số lớp và  $p_i$  là tỷ lệ mẫu chung của từng lớp

- Chọn ngẫu nhiên  $m' < m$  đặc trưng để tìm điểm chia tối ưu trong tập dữ liệu thứ t bằng cách lấy lần lượt trung bình giữa hai điểm gần nhau nhất sau khi đã sắp xếp số liệu

- Trong danh sách gồm các điểm chia, ta chọn điểm chia có Total Gini đạt GTNN thỏa mãn:

$$GiniTotal(D_j) = \frac{n}{k} * Gini(D_{j;right}) + \frac{k-n}{k} * Gini(D_{j;left})$$

trong đó:

- n là số điểm được chia ở nhánh phải
- k là số điểm tổng của cả hai nhánh thỏa mãn  $0 \leq k \leq K$
- Đặc trưng thứ j thỏa mãn  $1 \leq j \leq m$
- Giả sử, cây quyết định trên có điểm chia thỏa mãn thì ta sẽ lặp lại các bước trên với m' - 1 đặc trưng còn lại tại mỗi nhánh của cây

3. Sau khi thực hiện thuật toán Decision Tree như trên, ta tìm được cây quyết định thứ t với các điểm chia theo các đặc trưng như trên ta gọi là  $h_t(X_{Kt})$ . Khi đó, ta có T cây quyết định như sau:

$$DecisionTree(X_{Kt}) = (h_1(X_{K1}), h_2(X_{K2}), \dots, h_T(X_{KT}))$$

4. Với các mô hình cây quyết định Decision Tree như trên, ta có thể dự đoán điểm dữ liệu mới  $X_{new} = (x_1, x_2, \dots, x_m)$  thuộc phân lớp rủi ro nào theo ánh xạ:

$$f: \begin{matrix} R^m \\ X_{new} \end{matrix} \longrightarrow \begin{matrix} R^T \\ DecisionTree(X_{new}) \end{matrix}$$

thỏa mãn  $DecisionTree(X_{new}) = (h_1(X_{new}), h_2(X_{new}), \dots, h_T(X_{new}))$  với  $T = 10$  và có giá trị dự đoán là  $y_{new}$ , trong đó:

$$y_{new} = \begin{cases} 1 & \text{nếu có nhiều hơn số lớp 1 trong } DecisionTree(X_{new}) \\ 0 & \text{nếu có nhiều hơn số lớp 0 trong } DecisionTree(X_{new}) \end{cases}$$

#### 4.3.4 Mô hình ExtraTrees Classifier:

Mô hình ExtraTrees Classifier là 1 dạng đặc biệt của Random Forest, tuy nhiên sau khi chọn một trong số các thuộc tính bất kỳ thì ta sẽ chọn ngẫu nhiên điểm chia cho cây.

Cho bộ dữ liệu Datasets có  $X = (X_1, X_2, \dots, X_n)^T$  và  $Y = (y_1, y_2, \dots, y_n)^T$  tương ứng như sau:

$$X = \begin{pmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \dots & \dots & \dots & \dots \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{pmatrix}; Y = \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix} \in \{0, 1\}^n$$

Biết rằng bộ dữ liệu trên thỏa mãn ánh xạ:

$$f: \begin{matrix} R^{n \times m} \\ (X_1, X_2, \dots, X_n) \end{matrix} \longrightarrow \begin{matrix} R^n \\ F(X_1, X_2, \dots, X_n) \end{matrix}$$

*Chú ý: Mỗi vector  $X_i = (x_{i1}, x_{i2}, \dots, x_{im})$  đại diện cho đặc trưng  $X_i$  trong bộ dữ liệu với giá trị dự đoán  $y_i$  tương ứng.*

Từ bộ dữ liệu ban đầu, ta tạo ra T tập dữ liệu con mới ngẫu nhiên tương ứng với T cây quyết định, trong đó mỗi cây quyết định đều chứa K điểm dữ liệu trong không gian. Khi đó, mỗi cây quyết định sẽ được thực hiện theo thuật toán Decision Tree như sau:

1. Với tập dữ liệu thứ t ( $1 \leq t \leq T$ ), ta có:

$$X_{Kt} = \begin{pmatrix} X_{i+1} \\ X_{i+2} \\ \dots \\ X_{i+K} \end{pmatrix} = \begin{pmatrix} x_{(i+1)1} & x_{(i+1)2} & \dots & x_{(i+1)m} \\ x_{(i+2)1} & x_{(i+2)2} & \dots & x_{(i+2)m} \\ \dots & \dots & \dots & \dots \\ x_{(i+K)1} & x_{(i+K)2} & \dots & x_{(i+K)m} \end{pmatrix}; Y_{Kt} = \begin{pmatrix} y_{i+1} \\ y_{i+2} \\ \dots \\ y_{i+K} \end{pmatrix}$$

biết rằng  $Y_{Kt} \in \{0, 1\}^K$

2. Huấn luyện mô hình t thành dạng cây quyết định (Decision Tree) dựa trên tập dữ liệu thứ t như sau:

- Chọn ngẫu nhiên  $m' < m$  đặc trưng để tìm một số điểm chia ngẫu nhiên trong tập dữ liệu thứ t
- Trong danh sách gồm các điểm chia, ta chọn điểm chia có Total Gini đạt GTNN thỏa mãn:

$$GiniTotal(D_j) = \frac{n}{k} * Gini(D_{j:right}) + \frac{k-n}{k} * Gini(D_{j:left})$$

trong đó:

- n là số điểm được chia ở nhánh phải
- k là số điểm tổng của cả hai nhánh thỏa mãn  $0 \leq k \leq K$
- Đặc trưng thứ j thỏa mãn  $1 \leq j \leq m$
- Giả sử, cây quyết định trên có điểm chia thỏa mãn thì ta sẽ lặp lại các bước trên với  $m' - 1$  đặc trưng còn lại tại mỗi nhánh của cây

3. Sau khi thực hiện thuật toán Decision Tree như trên, ta tìm được cây quyết định thứ t với các điểm chia theo các đặc trưng như trên ta gọi là  $h_t(X_{Kt})$ . Khi đó, ta có T cây quyết định như sau:

$$DecisionTree(X_{Kt}) = (h_1(X_{K1}), h_2(X_{K2}), \dots, h_T(X_{KT}))$$

4. Với các mô hình cây quyết định Decision Tree như trên, ta có thể dự đoán điểm dữ liệu mới  $X_{new} = (x_1, x_2, \dots, x_m)$  thuộc phân lớp rủi ro nào theo ánh xạ:

$$f: \begin{matrix} R^m \\ X_{new} \end{matrix} \longrightarrow \begin{matrix} R^T \\ DecisionTree(X_{new}) \end{matrix}$$

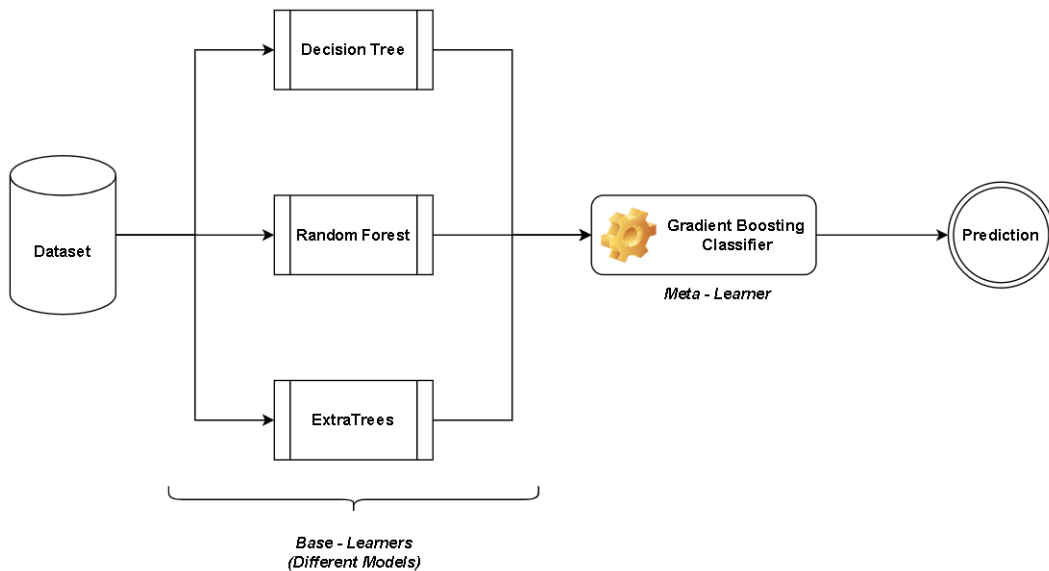
thỏa mãn  $DecisionTree(X_{new}) = (h_1(X_{new}), h_2(X_{new}), \dots, h_T(X_{new}))$  với  $T = 10$  và có giá trị dự đoán là  $y_{new}$ , trong đó:

$$y_{new} = \begin{cases} 1 & \text{nếu có nhiều hơn số lớp 1 trong } DecisionTree(X_{new}) \\ 0 & \text{nếu có nhiều hơn số lớp 0 trong } DecisionTree(X_{new}) \end{cases}$$

## 4.4 Mô hình Gradient Stacking Classifier:

### 4.4.1 Khái niệm Gradient Stacking Classifier:

Mô hình Gradient Stacking Classifier là một phương pháp học tập kết hợp mạnh mẽ trong Ensemble Learning thuộc nhóm Stacking. Trong đó, nhiều mô hình phi tuyến khác loại dạng Decision Tree, Random Forest, ExtraTrees được huấn luyện độc lập với nhau và có mô hình tổng hợp Gradient Boosting Classifier sẽ kết hợp kết quả của chúng để tạo ra dự đoán chính xác hơn nữa.



Hình 17: Minh họa thuật toán Gradient Stacking Classifier

Trong đề tài khóa luận tốt nghiệp này, mô hình Gradient Stacking Classifier sẽ kết hợp cùng lúc cả ba phương pháp Boosting, Bagging, Stacking trong Ensemble Learning. Điều đó, sẽ tận dụng tối đa cách kết hợp các mô hình trong Ensemble Learning.

#### 4.4.2 Quy trình thuật toán:

Cho bộ dữ liệu Datasets có  $X = (X_1, X_2, \dots, X_n)^T$  và  $Y = (y_1, y_2, \dots, y_n)^T$  tương ứng như sau:

$$X = \begin{pmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \dots & \dots & \dots & \dots \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{pmatrix}; Y = \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix} \in \{0, 1\}^n$$

Biết rằng bộ dữ liệu trên thỏa mãn ánh xạ:

$$f: \begin{matrix} R^{n \times m} \\ (X_1, X_2, \dots, X_n) \end{matrix} \longrightarrow \begin{matrix} R^n \\ F(X_1, X_2, \dots, X_n) \end{matrix}$$

*Chú ý: Mỗi vector  $X_i = (x_{i1}, x_{i2}, \dots, x_{im})$  đại diện cho đặc trưng  $X_i$  trong bộ dữ liệu với giá trị dự đoán  $y_i$  tương ứng.*

Các bước thực hiện của thuật toán đối với bộ dữ liệu:

1. Đối với bộ dữ liệu trên, ta huấn luyện chúng đối với từng mô hình Decision Tree, Random Forest, ExtraTrees. Khi đó, ta có được bộ dữ liệu đầu ra mới như sau:

$$X' = \begin{pmatrix} X'_1 \\ X'_2 \\ \dots \\ X'_n \end{pmatrix} = \begin{pmatrix} x'_{11} & x'_{12} & x'_{13} \\ x'_{21} & x'_{22} & x'_{23} \\ \dots & \dots & \dots \\ x'_{n1} & x'_{n2} & x'_{n3} \end{pmatrix}; Y = \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix} \in \{0, 1\}^n$$

Khi đó, ta có bảng sau:

STT	DecisionTree	RandomForest	ExtraTrees	$y_i$
1	$x'_{11}$	$x'_{12}$	$x'_{13}$	$y_1$
2	$x'_{21}$	$x'_{22}$	$x'_{23}$	$y_2$
...	...	...	...	...
n	$x'_{n1}$	$x'_{n2}$	$x'_{n3}$	$y_n$

2. Với bộ dữ liệu mới ( $X'$ ,  $Y$ ) ta sẽ huấn luyện tiếp với mô hình Gradient Boosting Classifier thỏa mãn ánh xạ sau:

$$f: \begin{matrix} R^{3n} \\ (X'_1, X'_2, \dots, X'_n) \end{matrix} \longrightarrow \begin{matrix} R^n \\ F(X'_1, X'_2, \dots, X'_n) \end{matrix}$$

Trong đó, mỗi vector  $X'_i = (x'_{i1}, x'_{i2}, x'_{i3})$  đại diện cho đặc trưng  $X'_i$  trong bộ dữ liệu mới ( $X'$ ,  $Y$ ) với giá trị dự đoán  $y_i$  tương ứng.



## 5 Kết quả thực nghiệm

### 5.1 So sánh giữa các mô hình:

Model	Recall Score	Accuracy
Gradient Boosting Classifier	90%	87%
Bagging Classifier	99%	84%
Logistic Regression	61%	54%
Logistic Stacking Classifier	99%	88%
Gradient Stacking Classifier	99%	90%

Mô hình sử dụng Recall làm chỉ số đánh giá để đo lường vì mục tiêu bài toán là giảm thiểu rủi ro thẻ tín dụng => chọn chỉ số phụ thuộc vào tình hình hiện tại.

Một số thông số đánh giá khác của mô hình:

- Accuracy là độ chính xác tổng quát của mô hình với tỷ lệ dự đoán đúng trên tổng số dữ liệu:

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

- Precision là tỷ lệ dự đoán đúng Positive khi cần giảm thiểu dự đoán sai Positive (đánh giá chính xác cho khách hàng):

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

- Recall đo lường khả năng của mô hình trong việc phát hiện đúng các đối tượng thuộc lớp dương tính (không bỏ sót người có rủi ro), có công thức:

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

=> Recall không bỏ sót bất kỳ đối tượng quan trọng nào, chấp nhận một số lỗi sai (tỷ lệ phát hiện chính xác nhạy cảm)

Ngoài ra, thông số F1-Score là chỉ số kết hợp giữa Precision và Recall để đánh giá hiệu suất phân loại của mô hình trong các bài toán phân loại dữ liệu mất cân bằng, được tính theo công thức trung bình điều hòa:

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

Ví dụ: phát hiện gian lận, dự đoán rủi ro tín dụng, ...

## 5.2 Lý do lựa chọn Gradient Stacking Classifier:

### 5.2.1 Ưu điểm:

- Kết hợp hiệu quả giữa hai phương pháp Stacking và Boosting trong Ensemble Learning.
- Mô hình tổng hợp Gradient Boosting có khả năng xử lý dữ liệu phi tuyến tốt hơn dựa trên cây quyết định Decision Tree phức tạp, trong khi Logistic Regression chỉ phù hợp cho dữ liệu tuyến tính.
- Tốc độ xử lý nhanh và đạt hiệu quả tốt hơn trên các bộ dữ liệu đủ lớn so với các mô hình học sâu khác.

### 5.2.2 Nhược điểm:

- Tiêu tốn nhiều thời gian huấn luyện các mô hình khi kết hợp giữa cả hai phương pháp Stacking và Gradient Boosting
- Dễ bị overfitting khi không được điều chỉnh đúng cách

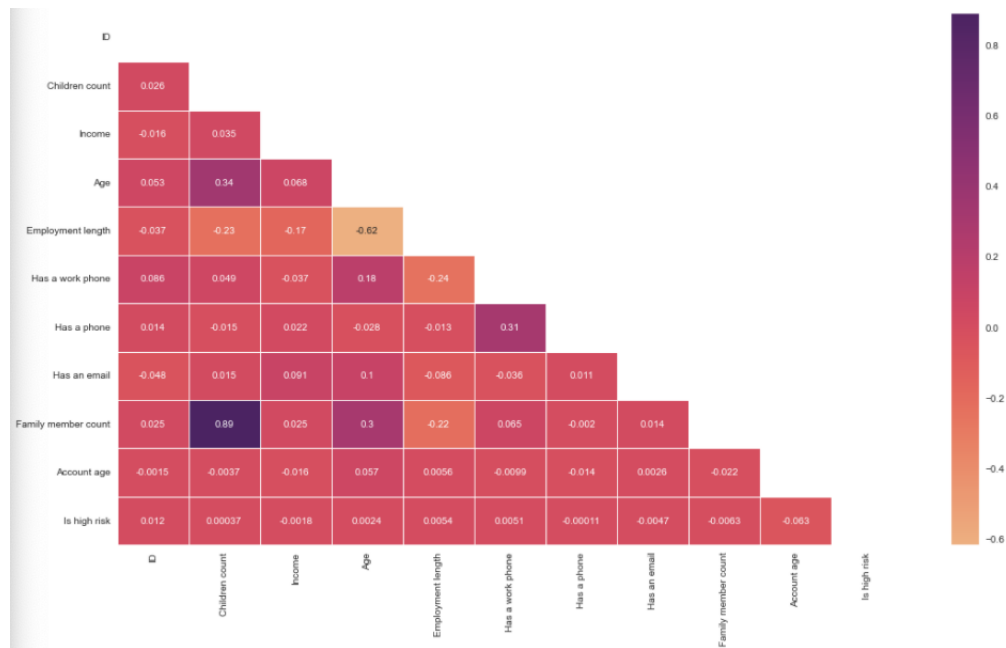
## 5.3 Kết quả thu được từ mô hình Gradient Stacking Classifier:

### 5.3.1 Mối tương quan giữa các đặc trưng:

Ma trận tương quan (correlation matrix) là một công cụ thống kê dùng để biểu thị mức độ liên quan tuyến tính giữa các đặc trưng (features) trong một bộ dữ liệu. Việc xây dựng và quan sát ma trận tương quan giúp ta nhận diện các đặc trưng có thể bị dư thừa thông tin (tương quan cao với nhau), từ đó hỗ trợ việc lựa chọn đặc trưng (feature selection) và giảm chiều dữ liệu (dimensionality reduction).

Ma trận tương quan giữa các đặc trưng trong bộ dữ liệu biểu thị mức độ liên quan giữa các đặc trưng đó với nhau:

- Giá trị trong ma trận nằm trong khoảng -1 đến +1
  - Nếu giá trị  $> 0 \Rightarrow$  tương quan đồng biến (cùng tăng, cùng giảm)
  - Nếu giá trị  $< 0 \Rightarrow$  tương quan nghịch biến (A tăng B giảm, A giảm B tăng)
  - Nếu giá trị  $= 0 \Rightarrow$  không có mối tương quan giữa hai đặc trưng
- Màu sắc càng đậm thì mối tương quan càng mạnh và ngược lại



Hình 18: Ma trận tương quan các đặc trưng bằng Heatmap

### 5.3.2 Ma trận nhầm lẫn:

Ma trận nhầm lẫn là một công cụ quan trọng trong đánh giá hiệu năng của các mô hình phân loại. Nó cung cấp cái nhìn tổng thể về cách mô hình hoạt động trên từng lớp cụ thể trong bộ dữ liệu, không chỉ đơn thuần dựa vào độ chính xác (accuracy) chung. Đối với một bài toán phân loại nhị phân (binary classification), ma trận nhầm lẫn là một bảng có kích thước  $2 \times 2$ .

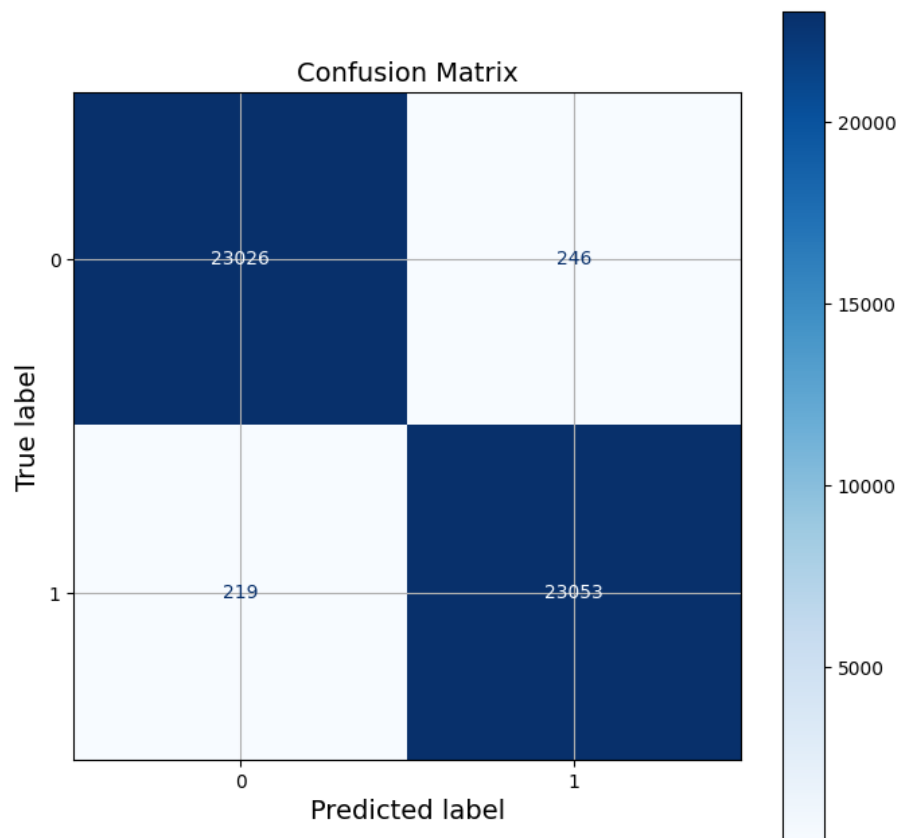
Ma trận nhầm lẫn giúp đánh giá và cải thiện hiệu suất, độ chính xác của mô hình phân loại. Chúng cung cấp sự phân tích toàn diện về các dự đoán của mô hình so với kết quả thực tế, cung cấp thông tin chi tiết về các loại và tần suất do mô hình tạo ra.

- True Positives (TP): các trường hợp mô hình dự đoán đúng lớp dương tính khi nó thực sự dương tính
- True Negatives (TN): các trường hợp mô hình dự đoán đúng lớp âm tính khi nó thực sự âm tính
- False Positives (FP): các trường hợp mô hình dự đoán sai lớp dương tính khi nó thực sự âm tính
- False Negatives (FN): các trường hợp mô hình dự đoán sai lớp âm tính khi nó thực sự dương tính

		Predicted Class		
		Positive	Negative	
Actual Class	Positive	True Positive (TP)	False Negative (FN) <b>Type II Error</b>	<b>Sensitivity</b> $\frac{TP}{(TP + FN)}$
	Negative	False Positive (FP) <b>Type I Error</b>	True Negative (TN)	<b>Specificity</b> $\frac{TN}{(TN + FP)}$
		<b>Precision</b> $\frac{TP}{(TP + FP)}$	<b>Negative Predictive Value</b> $\frac{TN}{(TN + FN)}$	<b>Accuracy</b> $\frac{TP + TN}{(TP + TN + FP + FN)}$

Hình 19: Minh họa về ma trận nhầm lẫn tổng quát

Trong đề tài khóa luận này đã đưa ra ma trận nhầm lẫn (Confusion Matrix) thông qua thuật toán phân loại Gradient Stacking Classifier như sau:

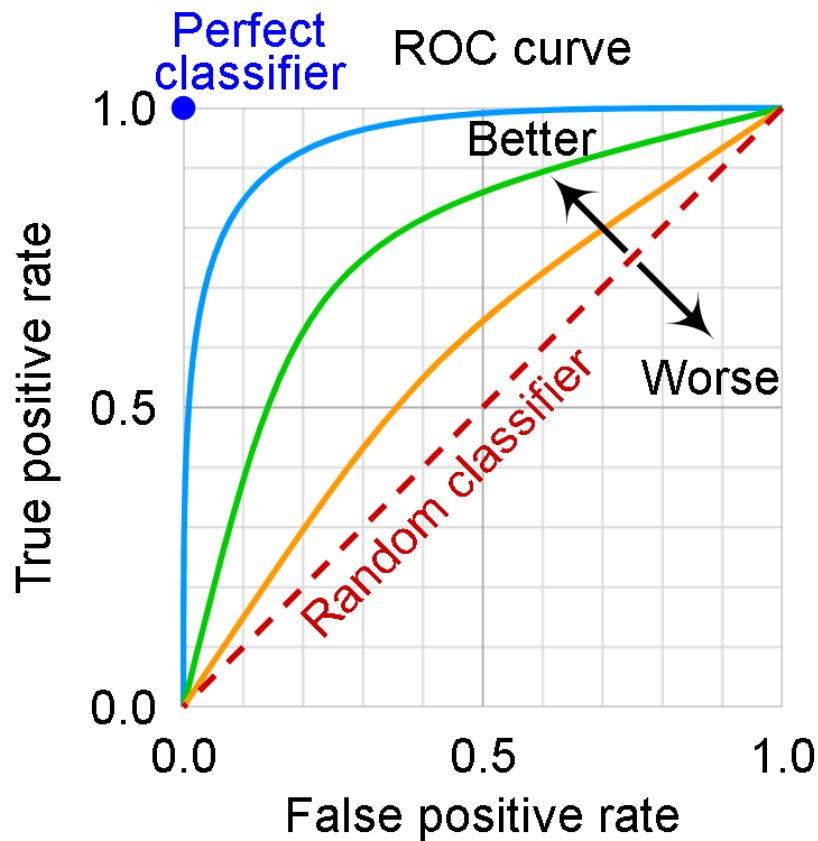


Hình 20: Ma trận nhầm lẫn của Gradient Stacking Classifier

### 5.3.3 Đường cong ROC:

ROC curve (Receiver Operating Characteristic curve) là đồ thị dùng để đánh giá hiệu suất của mô hình phân loại trong việc phân biệt giữa các lớp True/False. Với đường cong ROC này, ta có thể dễ dàng so sánh hiệu suất của các bộ phân loại khác nhau:

- Đường cong càng gần góc trên bên trái của biểu đồ mà không chạm hẳn vào góc xa, thì mô hình càng tốt
- Đường cong nằm dưới đường màu đỏ Random classifier thì hiệu suất kém hơn so với dự đoán ngẫu nhiên
- Trong trường hợp đường cong hoàn hảo chạm đúng góc Perfect classifier thì không phải là bộ phận phân loại tốt của mô hình vì xảy ra hiện tượng overfitting (quá khớp)



Hình 21: Minh họa đường cong ROC trên tập dữ liệu

- True Positive Rate = Sensitivity = Recall:

$$TPR = \frac{TP}{TP + FN}$$

- False Positive Rate:

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

Giải thích về Area Under Curve (AUC) của đường cong ROC:

- $AUC = 0$  là mô hình không tốt (phân loại sai mọi trường hợp)
- $AUC = 1$  là mô hình hoàn hảo (phân loại chính xác mọi trường hợp)
- $0.5 < AUC < 1$  là mô hình tốt hơn ngẫu nhiên
- $0 < AUC < 0.5$  là mô hình tệ hơn ngẫu nhiên

=> Khả năng phân biệt của mô hình so với phân loại ngẫu nhiên

## 5.4 Triển khai và xây dựng giao diện mô hình với Streamlit Web Interface:

Sau khi đã chọn lọc và training được mô hình phù hợp, ta sẽ thiết kế và xây dựng ra giao diện cho mô hình để cho phép người dùng có thể dễ dàng tương tác trực tiếp trên Website. Khi đó, người dùng có thể nhập dữ liệu hồ sơ đầu vào của khách hàng và kiểm tra, đánh giá xem khách hàng đó có khả năng được phê duyệt thẻ tín dụng hay không.

### 5.4.1 Ngôn ngữ lập trình Python:

Python là một ngôn ngữ lập trình bậc cao, đơn giản, dễ đọc và dễ học, được sử dụng rộng rãi trong nhiều lĩnh vực khác nhau, đặc biệt là trí tuệ nhân tạo (AI), khoa học dữ liệu (Data Science) và phát triển web.

Trong dự án này, Python đóng vai trò quan trọng như sau:

- Xây dựng mô hình Machine Learning: Sử dụng thư viện scikit-learn để huấn luyện và đánh giá mô hình Gradient Boosting Classifier.
- Xử lý dữ liệu: Sử dụng các thư viện như pandas, numpy để làm sạch, chuyển đổi và xử lý dữ liệu đầu vào.
- Triển khai giao diện: Sử dụng Streamlit để xây dựng ứng dụng web giúp người dùng nhập dữ liệu và nhận kết quả dự đoán trực tiếp.

Tóm lại, Python không chỉ là ngôn ngữ lập trình hỗ trợ việc xây dựng mô hình và giao diện, mà còn là cầu nối giữa việc phân tích và xử lý dữ liệu, trí tuệ nhân tạo. Việc lựa chọn Python là nền tảng phát triển cho đề tài khóa luận này là một quyết định hợp lý, góp phần đảm bảo hiệu quả, độ chính xác và tính mở rộng cho hệ thống trong tương lai.

### 5.4.2 Triển khai mô hình với Streamlit:

Streamlit là một framework mã nguồn mở dùng để xây dựng giao diện web cho các ứng dụng Machine Learning và Data Science bằng ngôn ngữ Python. Với Streamlit, người dùng có thể dễ dàng biến các mô hình học máy, biểu đồ trực quan hoặc phân tích dữ liệu thành các ứng dụng web tương tác chỉ với vài dòng mã không quá phức tạp.



Hình 22: Framework Streamlit của Python

Vì Framework Streamlit chỉ hỗ trợ các file có định dạng đuôi là `.py` nên ta sẽ chuyển đổi `file_name.ipynb` về định dạng chuẩn `file_name.py` trong Python.

Về phía giao diện tương tác, hệ thống thêm hồ sơ khách hàng vào tập dữ liệu huấn luyện và thực hiện toàn bộ quá trình tiền xử lý dữ liệu (Data Preprocessing). Khi đó, hàng cuối cùng được trích xuất tương ứng với hồ sơ của người đăng ký.

*Lưu ý: Hồ sơ của người đăng ký chỉ được thêm vào tập dữ liệu huấn luyện chứ không huấn luyện lại toàn bộ mô hình vì điều này có thể dẫn tới hiện tượng overfitting.*

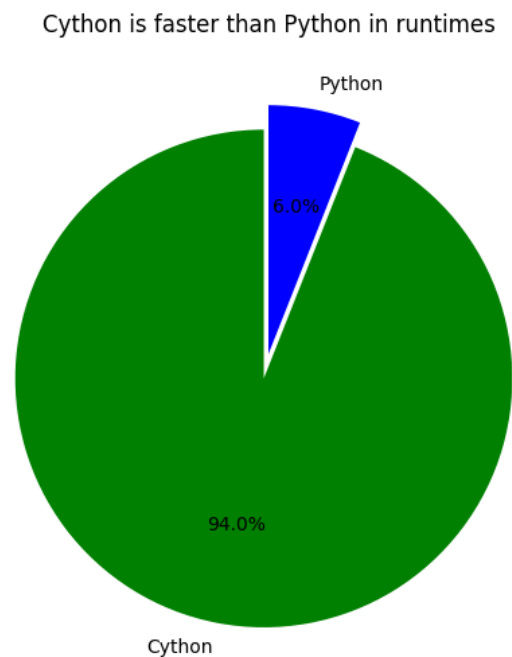
### 5.4.3 Ngôn ngữ lập trình mở rộng Cython:

Cython là một siêu ngôn ngữ lập trình Python, cho phép biên dịch mã của Python sang mã máy ngôn ngữ C/C++ nhằm tối ưu hóa hiệu suất. Ngôn ngữ Cython kết hợp cú pháp thân thiện của Python cùng với tốc độ xử lý đáng kể của C/C++, đặc biệt là trong các bài toán tính toán khoa học và xử lý dữ liệu lớn trong mô hình học máy.

Trong dự án này, Cython sử dụng để tối ưu hóa các phần quan trọng:

- Chia thành các tập dữ liệu trên bộ dữ liệu lớn
- Tối ưu hóa các hàm tiền xử lý dữ liệu (Data Preprocessing)
- Tối ưu bước dự đoán mô hình để giảm độ trễ khi đưa ra kết quả

Sau khi triển khai bằng ngôn ngữ Cython, thời gian chạy của các bước tiền xử lý giảm đáng kể, giúp cải thiện trải nghiệm người dùng khi dự đoán kết quả. Bên cạnh đó, việc tối ưu hóa pipeline dự đoán giúp hệ thống phản hồi nhanh hơn, đặc biệt khi làm việc với lượng lớn dữ liệu.



Hình 23: So sánh tốc độ xử lý dữ liệu giữa Cython và Python

Qua nhiều lần thử nghiệm, mặc dù trong một số trường hợp hiếm hoi Cython có thể chậm hơn Python do các yếu tố như quản lý bộ nhớ hoặc tối ưu hóa chưa triệt để, nhưng nhìn chung, Cython vẫn cho thấy hiệu suất vượt trội hơn. Phần lớn các lần chạy, Cython có tốc độ nhanh hơn Python, giúp cải thiện đáng kể thời gian xử lý và hiệu quả tính toán.

## 5.5 Sản phẩm đề tài khóa luận tốt nghiệp:

Sản phẩm của đề tài khóa luận này là một trang web ứng dụng học máy tích hợp giao diện tương tác người dùng, được xây dựng bằng ngôn ngữ lập trình Python kết hợp với thư viện Streamlit. Sản phẩm này nhằm mục đích dự đoán khả năng rủi ro thẻ tín dụng của khách hàng và chấm điểm thẻ tín dụng ngân hàng của khách hàng đó dựa vào hồ sơ tín dụng.



# Credit Card Approval Prediction

## Avatar Picture


Choose an image...



Drag and drop file here

Limit 200MB per file • JPG, PNG, JPEG

Browse files

 Please upload your profile image

## Thông tin cá nhân

### Fullname

Enter your fullname:

### Gender

Select your gender:

☒ Male ☐ Female

Selected Gender: Male

### Age

Enter your year of birth:

 - +

Select your age:

22

18 70

The widget with key "year\_of\_birth" was created with a default value but also had its value set via the Session State API.

Selected Age: 22 years

Year of Birth: 2003

## Thông tin tình trạng quan hệ

### Marital status

Select your marital status:

 ▼

## Family member count

Select your family member count:

1



## Thông tin tài sản và nguồn thu nhập

## Dwelling type

Select the type of dwelling:

House / apartment



## Income

Enter your income (in USD):

0

## Ownship Information

Do you own a car?



Yes



No

Do you own a property?



Yes



No

## Thông tin kinh nghiệm làm việc

## Employment Information

Select your employment status:

Working



Select your employment length:



0

30

## Education level

Select your education status:

Secondary school



Job Title (Majority):

## Thông tin liên hệ

## Work phone

Do you have a work phone?



Yes



No

Enter your work phone number:

## Phone

Do you have a phone?

☒ Yes

☐ No

Enter your phone number:

## Email

Do you have an email?

☒ Yes

☐ No

Enter your email:

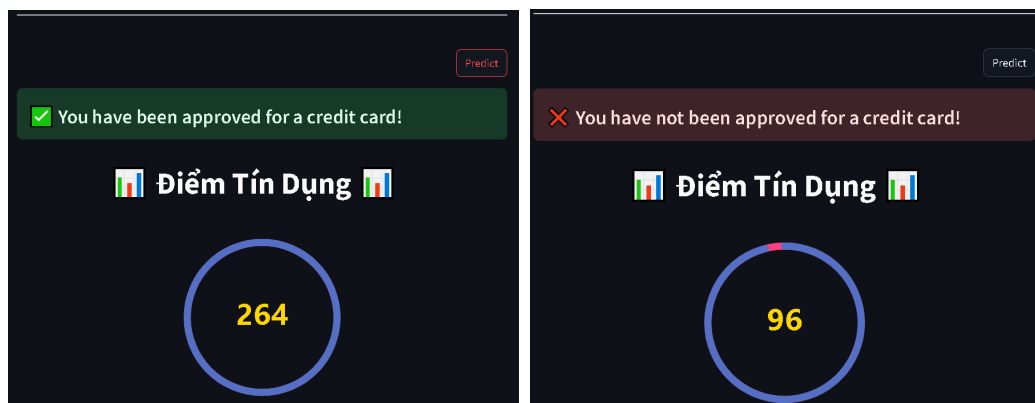
---

Predict

Tại đây, khách hàng sẽ nhập các thông tin cơ bản của hồ sơ tín dụng, sau đó nhấn nút Predict để hệ thống có thể dự đoán được khả năng rủi ro thẻ tín dụng của mình và từ đó có thể chấm điểm được thẻ tín dụng của khách hàng đó.

Điểm tín dụng và khả năng rủi ro tín dụng tỷ lệ nghịch với nhau. Trong đó, điểm tín dụng càng cao thì khả năng rủi ro càng thấp và ngược lại:

- Nếu khách hàng có khả năng rủi ro tín dụng tức là điểm tín dụng sẽ dưới mức 100 điểm
- Nếu khách hàng không có khả năng rủi ro tín dụng thì điểm tín dụng sẽ trên mức 100 điểm



Hình 24: Đánh giá sự chấp thuận và chấm điểm thẻ tín dụng ngân hàng

# KẾT LUẬN

Rủi ro tín dụng là một trong những thách thức quan trọng trong lĩnh vực ngân hàng – tài chính, đặc biệt là việc phê duyệt và chấm điểm thẻ tín dụng ngân hàng cho khách hàng. Nghiên cứu này tập trung vào việc ứng dụng của phương pháp Ensemble Learning (Boosting, Bagging, Stacking) để cải thiện độ chính xác trong việc dự đoán số lượng lớn khách hàng có nguy cơ rủi ro cao. Đặc biệt, mô hình học máy Gradient Stacking Classifier được sử dụng để phân tích và đánh giá trên tập dữ liệu phi tuyến dạng Decision Tree đạt hiệu quả khá tốt khi kết hợp cả hai thuật toán Stacking và Gradient Boosting.

Quy trình tiền xử lý và huấn luyện mô hình dựa trên các bước khai phá, phân tích dữ liệu lớn như phân tích đơn biến, đa biến cùng với các dạng biểu đồ trực quan hóa dữ liệu để lựa chọn ra những đặc trưng phù hợp quyết định độ chính xác và hiệu suất của mô hình đó. Kết quả dự kiến cho thấy Ensemble Learning giúp cải thiện khả năng nhận diện rủi ro cao, hỗ trợ ngân hàng có thêm những giải pháp hợp lý trong việc giảm thiểu rủi ro thẻ tín dụng của khách hàng.

Trong thời gian tới, đề tài khóa luận sẽ tập trung vào việc phát triển thêm các mô hình dự đoán rủi ro tín dụng mới bằng cách áp dụng những kỹ thuật phân tích tiên tiến và thử nghiệm các phương pháp khác nhau. Cụ thể, việc tìm hiểu thêm các yếu tố ảnh hưởng đến rủi ro tín dụng sẽ giúp việc chấm điểm thẻ tín dụng hiệu quả hơn. Ngoài ra, việc ứng dụng kết hợp các phương pháp học máy, học sâu và học tăng cường có thể mang lại hiệu quả đáng kể, giúp mô hình có thể học hỏi tốt hơn từ dữ liệu phức tạp và cải thiện kết quả dự báo, từ đó nâng cao khả năng đánh giá và quản lý rủi ro tín dụng trong tương lai.

Demo Web: [creditcardapprovalpredictiondeployment.streamlit.app](https://creditcardapprovalpredictiondeployment.streamlit.app)

Source Code: Credit Card Project

# Tài liệu

## Tiếng Việt:

- [1] Phạm Đình Khánh (2021), Credit Score Card with Deep AI Book, Deep AI KhanhBlog.

## Tiếng Anh:

- [2] Giulio Carlone (2021), *Introduction to Credit Risk*, CRC Press, Boca Raton United States of America.
- [3] Khan Academy (2025), Multivariable Calculus Tutorial.
- [4] Stern Semasuka (2022), Blog for Credit Card Approval Prediction, Machine Intelligence Blog.
- [5] Ng Yong Kad (2020), Credit Scoring Development Using R, RPubs by RStudio.
- [6] Kaggle (2024), Free Datasets for Data Science Community.

**TÓM TẮT KHOÁ LUẬN TỐT NGHIỆP NĂM HỌC 2025**  
**ỨNG DỤNG ENSEMBLE LEARNING TRONG RỦI RO TÍN DỤNG**  
**NGÂN HÀNG**

Họ và tên sinh viên: Tạ Quang Tùng

Ngày sinh: 06/03/2003

Mã SV: 21000712

Khoá: QH.2021.T

Khoa: Toán – Cơ – Tin học

Họ và tên cán bộ hướng dẫn: Hoàng Thị Phương Thảo

Tóm tắt nội dung khoá luận tốt nghiệp:

Rủi ro tín dụng là một trong những thách thức lớn đối với các tổ chức tài chính, đặc biệt trong quá trình xét duyệt và chấm điểm thẻ tín dụng cho khách hàng. Khóa luận này tập trung vào việc ứng dụng phương pháp học máy Ensemble Learning nhằm cải thiện hiệu quả dự đoán rủi ro tín dụng, từ đó hỗ trợ ngân hàng trong việc ra quyết định chính xác hơn. Cụ thể, đề tài triển khai mô hình Gradient Stacking Classifier, kết hợp các mô hình cơ sở như Decision Tree, Random Forest và Extra Trees, với Gradient Boosting đóng vai trò là mô hình tổng hợp. Mô hình được huấn luyện trên tập dữ liệu phi tuyến thông qua quy trình tiền xử lý dữ liệu bao gồm phân tích đơn biến, đa biến, trực quan hóa bằng biểu đồ và lựa chọn đặc trưng phù hợp để tối ưu hóa hiệu suất mô hình. Thông qua thực nghiệm, Gradient Stacking Classifier cho thấy khả năng nhận diện khách hàng có nguy cơ rủi ro cao với độ chính xác cao hơn so với các mô hình học máy đơn lẻ. Kết quả cho thấy phương pháp Ensemble Learning không chỉ giúp cải thiện khả năng phân loại mà còn góp phần xây dựng hệ thống chấm điểm tín dụng hiệu quả hơn, từ đó hỗ trợ ngân hàng trong việc giảm thiểu rủi ro tín dụng và nâng cao chất lượng quản lý khách hàng. Khóa luận kỳ vọng sẽ góp phần nhỏ vào việc áp dụng các kỹ thuật học máy hiện đại vào lĩnh vực tài chính – ngân hàng trong thực tế.

Từ khoá: Rủi ro tín dụng; chấm điểm tín dụng; Ensemble Learning; Gradient Stacking Classifier; tài chính – ngân hàng.